

RK611

DISKLESS CONT NO. 4
MD-11-DZR6D-A

EP-DZR6D-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

DEC 1976
digital
MADE IN USA

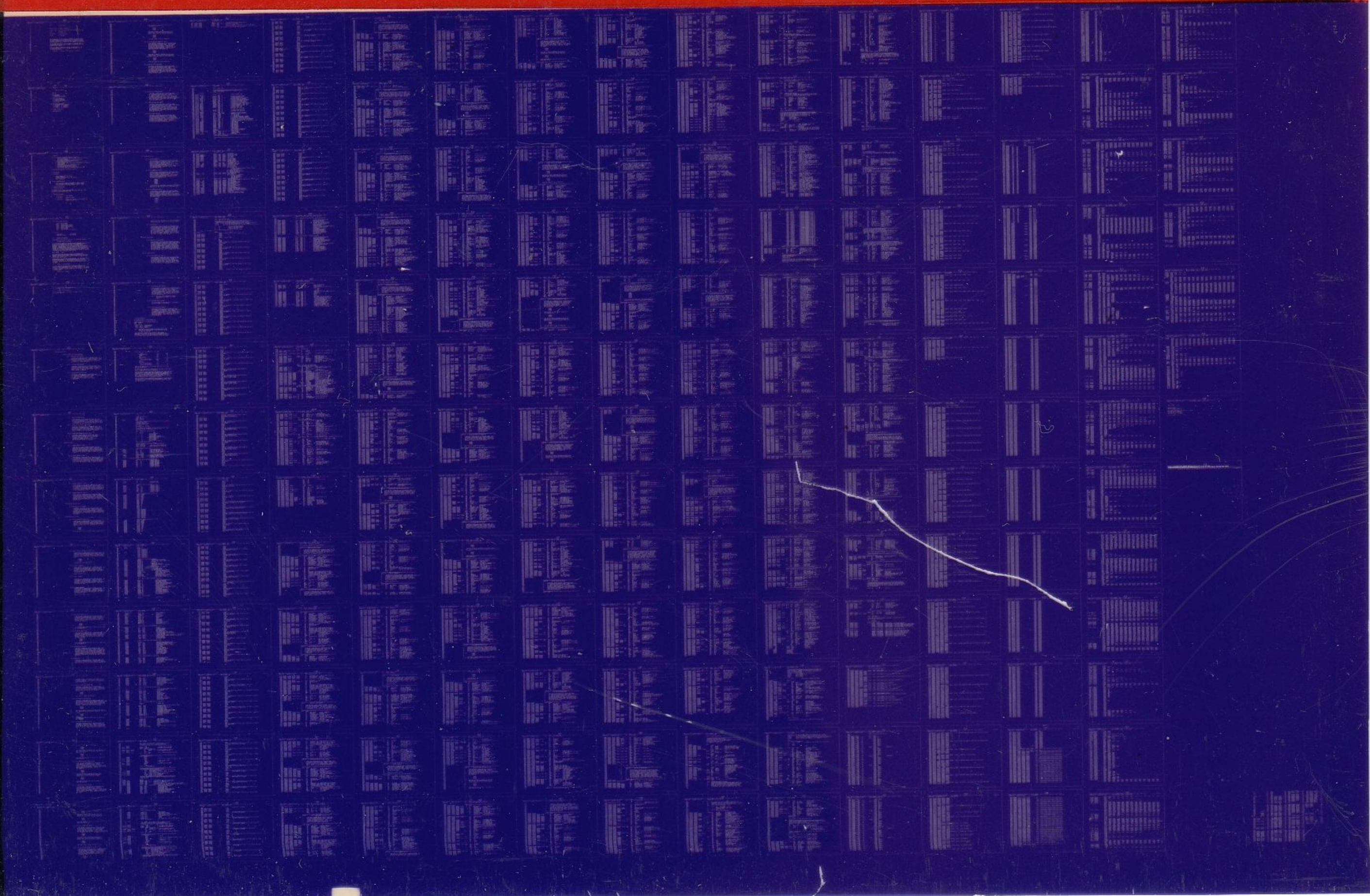


TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PROGRAMS
- 3.0 OPERATING PROGRAMS
 - 3.1 LOADING PROCEDURE
 - 3.2 STARTING PROCEDURE
 - 3.3 OPTIONAL SWITCH SETTING
 - 3.4 RUN TIME
- 4.0 OPERATING PROCEDURES
 - 4.1 "SOFTWARE" SWITCH REGISTER
 - 4.2 CONTROL C (1C) OPERATION
 - 4.3 CONTROL S (1S) OPERATION
 - 4.4 CONTROL Q (1Q) OPERATION
- 5.0 PROGRAM DESCRIPTION
- 6.0 ERROR REPORTING

UNIVERSITY MICROFILMS
SERIALS ACQUISITION
300 N ZEEB RD
ANN ARBOR MI 48106

108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160

3.3 OPTIONAL SWITCH SETTINGS

- SW15 - HALT PROGRAM
- SW14 - LOOP ON TEST
- SW13 - INHIBIT ERROR TYPE OUT
- SW12 - ABORT AFTER 20 ERRORS
- SW11 - INHIBIT ITERATION COUNT
- SW10 - BELL ON ERROR
- SW9 - LOOP ON ERROR
- SW8 - LOOP ON TEST IN SWITCHES 0-7

3.4 RUN TIME

FIRST PASS	:25 MINUTES
SUBSEQUENT PASSES	3:15 MINUTES

4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPRIATE ADDRESS.

4.1 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E., AN 11/04 OR 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RK06 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

SWR = NNNNNN NEW ='

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

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

4.2 CONTROL C (↑C) OPERATION

IF ↑C IS TYPED AT ANY TIME DURING THE PROGRAM EXECUTION THE PROGRAM IS HALTED IMMEDIATELY. IF A MONITOR IS PRESENT (XXDP CHAIN, ACT, APT) THE PROGRAM RETURNS CONTROL TO THE MONITOR. IF NO MONITOR IS PRESENT, THE CPU IS HALTED. DEPRESSING THE CONTINUE KEY WILL DO A PROGRAM RESTART.

4.3 CONTROL S (↑S) OPERATION

F01

161
162
163
164
165
166
167
168
169

IF 'S IS TYPED AT ANY TIME THE PROGRAM WILL GO INTO A STALL LOOP
UNTIL A CONTROL Q ('Q) IS TYPED.

4.4 CONTROL Q ('Q) OPERATION

IF A 'S HAS BEEN TYPED, TYPING THE 'Q CANCELS THE STALL
INITIATED BY THE 'S.

5.0 PROGRAM DESCRIPTION

**TYPE C INSTRUCTION'S DRIVE MESSAGES

TEST 1 READ DATA SEEK MESSAGE

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA TO AN RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND MAKE SURE IT IS GENERATED PROPERLY.

TEST 2 WRITE DATA SEEK MESSAGE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE. CHECK IF PROPER MESSAGE IS ASSEMBLED.

TEST 3 SEEK MESSAGE FOR WRITE CHECK

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK IN SEEK COMMAND. MAKE SURE CORRECT MESSAGE IS ASSEMBLED.

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

H01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA
DZR6DA.P11 28-SEP-76 15:50

MACY11 27(1006) 05-OCT-76 09:06 PAGE 7

SEQ 0007

203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250

TEST 4 READ DATA CLEAR MESSAGE

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE CONTROLLER ON DIAGNOSTIC MODE. ISSUE A READ DATA TO AN RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT

TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE AND DRIVE CLEAR CHECK IF PROPER DRIVE CLEAR MESSAGE IS ASSEMBLED.

TEST 6 DRIVE CLEAR MESSAGE FOR WRITE CHECK

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR. MAKE SURE CORRECT MESSAGE IS ASSEMBLED.

**HEADER GENERATION

TEST 7 HEADER GENERATION (PART 1)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0 SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3 TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR CYLINDER 1-1777.

TEST 10 HEADER GENERATION (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3 TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR HEADS 1-

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

TEST 11 HEADER GENERATION (PART 3)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR SECTORS 1-25

TEST 12 HEADER GENERATION (PART 4)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR 2 SECTOR FORMAT.

**NPR TRANSFER FOR WRITE DATA

TEST 13 WRITE DATA NPR TRANSFER

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 67 WORDS, TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, TRACK 0, DRIVE 0. CLOCK IN SEEK AND DRIVE CLEAR MESSAGE GIVE ENOUGH CLOCK PULSE FOR 66 SILO WORDS. MAKE SURE DAT LATE DOES NOT OCCUR. READ BACK 66 WORDS AND VERIFY THEY ARE CORRECT.

**HEADER RECOGNITION TESTS

TEST 14 WRITE DATA HEADER RECOGNITION

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
140000
140000

MAKE SURE WRITE GATE SETS SHOWING CORRECT HEADER RECOGNI

310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363

TEST 15 SECTOR PULSE DETECTION FOR WRITE DATA

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE AN INDEX PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
140000
140000

MAKE SURE WRITE GATE DOES NOT SET.

TEST 16 SECTOR INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0 TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE AND PROPER HEADER. MAKE SURE THAT WRITE GATE SETS.

REPEAT FOR SECTOR 1-24.

TEST 17 TRACK INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0 TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE AND PROPER HEADER. MAKE SURE THAT WRITE GATE SETS.

TEST 20 CYLINDER INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0 TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2, SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE AND PROPER HEADER. MAKE SURE THAT WRITE GATE SETS.

REPEAT FOR CYLINDER = 1-632.

K01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA
DZR6DA.P11 28-SEP-76 15:50

MACY11 27(1006) 05-OCT-76 09:06 PAGE 10

SEG 0010

364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419

TEST 21 BAD SECTOR ERROR (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
040000
040000

MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS IS NOT INCREMENTED.

TEST 22 BAD SECTOR ERROR (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
100000
100000

MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS IS NOT INCREMENTED.

TEST 23 OPERATION INCOMPLETE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253, HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH 1 BIT DIFFERENT IN 30 BITS OF OPI DETERMINATION. ALL SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS THAT SET.

TEST 24 HEADER VRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1253, HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR

L01

MESSAGES SIMULATE A SECTOR PULSE AND HEADER WITH BIT 0 OF THE VRC INCORRECT. MAKE SURE ONLY HEADER VRC AND CONTROLLER ERROR ARE THE ONLY ERRORS SET. REPEAT FOR BITS 1-15 OF VRC.

TEST 25 BAD SECTOR ERROR AND HEADER VRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300, HEAD 1, SECTOR 17. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE THE FOLLOWING HEADER:

000300
040057
040356

MAKE SURE ONLY HEADER VRC ERROR SETS.

TEST 26 GOOD HEADER AND PREVIOUS BSE

CLEAR RK611 CONTROLLER WITH CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 100, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING 3 WORDS WITH A BAD SECTOR INDICATION:

000100
040000
040100

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOW 3 WORDS:

000100
140001
140101

MAKE SURE WRITE GATE SETS INDICATING THAT HEADER HAS BEEN RECOGNIZED.

TEST 27 GOOD HEADER AND PREVIOUS HVRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 200, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE

40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

MO1

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA
DZR6DA.P11 28-SEP-76 15:50

MACY11 27(1006) 05-OCT-76 09:06 PAGE 12

SEQ 0012

FOLLOWING 3 WORDS WITH A BAD HEADER VRC:

000200
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING 3 WORDS:

000200
140001
140201

MAKE SURE WRITE GATE SEES INDICATING THAT HEADER HAS BEEN RECOGNIZED.

TEST 30 BAD SECTOR ERROR AND PREVIOUS HVRC

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 400, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING 3 WORDS WITH A BAD HEADER VRC:

000400
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT SET. SIMULATE A SECTOR PULSE AND A HEADER CONSISTING OF THE FOLLOWING 3 WORDS:

000400
040001
040401

MAKE SURE BAD SECTOR ERROR SETS AND HEADER VRC ERROR DOES NOT SET.

TEST 31 HEADER VRC AND PREVIOUS BSE

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OR ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 140, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING 3 WORDS WITH A HEADER VRC ERROR:

475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528

530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583

000140
040000
040140

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT SET. STIMULATE A SECTOR PULSE AND A HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000140
140001
140101

MAKE SURE HEADER VRC ERROR SETS AND BAD SECTOR ERROR DOES NOT SET.

TEST 32 OPI AND HVRC ON LAST HEADER

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 240, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:

000240
140000
140240

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT SET. SIMULATE A SECTOR PULSE AND A HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000240
140000
140040

MAKE SURE HEADER VRC AND OPI ERROR SET.

TEST 33 OPI AND PREVIOUS HEADER VRC (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300, HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:

000300
140000
140300

586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640

THEN SIMULATE A 3 WORD HEADER CONSISTING ON
THE FOLLOWING DATA:

000300
140000
140200

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000300
140000
140300

MAKE SURE HEADER VRC AND OPI ERRORS SET.

TEST 34 OPI AND PREVIOUS HEADER VRC (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 40, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
PULSE AND A HEADER CONSISTING OF THE FOLLOWING
THREE WORDS HAVING A BAD HEADER VRC:

000040
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND 31
HEADERS CONSISTING OF THE FOLLOWING THREE WORDS:

000040
140000
140040

MAKE SURE HEADER VRC AND OPI ERRORS SET.

TEST 35 BSE AND CONTROLLER ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A BSE ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

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

TEST 36 HVRC AND CONTROLLER ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

TEST 37 READ DATA AND HVRC ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

TEST 40 WRITE CHECK AND HVRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
A HEADER WITH A HEADER VRC ERROR. MAKE SURE
HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
AND MAKE SURE CONTROLLER ERROR RESETS.

681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
73E

**ECC GENERATION TESTS

TEST 41 ECC INITIALIZATION

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 10 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY 10
DATA WORDS OF ZEROES. MAKE SURE THE ECC PATTERN
REGISTER REMAINS ZERO.

TEST 42 ECC GENERATION (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY THE
FOLLOWING SIX WORDS OF DATA:

005001
040040
020004
000064
00000C
000000

CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

TEST 43 ECC GENERATION (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH ONLY THE FOLLOWING
SIX WORDS OF DATA:

177777
177777
177777
177777
177777
177777

27

737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786

CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

TEST 44 ECC WRITING

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS AND THE TWO ECC WORDS. MAKE SURE THE TWO ECC WORDS ARE CORRECT AND WRITTEN PROPERLY. CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECTOR.

**PARTIAL WRITE DATA

TEST 45 ZERO FILL ON WRITE DATA

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS AND THE TWO ECC WORDS. CHECK THE SECTOR FOR ZERO FILL AND MAKE SURE THE TWO ECC WORDS ARE WRITTEN PROPERLY. CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECT

**18 BIT FORMAT WRITES

TEST 46 18 BIT WRITE DATA (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777. VERIFY THAT TWO ZERO BITS ARE PRESENT FOR EACH WORD.

787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842

TEST 47 18 BIT WRITE DATA (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777
WITH BAD PARITY SET. VERIFY WRITING OF TWO PARITY
BITS ON SIMULATED DISK.

NOTE: THIS TEST IS ONLY EXECUTED IF MEMORY
PARITY ENABLE IS PRESENT FOR BUFFER
LOCATION.

TEST 50 CLEAR BAD PARITY BUFFER

THE SOLE PURPOSE OF THIS TEST IS TO CLEAR BADPAR
BUFFER AND REMOVE THE BAD PARITY.

TEST 51 18 BIT WRITE DATA (PART 3)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 18 BIT WORDS
AND THE 32 BIT ECC. VERIFY THAT THE ECC IS
WRITTEN CORRECTLY.

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION

TEST	ERROR	
NUM	PC	
XXXXXX	YYYYYY	
EXPECT	ACTUAL	OTHER PERTENANT
REG	REG	INFORMATION
ZZZZZZ	WWWWWW	AAAAAA

NOTE: MORE THAN ONE SET OF EXPECT/ACTUAL REGISTERS MAY BE
PRINTED OUT. OTHER PERTENANT INFORMATION MAY CONSIST
OF MORE THAN ONE WORD.

OTHER PERTINENT INFORMATION MAY CONTAIN A WORD LABELED
"BIT COUNT". THIS COUNT IS REPORTED WHEN THE OPERATION BEING
PERFORMED INVOLVES CLOCKING THE CONTROLLER THROUGH ONE OR MORE
OF THE VARIOUS FIELDS THAT MAKE UP THE PHYSICAL SECTOR FORMAT.

04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76

THESE FIELDS (ALL VALUES GIVEN IN OCTAL) ARE:

FIELD	BITS	WORDS
HEADER PREAMBLE	400	20
HEADER	60	3
GAP	100	4
DATA PREAMBLE	400	20
DATA		
(22(10) SECTOR/TRACK)	10000	400
(20(10) SECTOR/TRACK)	11000	400
ECC	40	2
POSTAMBLE	20	1
GAP		
(22(10) SECTOR/TRACK)	160	7
(20(10) SECTOR/TRACK)	140	6

REFER TO THE RK06 UNIBUS DISK SUBSYSTEM SPECIFICATION FOR MORE DETAILED DESCRIPTION.

THE "BIT COUNT" REPORTED IS INITIALIZED AT THE START OF EACH FIELD AND IS INCREMENTED FOR EACH BIT PROCESSED.

WHEN THE OPERATION BEING PERFORMED INVOLVES WRITING, OTHER PERTINENT INFORMATION MAY CONTAIN WORDS LABELED PRESENT BIT, PRESENT BIT -1, PRESENT BIT -2, AND PRESENT BIT +1. THESE BITS ARE PRESENTED IN THIS WAY TO SHOW THE WRITE DATA STREAM IN THE SAME MANNER THAT THE RK611 LOOKS AT THE DATA STREAM TO COMPUTE PRECOMPENSATION ADVANCE AND PRECOMPENSATION DELAY IN THE WRITE OPERATION.

WHEN THE OPERATION BEING PERFORMED INVOLVES READING, OTHER PERTINENT INFORMATION MAY CONTAIN PREVIOUS BIT AND PRESENT BIT WORDS. THESE BITS RELATE TO RK611 LOGIC THAT DETERMINES WHEN THE BIT IS VALID FROM THE DECODER.

```

877
878
879
880
881      167400
882      000001
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908      001100
909
910
911
912
913      000011
914      000012
915      000015
916      000200
917      177776
918
919      177774
920      177772
921      177570
922      177570
923
924
925      000000
926      000001
927      000002
928      000003
929      000004
930      000005
931      000006
932      000007

%
.NLIST  CND,MD,MC,TOC
.LIST   ME
.ENABL  ABS,AMA
$SWR=   167400
$TN=    1
.TITLE  RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA
:*COPYRIGHT (C) 1976
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY ROY SPITZER
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
:*
.SBTTL  OPERATIONAL SWITCH SETTINGS
:*
*      SWITCH          USE
*-----
*      15             HALT ON ERROR
*      14             LOOP ON TEST
*      13             INHIBIT ERROR TYPEOUTS
*      12             ABORT PROGRAM AFTER 20 ERRORS
*      11             INHIBIT ITERATIONS
*      10             BELL ON ERROR
*      9              LOOP ON ERROR
*      8              LOOP ON TEST IN SWR<7:0>
.SBTTL  BASIC DEFINITIONS
:*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
:*MISCELLANEOUS DEFINITIONS
MT=     11             ;;CODE FOR HORIZONTAL TAB
LF=     12             ;;CODE FOR LINE FEED
CR=     15             ;;CODE FOR CARRIAGE RETURN
CRLF=   200           ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS=     177776        ;;PROCESSOR STATUS WORD
.EQUIV  PS,PSW
STKLMT= 177774        ;;STACK LIMIT REGISTER
PIRQ=   177772        ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR=   177570        ;;HARDWARE SWITCH REGISTER
DDISP=  177570        ;;HARDWARE DISPLAY REGISTER
:*GENERAL PURPOSE REGISTER DEFINITIONS
R0=     %0            ;;GENERAL REGISTER
R1=     %1            ;;GENERAL REGISTER
R2=     %2            ;;GENERAL REGISTER
R3=     %3            ;;GENERAL REGISTER
R4=     %4            ;;GENERAL REGISTER
R5=     %5            ;;GENERAL REGISTER
R6=     %6            ;;GENERAL REGISTER
R7=     %7            ;;GENERAL REGISTER

```

933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988

000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

SP= %6 ;;STACK POINTER
PC= %7 ;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
PRO= 0 ;;PRIORITY LEVEL 0
PRI= 40 ;;PRIORITY LEVEL 1
PR2= 100 ;;PRIORITY LEVEL 2
PR3= 140 ;;PRIORITY LEVEL 3
PR4= 200 ;;PRIORITY LEVEL 4
PR5= 240 ;;PRIORITY LEVEL 5
PR6= 300 ;;PRIORITY LEVEL 6
PR7= 340 ;;PRIORITY LEVEL 7

.*"SWITCH REGISTER" SWITCH DEFINITIONS
SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09, SW9
.EQUIV SW08, SW8
.EQUIV SW07, SW7
.EQUIV SW06, SW6
.EQUIV SW05, SW5
.EQUIV SW04, SW4
.EQUIV SW03, SW3
.EQUIV SW02, SW2
.EQUIV SW01, SW1
.EQUIV SW00, SW0

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)
BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4


```

989      000002      BIT01= 2
990      000001      BIT00= 1
991      .EQUIV      BIT09,BIT9
992      .EQUIV      BIT08,BIT8
993      .EQUIV      BIT07,BIT7
994      .EQUIV      BIT06,BIT6
995      .EQUIV      BIT05,BIT5
996      .EQUIV      BIT04,BIT4
997      .EQUIV      BIT03,BIT3
998      .EQUIV      BIT02,BIT2
999      .EQUIV      BIT01,BIT1
1000     .EQUIV      BIT00,BIT0

```

```

1001     .: *BASIC "CPU" TRAP VECTOR ADDRESSES
1002     000004     ERRVEC= 4      ;: TIME OUT AND OTHER ERRORS
1003     000010     RESVEC= 10     ;: RESERVED AND ILLEGAL INSTRUCTIONS
1004     000014     TBITVEC=14     ;: "T" BIT
1005     000014     TRTVEC= 14     ;: TRACE TRAP
1006     000014     BPTVEC= 14     ;: BREAKPOINT TRAP (BPT)
1007     000014     IOTVEC= 20     ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
1008     000020     PWRVEC= 24     ;: POWER FAIL
1009     000024     EMTVEC= 30     ;: EMULATOR TRAP (EMT) **ERROR**
1010     000030     TRAPVEC=34     ;: "TRAP" TRAP
1011     000034     TKVEC= 60      ;: TTY KEYBOARD VECTOR
1012     000060     TPVEC= 64      ;: TTY PRINTER VECTOR
1013     000064     PIRQVEC=240    ;: PROGRAM INTERRUPT REQUEST VECTOR
1014     000114     MEMVEC= 114    ;: MEMORY PARITY VECTOR
1015     172100     MEMBAS= 172100 ;: MEMORY PARITY BUS ADDRESS
1016     000001     PAR.EN= 1      ;: ENABLE MEMORY PARITY
1017     000005     WR.PAR= 5      ;: WRITE BAD PARITY
1018     000210     AVECT1= 210    ;: DEFINE RK611 VECTOR ADDRESS
1019     000240     APRIOR= PRS    ;: DEFINE RK611 PRIORITY
1020     177440     ABASE= 177440 ;: DEFINE BASE OF RK611 REGISTERS

```

```

1021     .SBTTL    RK611 CONTROLLER REGISTER DEFINITION
1022
1023     000000     RKCS1= 0      ;: CONTROL AND STATUS REGISTER 1
1024     000002     RKWC= 2      ;: WORD COUNT REGISTER
1025     000004     RKBA= 4      ;: BUS ADDRESS REGISTER
1026     000006     RKDA= 6      ;: DESIRED TRACK SECTOR REGISTER
1027     000010     RKCS2= 10     ;: CONTROL AND STATUS REGISTER 2
1028     000012     RKDS= 12     ;: DRIVE STATUS REGISTER
1029     000014     RKER= 14     ;: ERROR REGISTER
1030     000016     RKASOF= 16    ;: ATTENTION SUMMARY AND OFFSET REGISTER
1031     000020     RKDCYL= 20    ;: DESIRED CYLINDER REGISTER
1032     000024     RKDB= 24     ;: DATA BUFFER
1033     000026     RKMR1= 26    ;: MAINTENANCE REGISTER 1
1034     000034     RKMR2= 34    ;: MAINTENANCE REGISTER 2
1035     000036     RKMR3= 36    ;: MAINTENANCE REGISTER 3
1036     000030     RKECPS= 30   ;: ECC POSITION INFORMATION
1037     000032     RKECPT= 32   ;: ECC PATTERN INFORMATION
1038     000022     RKSPAR= 22   ;: SPARE REGISTER

```

```

1039     .SBTTL    DRIVE COMMANDS
1040
1041     000001     SELDRV= 01     ;: SELECT DRIVE

```

1045	000003	PACK=	03	:PACK ACKNOWLEDGE
1046	000005	CLEAR=	05	:DRIVE CLEAR
1047	000007	UNLOAD=	07	:UNLOAD
1048	000011	SRTSPL=	11	:START SPINDLE
1049	000013	RECAL=	13	:RECALIBRATE
1050	000015	OFFSET=	15	:OFFSET
1051	000017	SEEK=	17	:SEEK
1052	000021	RDDATA=	21	:READ DATA
1053	000023	WRDATA=	23	:WRITE DATA
1054	000025	RDHEAD=	25	:READ HEADER
1055	000027	WRHEAD=	27	:WRITE HEADER AND DATA
1056	000031	WRTCHK=	31	:WRITE CHECK
1057	000300	INTR=	300	:GENERATE INTERRUPT TO CPU

.SBTTL CONTROL AND STATUS REGISTER 1 BITS

1061	000001	GO=	BIT0	:GO BIT
1062	000100	IE=	BIT6	:INTERRUPT ENABLE
1063	000200	RDY=	BIT7	:CONTROLLER READY
1064	000400	BA16=	BIT8	:BUS ADDRESS BIT 16
1065	001000	BA17=	BIT9	:BUS ADDRESS BIT 17
1066	002000	CDT=	BIT10	:CONTROLLER DRIVE TYPE (0=RK06)
1067	004000	CTO=	BIT11	:CONTROLLER T.MED OUT WAITING FOR : DRIVE RESPONSE
1069	010000	CFMT=	BIT12	:CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1070	020000	SPAR=	BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1071	040000	DI=	BIT14	:DRIVE INTERRUPT
1072	100000	CERR=	BIT15	:CONTROLLER ERROR
1073	100000	CCLR=	BIT15	:CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1077	000007	DRVMSK=	7	:MASK FOR DRIVE SELECTION CODE
1078	000010	RLS=	BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
1079	000020	BAI=	BIT4	:BUS ADDRESS INCREMENT INHIBIT
1080	000040	SCLR=	BIT5	:CLEAR CONTROLLER AND ALL DRIVES
1081	000100	IR=	BIT6	:INPUT READY
1082	000200	OR=	BIT7	:OUTPUT READY
1083	000400	UFE=	BIT8	:UNIT FIELD ERROR
1084	001000	MDS=	BIT9	:MULTIPLE DRIVE SELECT
1085	002000	PGE=	BIT10	:PROGRAMMING ERROR
1086	004000	NEM=	BIT11	:NON-EXISTENT MEMORY
1087	010000	NED=	BIT12	:NON-EXISTENT DRIVE
1088	020000	UPE=	BIT13	:UNIBUS PARITY ERROR
1089	040000	WCE=	BIT14	:WRITE CHECK ERROR
1090	100000	DLT=	BIT15	:DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1094	000001	ILF=	BIT0	:ILLEGAL FUNCTION CODE
1095	000002	SKI=	BIT1	:SEEK INCOMPLETE
1096	000004	NXF=	BIT2	:NON-EXECUTABLE DRIVE FUNCTION
1097	000010	DRPAR=	BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
1098	000020	FMTE=	BIT4	:FORMAT ERROR
1099	000040	DTYPE=	BIT5	:DRIVE TYPE ERROR
1100	000100	ECH=	BIT6	:ECC HARD

1101	000200	BSE=	BIT7	:BAD SECTOR ERROR
1102	000400	HVRC=	BIT8	:HEADER VRC ERROR
1103	001000	COE=	BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1104	002000	IDAE=	BIT10	:INVALID DISK ADDRESS ERROR
1105	004000	WLE=	BIT11	:WRITE LOCK ERROR
1106	010000	DTE=	BIT12	:DRIVE TIMING ERROR
1107	020000	OPT=	BIT13	:OPERATION (SEARCH) INCOMPLETE
1108	040000	UNS=	BIT14	:DRIVE UNSAFE
1109	100000	DCK=	BIT15	:DATA CHECK
1110		.SBTTL STATUS REGISTER BIT DEFINITION		
1111				
1112				
1113	000001	DRA=	BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF THIS BIT IS RESET)
1114				
1115	000004	OFST=	BIT2	:DRIVE OFFSET
1116	000010	ACLO=	BIT3	:AC LOW
1117	000020	SPDLSS=	BIT4	:SPEED LOSS
1118	000040	DROT=	BIT5	:DRIVE OFF TRACK
1119	000100	VV=	BIT6	:VOLUME VALID
1120	000200	DRDY=	BIT7	:DRIVE READY
1121	000400	DDT=	BIT8	:DRIVE TYPE (0=RK06)
1122	004000	WRL=	BIT11	:WRITE LOCK
1123	020000	PIP=	BIT13	:POSITIONING IN PROGRESS
1124	040000	DSC=	BIT14	:DRIVE STATUS CHANGE
1125	100000	SVAL=	BIT15	:STATUS VALID
1126				
1127		.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION		
1128				
1129				
1130	000017	MESMSK=	17	:MESSAGE MASK
1131				
1132	000020	PAT=	BIT4	:FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1133	000040	DMD=	BIT5	:DIAGNOSTIC MODE
1134	000100	MSP=	BIT6	:MAINTENANCE SECTOR PULSE
1135	000200	MIND=	BIT7	:MAINTENANCE INDEX
1136	000400	MCLK=	BIT8	:MAINTENANCE CLOCK
1137	001000	MERD=	BIT9	:MAINTENANCE ENCODED READ DATA
1138	002000	MEWD=	BIT10	:MAINTENANCE ENCODED WRITE DATA
1139	004000	PCA=	BIT11	:PRECOMPENSATION ADVANCE
1140	010000	PCD=	BIT12	:PRECOMPENSATION DELAY
1141	020000	ECCW=	BIT13	:ECC WORD IS BEING READ OR WRITTEN
1142	040000	WRTGAT=	BIT14	:WRITE GATE
1143	100000	RDGATE=	BIT15	:READ GATE
1144		.SBTTL TRANSMITTED MESSAGE A		
1145				
1146	000020	S. SEEK=	BIT4	:SEEK COMMAND
1147	000040	S. RECL=	BIT5	:RECALIBRATE COMMAND
1148	000100	S. STSP=	BIT6	:START SPINDLE COMMAND
1149	000200	S. RTC=	BIT7	:DRIVE RETURN TO CENTERLINE COMMAND
1150	000400	S. CLR=	BIT8	:CLEAR ERROR AND DSC
1151	001000	S. FMT=	BIT9	:FORMAT
1152	002000	S. UNLD=	BIT10	:UNLOAD
1153	004000	S. PACK=	BIT11	:SET VOLUME VALID (PACK ACKNOWLEDGE)
1154		.SBTTL TRAP CATCHER		
1155				
1156	000000		.=0	

M02

```

1157      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1158      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1159      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1160      .=174
1161 000174 000000  DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1162 000176 000000  SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
1163      .SBTTL  STARTING ADDRESS(ES)
1164 000200 000137 003334  JMP      2*START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1165 000204 000137 003324  JMP      RESTRT  ;;JUMP TO RESTART ROUTINE
1166      .=214
1167 000214 000137 003314  JMP      PARM    ;JUMP TO OPERATOR ASSIGNED PARAMETERS
1168      .SBTTL  ACT11 HOOKS
1169
1170      ;;*****
1171      ;HOOKS REQUIRED BY ACT11
1172      $SVPC=.      ;SAVE PC
1173      .=46
1174 000046 034076  SENDAD      ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOF
1175      .=52
1176 000052 000000  .WORD 0      ;;2)SET LOC.52 TO ZERO
1177      .=$SVPC      ;; RESTORE PC
1178      .=MEMVEC
1179 000114 034530  MEMERR      ;MEMPARITY ERROR HANDLER
1180 000116 000340  PR7
1181      .=1000
1182      .SBTTL  APT PARAMETER BLOCK
1183
1184      ;;*****
1185      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1186      ;;*****
1187      .SX=.      ;;SAVE CURRENT LOCATION
1188      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1189 000024 000200  200      ;;FOR APT START UP
1190      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
1191 000044 001000  $APTHDR   ;;POINT TO APT HEADER BLOCK
1192      .=$X      ;;RESET LOCATION COUNTER
1193      ;;*****
1194      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1195      ;INTERFACE SPEC.
1196
1197 001000  $APTHD:
1198 001000  $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1199 001002 001214  $MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1200 001004 000000  $TSTM:  .WORD      ;;RUN TIM OF LONGEST TEST
1201 001006 000000  $PASTM: .WORD      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1202 001010 000000  $UNITM: .WORD      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1203 001012 000032  .WORD  SETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1204      MSP=   BIT6      ;MAINTENANCE SECTOR PULSE
1205      MIND=  BIT7      ;MAINTENANCE INDEX
1206      MCLK=  BIT8      ;MAINTENANCE CLOCK
1207      MERD=  BIT9      ;MAINTENANCE ENCODED READ DATA
1208      MEWD=  BIT10     ;MAINTENANCE ENCODED WRITE DATA
1209      PCA=   BIT11     ;PRECOMPENSATION ADVANCE
1210      PCD=   BIT12     ;PRECOMPENSATION DELAY
1211      ECCW=  BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1212      WRTGAT=BIT14    ;WRITE GATE

```



```

1213          100000          RDGATE= BIT15          ;READ GATE
1214
1215          .SBTTL TRANSMITTED MESSAGE A
1216
1217          000020          S. SEEK= BIT4          ;SEEK COMMAND
1218          000040          S. RECL= BITS          ;RECALIBRATE COMMAND
1219          000100          S. STSP= BIT6          ;START SPINDLE COMMAND
1220          000200          S. RTC= BIT7          ;DRIVE RETURN TO CENTERLINE COMMAND
1221          000400          S. CLR= BIT8          ;CLEAR ERROR AND DSC
1222          001000          S. FMT= BIT9          ;FORMAT
1223          002000          S. UNLD= BIT10         ;UNLOAD
1224          004000          S. PACK= BIT11        ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1225          .SBTTL TRAP CATCHER
1226
1227          000000          . = 0
1228          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1229          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1230          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1231          000174          . = 174
1232          000174          000000          DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
1233          000176          000000          SWREG: .WORD 0          ;;SOFTWARE SWITCH REGISTER
1234          .SBTTL STARTING ADDRESS(ES)
1235          000200          000137          003334          JMP 2*START.;;JMP TO STARTING ADDRESS OF PROGRAM
1236          000204          000137          003324          JMP RESTR          ;JUMP TO RESTART ROUTINE
1237          000214          . = 214
1238          000214          000137          003314          JMP PARM          ;JUMP TO OPERATOR ASSIGNED PARAMETERS
1239          .SBTTL ACT11 HOOKS
1240
1241          ;*****
1242          ;HOOKS REQUIRED BY ACT11
1243          000220          $SVPC=          ;SAVE PC
1244          000046          . = 46
1245          000046          034076          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1246          000052          000052          . = 52
1247          000052          000000          .WORD 0          ;;2)SET LOC.52 TO ZERO
1248          000220          . = $SVPC          ;; RESTORE PC
1249          000114          . = MEMVEC
1250          000114          034530          MEMERR          ;MEMPARITY ERROR HANDLER
1251          000116          000340          PR7
1252          001000          . = 1000
1253          .SBTTL APT PARAMETER BLOCK
1254
1255          ;*****
1256          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1257          ;*****
1258          001000          . $X=          ;SAVE CURRENT LOCATION
1259          000024          . = 24          ;SET POWER FAIL TO POINT TO START OF PROGRAM
1260          000024          000200          200          ;FOR APT START UP
1261          000044          . = 44          ;POINT TO APT INDIRECT ADDRESS PNTR.
1262          000044          001000          $APTHDR          ;POINT TO APT HEADER BLOCK
1263          001000          . = $X          ;RESET LOCATION COUNTER
1264          ;*****
1265          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1266          ;INTERFACE SPEC.
1267
1268          001000          $APTHD:

```

B03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA
DZR6DA.P11 28-SEP-76 15:50

MACY11 27(1006) 05-OCT-76 09:06 PAGE 27

SEQ 0027

APT PARAMETER BLOCK

1269	001000	000000	\$HIBTS:	.WORD	0	:: TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1270	001002	001214	\$MBADR:	.WORD	\$MAIL	:: ADDRESS OF APT MAILBOX (BITS 3-15)
1271	001004	000000	\$YSM:	.WORD		:: RUN TIM OF LONGEST TEST
1272	001006	000000	\$PASTM:	.WORD		:: RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1273	001010	000000	\$UNITM:	.WORD		:: ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1274	001012	000032		.WORD	SETEND-\$MAIL 2	:: LENGTH MAILBOX-ETABLE (WORDS)

1275
1276
1277
1278
1279
1280
1281 001100
1282 001100
1283 001100 000000
1284 001102 000
1285 001103 000
1286 001104 000000
1287 001106 000000
1288 001110 000000
1289 001112 000000
1290 001114 000
1291 001115 001
1292 001116 000000
1293 001120 000000
1294 001122 000000
1295 001124 000000
1296 001126 000000
1297 001130 000000
1298 001132 000000
1299 001134 000
1300 001135 000
1301 001136 000000
1302 001140 177570
1303 001142 177570
1304 001144 177560
1305 001146 177562
1306 001150 177564
1307 001152 177566
1308 001154 000
1309 001155 002
1310 001156 012
1311 001157 000
1312 001160 000000
1313 001162 000000
1314 001164 000000
1315 001166 000000
1316 001170 000000
1317 001172 000000
1318 001174 000000
1319 001176 000000
1320 001200 000000
1321 001202 000000
1322 001204 177607 000377
1323 001210 077
1324 001211 015
1325 001212 000012
1326
1327
1328
1329
1330

.SBTTL COMMON TAGS

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

SCMTAG: . =1100
:: START OF COMMON TAGS
\$STNM: .WORD 0
:: CONTAINS THE TEST NUMBER
\$ERFLG: .BYTE 00
:: CONTAINS ERROR FLAG
\$ICNT: .WORD 00
:: CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 00
:: CONTAINS SCOPE LOOP ADDRESS
\$LPERR: .WORD 00
:: CONTAINS SCOPE RETURN FOR ERRORS
\$ERTTL: .WORD 00
:: CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 0
:: CONTAINS ITEM CONTROL BYTE
\$ERMAX: .BYTE 1
:: CONTAINS MAX. ERRORS PER TEST
\$ERRPC: .WORD 0
:: CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 00
:: CONTAINS ADDRESS OF 'GOOD' DATA
\$BDADR: .WORD 00
:: CONTAINS ADDRESS OF 'BAD' DATA
\$GDAT: .WORD 00
:: CONTAINS 'GOOD' DATA
\$BDAT: .WORD 00
:: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED
\$AUTOB: .BYTE 0
:: AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0
:: INTERRUPT MODE INDICATOR
\$SWR: .WORD 0
:: ADDRESS OF SWITCH REGISTER
\$DISP: .WORD 0
:: ADDRESS OF DISPLAY REGISTER
\$TKS: 177560
:: TTY KBD STATUS
\$TKB: 177562
:: TTY KBD BUFFER
\$STPS: 177564
:: TTY PRINTER STATUS REG. ADDRESS
\$STPB: 177566
:: TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0
:: CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2
:: CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12
:: INSERT FILL CHARS. AFTER A "LINE FEED"
\$STPFLG: .BYTE 0
:: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$STMP0: .WORD 0
:: USER DEFINED
\$STMP1: .WORD 0
:: USER DEFINED
\$STMP2: .WORD 0
:: USER DEFINED
\$STMP3: .WORD 0
:: USER DEFINED
\$STMP4: .WORD 0
:: USER DEFINED
\$STMP5: .WORD 0
:: USER DEFINED
\$STMP6: .WORD 0
:: USER DEFINED
\$STMP7: .WORD 0
:: USER DEFINED
\$TIMES: 0
:: MAX. NUMBER OF ITERATIONS
\$ESCAPE: 0
:: ESCAPE ON ERROR ADDRESS
\$BELL: .ASCIZ (<207><377><377>)
:: CODE FOR BELL
\$QUES: .ASCII '?'
:: QUESTION MARK
\$CRLF: .ASCII (<15>)
:: CARRIAGE RETURN
\$LF: .ASCIZ (<12>)
:: LINE FEED

.SBTTL APT MAILBOX-ETABLE

::*****
::EVEN

1331	001214		\$MAIL:		:: APT MAILBOX
1332	001214	000000	\$MSGTY:	.WORD	AMSGTY :: MESSAGE TYPE CODE
1333	001216	000000	\$FATAL:	.WORD	AFATAL :: FATAL ERROR NUMBER
1334	001220	000000	\$TESTN:	.WORD	ATESTN :: TEST NUMBER
1335	001222	000000	\$PASS:	.WORD	APASS :: PASS COUNT
1336	001224	000000	\$DEVCT:	.WORD	ADEVCT :: DEVICE COUNT
1337	001226	000000	\$UNIT:	.WORD	AUNIT :: I/O UNIT NUMBER
1338	001230	000000	\$MSGAD:	.WORD	AMSGAD :: MESSAGE ADDRESS
1339	001232	000000	\$MSGLG:	.WORD	AMSGLG :: MESSAGE LENGTH
1340	001234		\$ETABLE:		APT ENVIRONMENT TABLE
1341	001234	000	\$ENV:	.BYTE	AENV :: ENVIRONMENT BYTE
1342	001236	000	\$ENVM:	.BYTE	AENVM :: ENVIRONMENT MODE BITS
1343	001236	000000	\$SWREG:	.WORD	ASWREG :: APT SWITCH REGISTER
1344	001240	000000	\$USWR:	.WORD	ALSWR :: USER SWITCHES
1345	001242	000000	\$CPJOP:	.WORD	ACPJOP :: CPU TYPE, OPTIONS
1346			*		BITS 15-11=CPU TYPE
1347			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1348			*		11/70=06, PDQ=07, Q=10
1349			*		BIT 10=REAL TIME CLOCK
1350			*		BIT 9=FLOATING POINT PROCESSOR
1351			*		BIT 8=MEMORY MANAGEMENT
1352	001244	000	\$MAMS1:	.BYTE	AMAMS1 :: HIGH ADDRESS, M.S. BYTE
1353	001245	000	\$MTYP1:	.BYTE	AMTYP1 :: MEM. TYPE, BLK#1
1354			*		MEM. TYPE BYTE -- (HIGH BYTE)
1355			*		900 NSEC CORE=001
1356			*		300 NSEC BIPOLAR=002
1357			*		500 NSEC MOS=003
1358	001246	000000	\$MADR1:	.WORD	AMADR1 :: HIGH ADDRESS, BLK#1
1359			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1360	001250	000	\$MAMS2:	.BYTE	AMAMS2 :: HIGH ADDRESS, M.S. BYTE
1361	001251	000	\$MTYP2:	.BYTE	AMTYP2 :: MEM. TYPE, BLK#2
1362	001252	000000	\$MADR2:	.WORD	AMADR2 :: MEM. LAST ADDRESS, BLK#2
1363	001254	000	\$MAMS3:	.BYTE	AMAMS3 :: HIGH ADDRESS, M.S. BYTE
1364	001255	000	\$MTYP3:	.BYTE	AMTYP3 :: MEM. TYPE, BLK#3
1365	001256	000000	\$MADR3:	.WORD	AMADR3 :: MEM. LAST ADDRESS, BLK#3
1366	001260	000	\$MAMS4:	.BYTE	AMAMS4 :: HIGH ADDRESS, M.S. BYTE
1367	001261	000	\$MTYP4:	.BYTE	AMTYP4 :: MEM. TYPE, BLK#4
1368	001262	000000	\$MADR4:	.WORD	AMADR4 :: MEM. LAST ADDRESS, BLK#4
1369	001264	000210	\$VECT1:	.WORD	AVECT1 :: INTERRUPT VECTOR#1, BUS PRIORITY#1
1370	001266	000000	\$VECT2:	.WORD	AVECT2 :: INTERRUPT VECTOR#2, BUS PRIORITY#2
1371	001270	177440	\$BASE:	.WORD	ABASE :: BASE ADDRESS OF EQUIPMENT UNDER TEST
1372	001272	000000	\$DEVN:	.WORD	ADEVN :: DEVICE MAP
1373	001274	000000	\$CDW1:	.WORD	ACDW1 :: CONTROLLER DESCRIPTION WORD#1
1374	001276	000000	\$CDW2:	.WORD	ACDW2 :: CONTROLLER DESCRIPTION WORD#2
1375	001300		\$ETEND:		
1376			.MEXIT		

E03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 30
DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEQ 0030

1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432

001300
001300 046113
001302 044120
001304 042460
001306 043014
001310 000000
001312 000000
001314 0424E4
001316 04302C
001320 046160
001322 052054
001324 042506
001326 043044
001330 046160
001332 052072
001334 042506
001336 043044
001340 046160
001342 052113
001344 042506
001346 043044
001350 046235
001352 052054
001354 042506
001356 043044
001360 046235
001362 052072
001364 042506
001366 043044

.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:

: ERROR 1: UNEXPECTED MEMORY PARTIY ENABLE TRAP
EM000
DH000C
DT000
DF000
: ERROR 2: WRITE BIT ERROR
EMW:
0
0
DT002
DF002
: ERROR 3: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
CSI INCORRECT
EM300
EM4000
DT003
DF003
: ERROR 4: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
MESS A INCORRECT
EM300
EM4001
DT003
DF003
: ERROR 5: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
MESS B INCORRECT
EM300
EM4002
DT003
DF003
: ERROR 6: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA
CSI INCORRECT
EM301
EM4000
DT003
DF003
: ERROR 7: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA
MESS A INCORRECT
EM301
EM4001
DT003
DF003
: ERROR 10: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA

F03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 31
 DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEG 0031

1433			:	MESS B INCORRECT
1434	001370	046235	:	EM301
1435	001372	052113	:	EM4002
1436	001374	042506	:	DT003
1437	001376	043044	:	DF003
1438			:	ERROR 11: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1439			:	CSI INCORRECT
1440	001400	046313	:	EM302
1441	001402	052054	:	EM4000
1442	001404	042506	:	DT003
1443	001406	043044	:	DF003
1444			:	ERROR 12: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1445			:	MESS A INCORRECT
1446	001410	046313	:	EM302
1447	001412	052072	:	EM4001
1448	001414	042506	:	DT003
1449	001416	043044	:	DF003
1450			:	ERROR 13: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1451			:	MESS B INCORRECT
1452	001420	046313	:	EM302
1453	001422	052113	:	EM4002
1454	001424	042506	:	DT003
1455	001426	043044	:	DF003
1456			:	ERROR 14: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1457			:	CSI INCORRECT
1458	001430	046372	:	EM303
1459	001432	052054	:	EM4000
1460	001434	042506	:	DT003
1461	001436	043044	:	DF003
1462			:	ERROR 15: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1463			:	MESS A INCORRECT
1464	001440	046372	:	EM303
1465	001442	052072	:	EM4001
1466	001444	042506	:	DT003
1467	001446	043044	:	DF003
1468			:	ERROR 16: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1469			:	MESS B INCORRECT
1470	001450	046372	:	EM303
1471	001452	052113	:	EM4002
1472	001454	042506	:	DT003
1473	001456	043044	:	DF003
1474			:	ERROR 17: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1475			:	CSI INCORRECT
1476	001460	046450	:	EM304
1477	001462	052054	:	EM4000
1478	001464	042506	:	DT003
1479	001466	043044	:	DF003
1480			:	ERROR 20: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1481			:	MESS A INCORRECT
1482	001470	046450	:	EM304
1483	001472	052072	:	EM4001
1484	001474	042506	:	DT003
1485	001476	043044	:	DF003
1486			:	ERROR 21: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1487			:	MESS B INCORRECT
1488	001500	046450	:	EM304

1489	001502	052113	EM4002
1490	001504	042506	DT003
1491	001506	043044	DF003
1492			ERROR 22: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1493			CSI INCORRECT
1494	001510	046527	EM305
1495	001512	052054	EM4000
1496	001514	042506	DT003
1497	001516	043044	DF003
1498			ERROR 23: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1499			MESS A INCORRECT
1500	001520	046527	EM305
1501	001522	052072	EM4001
1502	001524	042506	DT003
1503	001526	043044	DF003
1504			ERROR 24: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1505			MESS B INCORRECT
1506	001530	046527	EM305
1507	001532	052113	EM4002
1508	001534	042506	DT003
1509	001536	043044	DF003
1510			ERROR 25: ATTEMPTING TO CHECK HEADER GENERATION
1511			WITH VARIOUS CYLINDER VALUES
1512			CSI INCORRECT
1513	001540	046607	EM306
1514	001542	052054	EM4000
1515	001544	042506	DT003
1516	001546	043070	DF025
1517			ERROR 26: ATTEMPTING TO CHECK HEADER GENERATION
1518			WITH VARIOUS CYLINDER VALUES
1519			FIRST WORD OF HEADER INCORRECT
1520	001550	046607	EM306
1521	001552	052134	EM4003
1522	001554	042506	DT003
1523	001556	043070	DF025
1524			ERROR 27: ATTEMPTING TO CHECK HEADER GENERATION
1525			WITH VARIOUS CYLINDER VALUES
1526			SECOND WORD OF HEADER INCORRECT
1527	001560	046607	EM306
1528	001562	052164	EM4004
1529	001564	042506	DT003
1530	001566	043070	DF025
1531			ERROR 30: ATTEMPTING TO CHECK HEADER GENERATION
1532			WITH VARIOUS TRACK VALUES
1533			CSI INCORRECT
1534	001570	046711	EM307
1535	001572	052054	EM4000
1536	001574	042506	DT003
1537	001576	043070	DF025
1538			ERROR 31: ATTEMPTING TO CHECK HEADER GENERATION
1539			WITH VARIOUS TRACK VALUES
1540			FIRST WORD OF HEADER INCORRECT
1541	001600	046711	EM307
1542	001602	052134	EM4003
1543	001604	042506	DT003
1544	001606	043070	DF025

H03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 33
 DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEQ 0033

1545	:	ERROR 32: ATTEMPTING TO CHECK HEADER GENERATION
1546	:	WITH VARIOUS TRACK VALUES
1547	:	SECOND WORD OF HEADER INCORRECT
1548	001610 046711	EM307
1549	001612 052164	EM4004
1550	001614 042506	DT003
1551	001616 043070	DF025
1552	:	ERROR 33: ATTEMPTING TO CHECK HEADER GENERATION
1553	:	WITH VARIOUS SECTOR VALUES
1554	:	CSI INCORRECT
1555	001620 047010	EM308
1556	001622 052054	EM4000
1557	001624 042506	DT003
1558	001626 043070	DF025
1559	:	ERROR 34: ATTEMPTING TO CHECK HEADER GENERATION
1560	:	WITH VARIOUS SECTOR VALUES
1561	:	FIRST WORD OF HEADER INCORRECT
1562	001630 047010	EM308
1563	001632 052134	EM4003
1564	001634 042506	DT003
1565	001636 043070	DF025
1566	:	ERROR 35: ATTEMPTING TO CHECK HEADER GENERATION
1567	:	WITH VARIOUS SECTOR VALUES
1568	:	SECOND WORD OF HEADER INCORRECT
1569	001640 047010	EM308
1570	001642 052164	EM4004
1571	001644 042506	DT003
1572	001646 043070	DF025
1573	:	ERROR 36: ATTEMPTING TO CHECK HEADER GENERATION
1574	:	WITH VARIOUS FORMAT VALUES
1575	:	CSI INCORRECT
1576	001650 047110	EM309
1577	001652 052054	EM4000
1578	001654 042506	DT003
1579	001656 043070	DF025
1580	:	ERROR 37: ATTEMPTING TO CHECK HEADER GENERATION
1581	:	WITH VARIOUS FORMAT VALUES
1582	:	FIRST WORD OF HEADER INCORRECT
1583	001660 047110	EM309
1584	001662 052134	EM4003
1585	001664 042506	DT003
1586	001666 043070	DF025
1587	:	ERROR 40: ATTEMPTING TO CHECK HEADER GENERATION
1588	:	WITH VARIOUS FORMAT VALUES
1589	:	SECOND WORD OF HEADER INCORRECT
1590	001670 047110	EM309
1591	001672 052164	EM4004
1592	001674 042506	DT003
1593	001676 043070	DF025
1594	:	ERROR 41: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1595	:	CSI INCORRECT
1596	001700 047210	EM310
1597	001702 052054	EM4000
1598	001704 042526	DT041
1599	001706 043114	DF041
1600	:	ERROR 42: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA

1601			:	CS2 INCORRECT
1602	001710	047210	:	EM310
1603	001712	052214	:	EM4005
1604	001714	042526	:	DT041
1605	001716	043114	:	DF041
1606			:	ERROR 43: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1607			:	ERROR REGISTER INCORRECT
1608	001720	047210	:	EM310
1609	001722	052232	:	EM4006
1610	001724	042526	:	DT041
1611	001726	043114	:	DF041
1612			:	ERROR 44: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1613			:	BUS ADD INCORRECT
1614	001730	047210	:	EM310
1615	001732	052256	:	EM4007
1616	001734	042526	:	DT041
1617	001736	043114	:	DF041
1618			:	ERROR 45: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1619			:	WORD COUNT INCORRECT
1620	001740	047210	:	EM310
1621	001742	052304	:	EM4008
1622	001744	042526	:	DT041
1623	001746	043114	:	DF041
1624			:	ERROR 46: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1625			:	CS1 INCORRECT AFTER READING DATA BUFFER
1626	001750	047210	:	EM310
1627	001752	052331	:	EM4009
1628	001754	042556	:	DT046
1629	001756	043150	:	DF046
1630			:	ERROR 47: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1631			:	CS2 INCORRECT AFTER READING DATA BUFFER
1632	001760	047210	:	EM310
1633	001762	052401	:	EM4010
1634	001764	042556	:	DT046
1635	001766	043150	:	DF046
1636			:	ERROR 50: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1637			:	ERROR REG INCORRECT AFTER READING DATA BUFFER
1638	001770	047210	:	EM310
1639	001772	052451	:	EM4011
1640	001774	042556	:	DT046
1641	001776	043150	:	DF046
1642			:	ERROR 51: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1643			:	DATA READ FROM MEMORY INCORRECT
1644	002000	047210	:	EM310
1645	002002	052527	:	EM4012
1646	002004	042576	:	DT051
1647	002006	043174	:	DF051
1648			:	ERROR 52: ATTEMPTING TO CHECK HEADER RECOGNITION
1649			:	CS1 INCORRECT
1650	002010	047275	:	EM311
1651	002012	052054	:	EM4000
1652	002014	042610	:	DT052
1653	002016	043220	:	DF052
1654			:	ERROR 53: ATTEMPTING TO CHECK HEADER RECOGNITION
1655			:	MRI INCORRECT AFTER GAP IN WRITE DATA
1656	002020	047275	:	EM311

J03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 35
 DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEG 0035

1657	002022	052567	EM4013
1658	002024	042624	DT053
1659	002026	043244	DF053
1660	:	:	ERROR 54: ATTEMPTING TO CHECK SECTOR INCREMENT
1661	:	:	DISK ADDRESS REG INCORRECT
1662	002030	047344	EM312
1663	002032	052635	EM4014
1664	002034	042634	DT054
1665	002036	043270	DF054
1666	:	:	ERROR 55: ATTEMPTING TO CHECK SECTOR INCREMENT
1667	:	:	CYLINDER ADDRESS INCORRECT
1668	002040	047344	EM312
1669	002042	052671	EM4015
1670	002044	042634	DT054
1671	002046	043270	DF054
1672	:	:	ERROR 56: ATTEMPTING TO CHECK TRACK INCREMENT
1673	:	:	DISK ADDRESS REG INCORRECT
1674	002050	047411	EM313
1675	002052	052635	EM4014
1676	002054	042634	DT054
1677	002056	043270	DF054
1678	:	:	ERROR 57: ATTEMPTING TO CHECK TRACK INCREMENT
1679	:	:	CYLINDER ADDRESS INCORRECT
1680	002060	047411	EM313
1681	002062	052671	EM4015
1682	002064	042634	DT054
1683	002066	043270	DF054
1684	:	:	ERROR 60: ATTEMPTING TO CHECK CYLINDER INCREMENT
1685	:	:	DISK ADDRESS INCORRECT
1686	002070	047455	EM314
1687	002072	052635	EM4014
1688	002074	042634	DT054
1689	002076	043270	DF054
1690	:	:	ERROR 61: ATTEMPTING TO CHECK CYLINDER INCREMENT
1691	:	:	CYLINDER ADDRESS INCORRECT
1692	002100	047455	EM314
1693	002102	052671	EM4015
1694	002104	042634	DT054
1695	002106	043270	DF054
1696	:	:	ERROR 62: ATTEMPTING TO CHECK SECTOR PULSE DETECTION WITH
1697	:	:	WRITE DATA
1698	:	:	MAINT. REG. 1 INCORRECT
1699	002110	047524	EM315
1700	002112	052731	EM4016
1701	002114	042624	DT053
1702	002116	043244	DF053
1703	:	:	ERROR 63: ATTEMPTING TO FORCE BAD SECTOR ERROR
1704	:	:	CS1 INCORRECT
1705	002120	047620	EM316
1706	002122	052054	EM4000
1707	002124	042556	DT046
1708	002126	043150	DF046
1709	:	:	ERROR 64: ATTEMPTING TO FORCE BAD SECTOR ERROR
1710	:	:	CS2 INCORRECT
1711	002130	047620	EM316
1712	002132	052214	EM4005

K03

1713	002134	042556	DT046
1714	002136	043150	DF046
1715	:	:	ERROR 65: ATTEMPTING TO FORCE BAD SECTOR ERROR
1716	:	:	ERROR REG INCORRECT
1717	002140	047620	EM316
1718	002142	052232	EM4006
1719	002144	042556	DT046
1720	002146	043150	DF046
1721	:	:	ERROR 66: ATTEMPTING TO FORCE HEADER VRC ERROR
1722	:	:	WHEN BAD SECTOR PRESENT
1723	:	:	CS1 INCORRECT
1724	002150	047665	EM317
1725	002152	052054	EM4000
1726	002154	042556	DT046
1727	002156	043150	DF046
1728	:	:	ERROR 67: ATTEMPTING TO FORCE HEADER VRC ERROR
1729	:	:	WHEN BAD SECTOR PRESENT
1730	:	:	CS2 INCORRECT
1731	002160	047665	EM317
1732	002162	052214	EM4005
1733	002164	042556	DT046
1734	002166	043150	DF046
1735	:	:	ERROR 70: ATTEMPTING TO FORCE HEADER VRC ERROR
1736	:	:	WHEN BAD SECTOR PRESENT
1737	:	:	ERROR REG INCORRECT
1738	002170	047665	EM317
1739	002172	052232	EM4006
1740	002174	042556	DT046
1741	002176	043150	DF046
1742	:	:	ERROR 71: ATTEMPTING TO FORCE HVRC ERROR
1743	:	:	CS1 INCORRECT
1744	002200	047763	EM318
1745	002202	052054	EM4000
1746	002204	042650	DT071
1747	002206	043314	DF071
1748	:	:	ERROR 72: ATTEMPTING TO FORCE HVRC ERROR
1749	:	:	CS2 INCORRECT
1750	002210	047763	EM318
1751	002212	052214	EM4005
1752	002214	042650	DT071
1753	002216	043314	DF071
1754	:	:	ERROR 73: ATTEMPTING TO FORCE HVRC ERROR
1755	:	:	ERROR REG INCORRECT
1756	002220	047763	EM319
1757	002222	052232	EM4006
1758	002224	042650	DT071
1759	002226	043314	DF071
1760	:	:	ERROR 74: ATTEMPTING TO FORCE OPERATION INCCMPLETE
1761	:	:	CS1 INCORRECT
1762	002230	050022	EM319
1763	002232	052054	EM4000
1764	002234	042676	DT074
1765	002236	043350	DF074
1766	:	:	ERROR 75: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1767	:	:	CS2 INCORRECT
1768	002240	050022	EM319

1769	002242	052214	EM4005
1770	002244	042676	DT074
1771	002246	043350	DF074
1772	:	:	ERROR 76: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1773	:	:	ERROR REG INCORRECT
1774	002250	050022	EM319
1775	002252	052232	EM4006
1776	002254	042676	DT074
1777	002256	043350	DF074
1778	:	:	ERROR 77: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1779	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1780	002260	050022	EM319
1781	002262	052567	EM4013
1782	002264	042720	DT077
1783	002266	043374	DF077
1784	:	:	ERROR 100: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1785	:	:	CS1 INCORRECT
1786	002270	050073	EM320
1787	002272	052054	EM4000
1788	002274	042676	DT074
1789	002276	043350	DF074
1790	:	:	ERROR 101: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1791	:	:	CS2 INCORRECT
1792	002300	050073	EM320
1793	002302	052214	EM4005
1794	002304	042676	DT074
1795	002306	043350	DF074
1796	:	:	ERROR 102: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1797	:	:	ERROR REG INCORRECT
1798	002310	050073	EM320
1799	002312	052232	EM4006
1800	002314	042676	DT074
1801	002316	043350	DF074
1802	:	:	ERROR 103: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1803	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1804	002320	050073	EM320
1805	002322	052567	EM4013
1806	002324	042720	DT077
1807	002326	043374	DF077
1808	:	:	ERROR 104: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1809	:	:	CS1 INCORRECT
1810	002330	050157	EM321
1811	002332	052054	EM4000
1812	002334	042676	DT074
1813	002336	043350	DF074
1814	:	:	ERROR 105: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1815	:	:	CS2 INCORRECT
1816	002340	050157	EM321
1817	002342	052214	EM4005
1818	002344	042676	DT074
1819	002346	043350	DF074
1820	:	:	ERROR 106: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1821	:	:	ERROR REG INCORRECT
1822	002350	050157	EM321
1823	002352	052232	EM4006
1824	002354	042676	DT074

M03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 38
 DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEQ 0038

1825	002356	043350	DF074
1826			ERROR 107: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1827			MR1 INCORRECT AFTER GAP IN WRITE DATA
1828	002360	050157	EM321
1829	002362	052567	EM4013
1830	002364	042720	DT077
1831	002366	043374	DF077
1832			ERROR 110: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1833			CS1 INCORRECT
1834	002370	050243	EM322
1835	002372	052054	EM4000
1836	002374	042676	DT074
1837	002376	043350	DF074
1838			ERROR 111: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1839			CS2 INCORRECT
1840	002400	050243	EM322
1841	002402	052214	EM4005
1842	002404	042676	DT074
1843	002406	043350	DF074
1844			ERROR 112: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1845			ERROR REG INCORRECT
1846	002410	050243	EM322
1847	002412	052232	EM4006
1848	002414	042676	DT074
1849	002416	043350	DF074
1850			ERROR 113: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1851			MR1 INCORRECT AFTER GAP IN WRITE DATA
1852	002420	050243	EM322
1853	002422	052567	EM4013
1854	002424	042720	DT077
1855	002426	043374	DF077
1856			ERROR 114: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1857			SECTOR ERROR CS1 INCORRECT
1858	002430	050326	EM323
1859	002432	052054	EM4000
1860	002434	042676	DT074
1861	002436	043350	DF074
1862			ERROR 115: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1863			SECTOR ERROR CS2 INCORRECT
1864	002440	050326	EM323
1865	002442	052214	EM4005
1866	002444	042676	DT074
1867	002446	043350	DF074
1868			ERROR 116: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1869			SECTOR ERROR ERROR INCORRECT
1870	002450	050326	EM323
1871	002452	052232	EM4006
1872	002454	042676	DT074
1873	002456	043350	DF074
1874			ERROR 117: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1875			SECTOR ERROR MR1 INCORRECT
1876	002460	050326	EM323
1877	002462	052567	EM4013
1878	002464	042720	DT077
1879	002466	043374	DF077
1880			ERROR 120: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER

1881			:		VRC ERROR CS1 INCORRECT
1882	002470	050421	:	EM324	
1883	002472	052054	:	EM4000	
1884	002474	042676	:	DT074	
1885	002476	043350	:	DF074	
1886			:	ERROR 121:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1887			:		VRC ERROR CS2 INCORRECT
1888	002500	050421	:	EM324	
1889	002502	052214	:	EM4005	
1890	002504	042676	:	DT074	
1891	002506	043350	:	DF074	
1892			:	ERROR 122:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1893			:		VRC ERROR ERROR REG INCORRECT
1894	002510	050421	:	EM324	
1895	002512	052232	:	EM4006	
1896	002514	042676	:	DT074	
1897	002516	043350	:	DF074	
1898			:	ERROR 123:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1899			:		VRC ERROR MRI INCORRECT
1900	002520	050421	:	EM324	
1901	002522	052567	:	EM4013	
1902	002524	042720	:	DT077	
1903	002526	043374	:	DF077	
1904			:	ERROR 124:	FORCING BAD SECTOR ERROR WITH PREV HEADER
1905			:		VRC ERROR CS1 INCORRECT
1906	002530	050514	:	EM325	
1907	002532	052054	:	EM4000	
1908	002534	042676	:	DT074	
1909	002536	043350	:	DF074	
1910			:	ERROR 125:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1911			:		CS2 INCORRECT
1912	002540	050514	:	EM325	
1913	002542	052214	:	EM4005	
1914	002544	042676	:	DT074	
1915	002546	043350	:	DF074	
1916			:	ERROR 126:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1917			:		ERROR REG INCORRECT
1918	002550	050514	:	EM325	
1919	002552	052232	:	EM4006	
1920	002554	042676	:	DT074	
1921	002556	043350	:	DF074	
1922			:	ERROR 127:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1923			:		MRI INCORRECT
1924	002560	050514	:	EM325	
1925	002562	052567	:	EM4013	
1926	002564	042720	:	DT077	
1927	002566	043374	:	DF077	
1928			:	ERROR 130:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1929			:		CS1 INCORRECT
1930	002570	050604	:	EM326	
1931	002572	052054	:	EM4000	
1932	002574	042676	:	DT074	
1933	002576	043350	:	DF074	
1934			:	ERROR 131:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1935			:		CS2 INCORRECT
1936	002600	050604	:	EM326	

1937	002602	052214	EM4005
1938	002604	042676	DT074
1939	002606	043350	DF074
1940		:	ERROR 132: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1941		:	ERROR REG INCORRECT
1942	002610	050604	EM326
1943	002612	052232	EM4006
1944	002614	042676	DT074
1945	002616	043350	DF074
1946		:	ERROR 133: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1947		:	MRI INCORRECT
1948	002620	050604	EM326
1949	002622	052556	EM4013
1950	002624	042720	DT077
1951	002626	043374	DF077
1952		:	ERROR 134: ATTEMPTING TO CHECK ECC PATTERN CLEARING
1953		:	ECC PATTERN INCORRECT
1954	002630	051023	EM329
1955	002632	052747	EM4017
1956	002634	042732	DT134
1957	002636	043420	DF134
1958		:	ERROR 135: ATTEMPTING TO CHECK ECC GENERATION
1959		:	ECC PATTERN INCORRECT
1960	002640	051074	EM330
1961	002642	052747	EM4017
1962	002644	042732	DT134
1963	002646	043420	DF134
1964		:	ERROR 136: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1965		:	CS1 INCORRECT
1966	002650	051137	EM331
1967	002652	052054	EM4000
1968	002654	042556	DT046
1969	002656	043150	DF046
1970		:	ERROR 137: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1971		:	CS2 INCORRECT
1972	002660	051137	EM331
1973	002662	052214	EM4005
1974	002664	042556	DT046
1975	002666	043150	DF046
1976		:	ERROR 140: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1977		:	ERROR REG. INCORRECT
1978	002670	051137	EM331
1979	002672	052232	EM4006
1980	002674	042556	DT046
1981	002676	043150	DF046
1982		:	ERROR 141: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1983		:	CS1 INCORRECT
1984	002700	051224	EM332
1985	002702	052054	EM4000
1986	002704	042556	DT046
1987	002706	043150	DF046
1988		:	ERROR 142: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1989		:	CS2 INCORRECT
1990	002710	051224	EM332
1991	002712	052214	EM4005
1992	002714	042556	DT046

C04

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 41
 DZR6DA.P11 28-SEP-76 15:50 ERROR POINTER TABLE

SEG 0041

1993	002716	043150		DF046
1994			:	ERROR 143: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1995			:	ERROR REG. INCORRECT
1996	002720	051224		EM332
1997	002722	052232		EM4006
1998	002724	042556		DT046
1999	002726	043150		DF046
2000			:	ERROR 144: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2001			:	CS1 INCORRECT
2002	002730	051405		EM335
2003	002732	052054		EM4000
2004	002734	042744		DT144
2005	002736	043444		DF144
2006			:	ERROR 145: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2007			:	CS2 INCORRECT
2008	002740	051405		EM335
2009	002742	052214		EM4005
2010	002744	042744		DT144
2011	002746	043444		DF144
2012			:	ERROR 146: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2013			:	ERROR REG. INCORRECT
2014	002750	051405		EM335
2015	002752	052232		EM4006
2016	002754	042744		DT144
2017	002756	043444		DF144
2018			:	ERROR 147: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2019			:	BUS ADDRESS INCORRECT
2020	002760	051405		EM335
2021	002762	052256		EM4007
2022	002764	042744		DT144
2023	002766	043444		DF144
2024			:	ERROR 150: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2025			:	WORD COUNT INCORRECT
2026	002770	051405		EM335
2027	002772	052304		EM4008
2028	002774	042744		DT144
2029	002776	043444		DF144
2030			:	ERROR 151: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2031			:	CYLINDER ADD INCORRECT
2032	003000	051405		EM335
2033	003002	052671		EM4015
2034	003004	042744		DT144
2035	003006	043444		DF144
2036			:	ERROR 152: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2037			:	DISK ADDRESS INCORRECT
2038	003010	051405		EM335
2039	003012	052635		EM4014
2040	003014	042744		DT144
2041	003016	043444		DF144
2042			:	ERROR 153: ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR
2043			:	CS1 INCORRECT
2044	003020	051461		EM336
2045	003022	052054		EM4000
2046	003024	042610		DT052
2047	003026	043220		DF052
2048			:	ERROR 154: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL

2049			:	CS1 INCORRECT
2050	003030	051536	:	EM337
2051	003032	052234	:	EM4000
2052	003034	042744	:	DT144
2053	003036	043444	:	DF144
2054			:	ERROR 155: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2055			:	CS2 INCORRECT
2056	003040	051536	:	EM337
2057	003042	052214	:	EM4005
2058	003044	042744	:	DT144
2059	003046	043444	:	DF144
2060			:	ERROR 156: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2061			:	ERROR REGISTER INCORRECT
2062	003050	051536	:	EM337
2063	003052	052232	:	EM4006
2064	003054	042744	:	DT144
2065	003056	043444	:	DF144
2066			:	ERROR 157: ATTEMPTIN PARTIAL SECTOR WRITE WITH ZERO FILL
2067			:	BUS ADDRESS INCORRECT
2068	003060	051536	:	EM337
2069	003062	052256	:	EM4007
2070	003064	042744	:	DT144
2071	003066	043444	:	DF144
2072			:	ERROR 160: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2073			:	WORD COUNT INCORRECT
2074	003070	051536	:	EM337
2075	003072	052304	:	EM4008
2076	003074	042744	:	DT144
2077	003076	043444	:	DF144
2078			:	ERROR 161: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2079			:	CYLINDER ADDRESS INCORRECT
2080	003100	051536	:	EM337
2081	003102	052671	:	EM4015
2082	003104	042744	:	DT144
2083	003106	043444	:	DF144
2084			:	ERROR 162: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2085			:	DISK ADDRESS INCORRECT
2086	003110	051536	:	EM337
2087	003112	052635	:	EM4014
2088	003114	042744	:	DT144
2089	003116	043444	:	DF144
2090			:	ERROR 163: ATTEMPTING WRITE DATA IN 18 BIT MODE
2091			:	RKECPT INCORRECT
2092	003120	052007	:	EM340
2093	003122	052747	:	EM4017
2094	003124	043004	:	DT151
2095	003126	043500	:	DF151
2096			:	ERROR 164: ATTEMPTING WRITE DATA IN 18 BIT MODE
2097			:	RKECPT INCORRECT
2098	003130	052007	:	EM340
2099	003132	052747	:	EM4017
2100	003134	042732	:	DT134
2101	003136	043420	:	DF134
2102			:	

E04

```

2103 .SBTTL TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2104
2105 003140 000000 T.CS1: .WORD 0 :CONTROL AND STATUS REGISTER 1
2106 003142 000000 T.WC: .WORD 00 :WORD COUNT REGISTER
2107 003144 000000 T.BA: .WORD 00 :BUS ADDRESS REGISTER
2108 003146 000000 T.DA: .WORD 00 :DESIRED TRACK SECTOR REGISTER
2109 003150 000000 T.CS2: .WORD 00 :CONTROL AND STATUS REGISTER 2
2110 003152 000000 T.DS: .WORD 00 :DRIVE STATUS REGISTER
2111 003154 000000 T.ER: .WORD 00 :ERROR REGISTER
2112 003156 000000 T.ASOF: .WORD 00 :ATTENTION SUMMARY AND OFFSET REGISTER
2113 003160 000000 T.DCYL: .WORD 00 :DESIRED CYLINDER REGISTER
2114 003162 000000 T.DB: .WORD 00 :DATA BUFFER
2115 003164 000000 T.MR1: .WORD 00 :MAINTENANCE REGISTER 1
2116 003166 000000 T.MR2: .WORD 00 :MAINTENANCE REGISTER 2
2117 003170 000000 T.MR3: .WORD 00 :MAINTENANCE REGISTER 3
2118 003172 000000 T.ECPS: .WORD 00 :ECC POSITION INFORMATION
2119 003174 000000 T.ECPT: .WORD 00 :ECC PATTERN INFORMATION
2120 003176 000000 T.SPARE: .WORD 0 :SPARE REGISTER

```

```

2121 .SBTTL EXPECTED RK611 CONTROLLER REGISTERS
2122
2123 003200 000000 E.CS1: .WORD 0 :CONTROL AND STATUS REGISTER 1
2124 003202 000000 E.WC: .WORD 00 :WORD COUNT REGISTER
2125 003204 000000 E.BA: .WORD 00 :BUS ADDRESS REGISTER
2126 003206 000000 E.DA: .WORD 00 :DESIRED TRACK SECTOR REGISTER
2127 003210 000000 E.CS2: .WORD 00 :CONTROL AND STATUS REGISTER 2
2128 003212 000000 E.DS: .WORD 00 :DRIVE STATUS REGISTER
2129 003214 000000 E.ER: .WORD 00 :ERROR REGISTER
2130 003216 000000 E.ASOF: .WORD 00 :ATTENTION SUMMARY AND OFFSET REGISTER
2131 003220 000000 E.DCYL: .WORD 00 :DESIRED CYLINDER REGISTER
2132 003222 000000 E.DB: .WORD 00 :DATA BUFFER
2133 003224 000000 E.MR1: .WORD 00 :MAINTENANCE REGISTER 1
2134 003226 000000 E.MR2: .WORD 00 :MAINTENANCE REGISTER 2
2135 003230 000000 E.MR3: .WORD 00 :MAINTENANCE REGISTER 3
2136 003232 000000 E.ECPS: .WORD 00 :ECC POSITION INFORMATION
2137 003234 000000 E.ECPT: .WORD 00 :ECC PATTERN INFORMATION
2138 003236 000000 E.SPARE: .WORD 0 :SPARE REGISTER

```

F04

2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165

```

003240 000214
003242 000243
003244 000000
003246 000000

003250 000000
003252 000000
003254 000000
003256 000000
003260 000000
003262 000000
003264 000000
003266 000000
003270 000000
003272 000000
003274 000000
003276 000
003277 000
003300 000000
003302 000000 000000 000000
003310 000000
003312 000000
  
```

.SBTTL PROGRAM DEFINED VARIABLES

```

RVVEC: .WORD 214
RKPRI: .WORD PR5
TRAPPC: .WORD 0
SRFLG: .WORD 0

ERRCNT: .WORD 0
P1.BIT: .WORD 00
PR.BIT: .WORD 00
M1.BIT: .WORD 00
M2.BIT: .WORD 00
BITCNT: .WORD 00
WRDCNT: .WORD 00
ECCHI: .WORD 00
ECCLO: .WORD 00
ECCXOR: .WORD 00
MEMPAR: .WORD 00
SECTOR: .BYTE 00
TRACK: .BYTE 00
CYLN: .WORD 00
HEADER: .WORD 0,0,0
HDRCNT: .WORD 0
SAVSWR: .WORD 0
  
```

```

:RK611 VECTOR ADDRESS
:RK611 PRIORITY
:TRAP PC FOR MEMORY PARITY OPTION
:START FLAG
: 0 = 200
: 1 = 214
:-1 = 204
:ERROR COUNT FOR ABORT ERROR THRESHOLD
:NEXT BIT
:PRESENT BIT
:BIT-1
:BIT-2
:BIT COUNT
:WORD COUNT FOR NPR TRANSFERS
:16 MOST SIGNIFICANT BIT OF ECC
:16 LEAST SIGNIFICANT BIT OF ECC
:FLAG FOR ECC GENERATION
:FLAG FOR MEMORY PARITY OPTION
:SECTOR FOR INCREMENT TEST
:TRACK FOR INCREMENT TEST
:CYLINDER FOR INCREMENT TEST
:EXPECTED FOR INCREMENT TEST
:NUMBER OF HEADERS FOR OPI'S
:STORAGE FOR SWITCH REGISTER
  
```

```

2166 .SBTTL PROGRAM SETUP
2167
2168 003314 012737 003001 003246 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2169 003322 000406 BR START1
2170
2171 003324 012737 177777 003246 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2172 003332 000402 BR START1
2173
2174 003334 005037 003246 START: CLR SRTFLG ;CLEAR START FLAG
2175 003340 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2176 003342 105037 001103 CLR#B #SERFLG ;CLEAR ERROR FLAG
2177 003346 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2178 003352 004737 040552 JSR PC,STKINT ;INIT KEYBOARD
2179 003356 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2180 003362 012746 003370 MOV #IS,-(SP) ;LOAD START OF PROGRAM
2181 003366 000002 RTI ;LOAD PSW
2182
2183 003370
2184
2185 003370 012706 001100 IS:
2186 003374 005026 .SBTTL INITIALIZE THE COMMON TAGS
2187 003376 022706 ;;CLEAR THE COMMON TAGS (%CMTAG) AREA
2188 003402 001374 MOV #%CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2189 003404 012706 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221

```

```

MOV #SERFLG ;CLEAR ERROR FLAG
INITIALIZE STACK POINTER
INIT KEYBOARD
LOAD STACK TO LOCK OUT ALL INTERRUPTS
LOAD START OF PROGRAM
LOAD PSW

IS:
.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS (%CMTAG) AREA
MOV #%CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;;CLEAR MEMORY LOCATION
CMP #SWR,R6 ;;DONE?
BNE -6 ;;LOOP BACK IF NO
MOV #STACK,SP ;;SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV #SCOPE,%IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
MOV #340,%IOTVEC+2 ;;LEVEL 7
MOV #ERROR,%EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
MOV #340,%EMTVEC+2 ;;LEVEL 7
MOV #TRAP,%TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV #340,%TRAPVEC+2 ;;LEVEL 7
MOV #SPWRDN,%PWRVEC ;;POWER FAILURE VECTOR
MOV #340,%PWRVEC+2 ;;LEVEL 7
MOV #ENDCT,%EOPCT ;;SETUP END-OF-PROGRAM COUNTER
CLR #TIMES ;;INITIALIZE NUMBER OF ITERATIONS
CLR #ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
MOV #1,%SERMAX ;;ALLOW ONE ERROR PER TEST
MOV #,%SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,%SLPERR ;;SETUP THE ERROR LOOP ADDRESS
;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
MOV #64,%ERRVEC ;;SET UP ERROR VECTOR
MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
CMP #-1,%SWR ;;TRY TO REFERENCE HARDWARE SWR
BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
;;AND THE HARDWARE SWR IS NOT = -1
BR 65$ ;;BRANCH IF NO TIMEOUT
64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
RTI
65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
MOV #DISPRG,DISPLAY
66$: MOV (SP)+,%ERRVEC ;;RESTORE ERROR VECTOR

```



```

2222 003616 005037 001222 CLR $PASS ;:CLEAR PASS COUNT
2223 003622 132737 000200 001235 BITB #APTSIZE,$ENVM ;:TEST USER SIZE UNDER APT
2224 003630 001403 BEQ 67$ ;:YES,USE NON-APT SWITCH
2225 003632 012737 001236 001140 MOV #SSWREG,$SWR ;:NO,USE APT SWITCH REGISTER
2226 003640 67$:
2227 003640 005037 003250 CLR ERRCNT
2228 .SBTTL TYPE PROGRAM NAME
2229 ;:TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2230 003644 005227 177777 INC #-1 ;:FIRST TIME?
2231 003650 001063 BNE 68$ ;:BRANCH IF NO
2232 003652 022737 034076 000042 CMP #SENDAD,$#42 ;:ACT-11?
2233 003660 001457 BEQ 68$ ;:BRANCH IF YES
2234 003662 104401 003730 TYPE 69$ ;:TYPE ASCIZ STRING
2235 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2236 003666 005737 000042 TST $#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
2237 003672 001012 BNE 70$ ;:BRANCH IF YES
2238 003674 123727 001234 000001 CMPB $ENV,$#1 ;:ARE WE RUNNING UNDER APT?
2239 003702 001406 BEQ 70$ ;:BRANCH IF YES
2240 003704 023727 001140 000176 CMP $SWR,$#SWREG ;:SOFTWARE SWITCH REG SELECTED?
2241 003712 001005 BNE 71$ ;:BRANCH IF NO
2242 003714 104406 GTSWR ;:GET SOFT-SWR SETTINGS
2243 003716 000403 BR 71$
2244 003720 112737 000001 001134 70$: MOVB $#1,$AUTOB ;:SET AUTO-MODE INDICATOR
2245 003726 71$:
2246 003726 000434 BR 68$ ;:GET OVER THE ASCIZ
2247 ;:69$: .ASCIZ <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 4 MAINDEC-11-DZR6D-A/<CRLF>
2248 68$:
2249 004020 005227 177777 INC #-1 ;:TYPE RUN TIME MESSAGE FIRST PASS ONLY
2250 004024 001002 BNE 2$
2251 004026 104401 043673 TYPE ,OPRO06
2252 004032 022737 000001 003246 2$: CMP $#1,$RTFLG ;:CHECK IF PARAMETER START
2253 004040 001117 BNE 15$ ;:NO,CONTINUE SETUP
2254 004042 104401 043524 5$: TYPE ,OPRO01 ;:TYPE "RK611 BUS ADDRESS ( ) ="
2255 004046 013746 001270 MOV $BASE,-(SP) ;:SAVE $BASE FOR TYPEOUT
2256 004052 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2257 004054 104401 043553 TYPE ,OPRO02
2258 004060 104412 RDOCT ;:GET VALUE
2259 004062 012637 001160 MOV (SP)+,$TMP0
2260 004066 001407 BEQ 7$ ;:CHECK IF <CR>
2261 004070 022737 160000 001160 CMP #160000,$TMP0 ;:CHECK IF IN I/O PAGE
2262 004076 101361 BHI 5$
2263 004100 013737 001160 001270 7$: MOV $TMP0,$BASE ;:LOAD NEW BUS ADDRESS
2264 004106 104401 043561 TYPE ,OPRO03 ;:TYPE "RK611 VECTOR ADDRESS ( ) ="
2265 004112 013746 001264 MOV $VECT1,-(SP) ;:GET VECTOR ADDRESS FOR TYPE OUT
2266 004116 042716 160000 BIC #160000,(SP) ;:CLEAR PRIORITY BITS
2267 004122 104402 TYPOC
2268 004124 104401 043553 TYPE ,OPRO02
2269 004130 104412 RDOCT ;:GET VALUE
2270 004132 012637 001160 MOV (SP)+,$TMP0
2271 004136 001407 BEQ 10$ ;:CHECK IF <CR>
2272 004140 022737 001000 001160 CMP #1000,$TMP0 ;:CHECK IF LEGAL
2273 004146 101757 BLOS 7$
2274 004150 013737 001160 001264 10$: MOV $TMP0,$VECT1 ;:LOAD NEW VECTOR ADDRESS
2275 004156 104401 043611 TYPE ,OPRO04 ;:TYPE "RK611 PRIORITY ( ) ="
2276 004162 005046 CLR -(SP) ;:MAKE ROOM ON THE STACK
2277 004164 113716 001265 MOVB $VECT1+1,(SP) ;:GET PRIORITY

```

2278	004170	006216		ASR	(SP)		;SHIF RIGH 5 BITS
2279	004172	006216		ASR	(SP)		
2280	004174	006216		ASR	(SP)		
2281	004176	006216		ASR	(SP)		
2282	004200	006216		ASR	(SP)		
2283	004202	104402		TYPOC			
2284	004204	104401	043553	TYPE	,OPR002		
2285	004210	104412		RDOCT			;GET VALUE
2286	004212	012637	001160	MOV	(SP)+,\$TMPO		
2287	004216	001430		BEG	15\$;CHECK FOR DEFAULT
2288	004220	022737	000007	001160	CMP	#7,\$TMPO	;CHECK IF LEGAL
2289	004226	103753		BLO	10\$		
2290	004230	022737	000004	001160	CMP	#4,\$TMPO	
2291	004236	101347		BHI	10\$		
2292	004240	006337	001160	ASL	\$TMPO		;SHIFT 5 BITS LEFT
2293	004244	006337	001160	ASL	\$TMPO		
2294	004250	006337	001160	ASL	\$TMPO		
2295	004254	006337	001160	ASL	\$TMPO		
2296	004260	006337	001160	ASL	\$TMPO		
2297	004264	042737	160000	001264	BIC	#160000,\$VECT1	;GET PRIORITY BITS
2298	004272	153737	001160	001265	BISB	\$TMPO,\$VECT1+1	
2299	004300	013737	001264	003240	MOV	\$VECT1,RKVEC	
2300	004306	042737	160000	003240	BIC	#160000,RKVEC	;GET VECTOR
2301	004314	113737	001265	003242	MOVB	\$VECT1+1,RKPRI	;GET PRIORITY
2302	004322	013702	001270	MOV	\$BASE,R2		;SET '611 BASE
2303	004326	005037	001202	CLR	\$ESCAPE		;CLEAR ESCAPE
2304	004332	004737	024402	NEWPAS: JSR	PC,PARCHK		;ENABLE MEMORY PARITY
2305	004336	012746	000000	MOV	#PRO,-(SP)		;LOCK OUT INTERRUPTS
2306	004342	012746	004350	MOV	#TST1,-(SP)		
2307	004346	000002		RTI			

J04

.SBTTL **TYPE C INSTRUCTION'S DRIVE MESSAGES

*TEST 1 READ DATA SEEK MESSAGE

*
* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
* RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
* 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND
* MAKE SURE IT IS GENERATED PROPERLY.
*

```

TST1: SCOPE
MOV #50, $TIMES ;; DO 50. ITERATIONS
MOV $BASE, R2 ;; LOAD RK611 BASE
MOV ! $CCLR, RKCS1(R2) ; CLEAR RK611
MOV $DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV, #-1, RKWC(R2) ; LOAD WORD COUNT
MOV $BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV #1777, RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ; LOAD TRACK
MOV #7, RKCS2(R2) ; LOAD DRIVE NUM.
MOV $CDT!CFMT!RDATA, RKCS1(R2) ; ISSUE RDATA WITH CDT SET IN
; 24 SECTOR FORMAT
; CLOCK IN DRIVE MESSAGE
1$: MOV $DMD!MCLK, RKMR1(R2)
MOV $DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2), T.CS1 ; STORE COMMAND STATUS REG. 1
MOV RKMR2(R2), T.MR2 ; STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2), T.MR3 ; STORE MAINT REG. 3 (MESS B)
MOV $CDT!CFMT!RDATA, E.CS1 ; LOAD EXPECTED CS1
MOV $S.SEEK!S.FMT!70007, E.MR2 ; LOAD EXPECTED MR2
MOV $37760, E.MR3 ; LOAD EXPECTED MR3
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2$ ; YES, CHECK MESSAGE A
ERROR 3 ; CS1 INCORRECT
MOV $CCLR, RKCS1(R2) ; CLEAR RK611
BR TST2 ;; GO TO NEXT TEST

```

```

2$: CMP E.MR2, T.MR2 ; CHECK MESS A CORRECT
BEQ 3$ ; YES, CHECK MESSAGE B
ERROR 4 ; MESS A INCORRECT
3$: CMP E.MR3, T.MR3 ; CHECK MESS B CORRECT
BEQ TST2 ; YES, GO ON TO NEXT TEST
ERROR 5 ; MESS B INCORRECT

```

*TEST 2 WRITE DATA SEEK MESSAGE

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777,
* TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE. CHECK IF
* PROPER MESSAGE IS ASSEMBLED.
*

2308					
2309					
2310					
2311					
2312					
2313					
2314					
2315					
2316					
2317					
2318					
2319					
2320	004350	000004			
2321	004352	012737	000062	001200	
2322	004360	013702	001270		
2323	004364	012762	100000	000000	
2324	004372	012762	000040	000026	
2325	004400	012762	177777	000002	
2326	004406	012762	055220	000004	
2327	004414	012762	001777	000020	
2328	004422	012762	003400	000006	
2329	004430	012762	000007	000010	
2330	004436	012762	012021	000000	
2331					
2332	004444	012700	000016		
2333	004450	012762	000440	000026	1\$:
2334	004456	012762	000040	000026	
2335	004464	005300			
2336	004466	001370			
2337	004470	016237	000000	003140	
2338	004476	016237	000034	003166	
2339	004504	016237	000036	003170	
2340	004512	012737	012021	003200	
2341	004520	012737	071027	003226	
2342	004526	012737	037760	003230	
2343	004534	023737	003200	003140	
2344	004542	001405			
2345	004544	104003			
2346	004546	012762	100000	000000	
2347	004554	000412			
2348					
2349	004556	023737	003226	003166	2\$:
2350	004564	001401			
2351	004566	104004			
2352	004570	023737	003230	003170	3\$:
2353	004576	001401			
2354	004600	104005			
2355					
2356					
2357					
2358					
2359					
2360					
2361					
2362					
2363					

K04

```

2364
2365
2366 004602 000004
2367 004604 012737 000062 001200
2368 004612 013702 001270
2369 004616 012762 100000 000000
2370 004624 012762 000040 000026
2371 004632 012762 177777 000002
2372 004640 012762 055220 000004
2373 004646 012762 001777 000020
2374 004654 012762 003400 000006
2375 004662 012762 000007 000010
2376 004670 012762 012023 000000
2377
2378 004676 012700 000016
2379 004702 012762 000440 000026 1$:
2380 004710 012762 000040 000026
2381 004716 005300
2382 004720 001370
2383 004722 016237 000000 003140
2384 004730 016237 000034 003166
2385 004736 016237 000036 003170
2386 004744 012737 012023 003200
2387 004752 012737 071227 003226
2388 004760 012737 037760 003230
2389 004766 023737 003200 003140
2390 004774 001405
2391 004776 104006
2392 005000 012762 100000 000000
2393 005006 000412
2394
2395 005010 023737 003226 003166 2$:
2396 005016 001401
2397 005020 104007
2398 005022 023737 003230 003170 3$:
2399 005030 001401
2400 005032 104010
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413 005034 000004
2414 005036 012737 000050 001200
2415 005044 013702 001270
2416 005050 012762 100000 000000
2417 005056 012762 000040 000026
2418 005064 012762 177777 000002
2419 005072 012762 055220 000004

```

```

;*****
;S12: SCOPE
MOV #50, $TIMES ;; DO 50. ITERATIONS
MOV $BASE, R2 ;; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;; CLEAR RK611
MOV #DMD, RKMR1(R2) ;; PUT RK611 IN DIAGNOSTIC MODE
MOV #-1, RKWC(R2) ;; LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ;; LOAD DUMMY BUS ADDRESS
MOV #1777, RKDCYL(R2) ;; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ;; LOAD TRACK
MOV #7, RKCS2(R2) ;; LOAD DRIVE NUM.
MOV #CDT!CFMT!WRDATA, RKCS1(R2) ;; ISSUE WRDATA WITH CDT SET IN
;; 24 SECTOR FORMAT
;; CLOCK IN DRIVE MESSAGE
1$: MOV #3*4+2, R0
MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2), T.CS1 ;; STORE COMMAND STATUS REG. 1
MOV RKMR2(R2), T.MR2 ;; STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2), T.MR3 ;; STORE MAINT REG. 3 (MESS B)
MOV #CDT!CFMT!WRDATA, E.CS1 ;; LOAD EXPECTED CS1
MOV #5, SEEK!S.RTC!S.FMT!70007, E.MR2 ;; LOAD EXPECTED MR2
MOV #37760, E.MR3 ;; LOAD EXPECTED MR3
CMP E.CS1, T.CS1 ;; CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2$ ;; YES, CHECK MESSAGE A
ERROR 6 ;; CS1 INCORRECT
MOV #CCLR, RKCS1(R2) ;; CLEAR RK611
BR TST3 ;; GO TO NEXT TEST
2$: CMP E.MR2, T.MR2 ;; CHECK MESS A CORRECT
BEQ 3$ ;; YES, CHECK MESSAGE B
ERROR 7 ;; MESS A INCORRECT
3$: CMP E.MR3, T.MR3 ;; CHECK MESS B CORRECT
BEQ TST3 ;; YES, GO ON TO NEXT TEST
ERROR 10 ;; MESS B INCORRECT

```

```

;*****
;TEST 3 SEEK MESSAGE FOR WRITE CHECK
;
; CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
; PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK
; OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
; CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
; IN SEEK COMMAND. MAKE SURE CORRECT MESSAGE
; IS ASSEMBLED.
;*****

```

```

;S13: SCOPE
MOV #50, $TIMES ;; DO 50 ITERATIONS
MOV $BASE, R2 ;; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;; CLEAR RK611
MOV #DMD, RKMR1(R2) ;; PUT RK611 IN DIAGNOSTIC MODE
MOV #-1, RKWC(R2) ;; LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ;; LOAD DUMMY BUS ADDRESS

```

L04

```

2420 005100 012762 001777 000020      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2421 005106 012762 003400 000006      MOV      #3400,RKDA(R2) ;LOAD TRACK
2422 005114 012762 000007 000010      MOV      #7,RKCS2(R2) ;LOAD DRIVE NUM.
2423 005122 012762 012031 000000      MOV      #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE WRTCHK WITH CDT SET IN
2424                                     ; 24 SECTOR FORMAT
2425 005130 012700 000016                                     ;CLOCK IN DRIVE MESSAGE
2426 005134 012762 000440 000026 15:    MOV      #DMD!MCLK,RKMR1(R2)
2427 005142 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2428 005150 005300                                     DEC      R0
2429 005152 001370                                     BNE     15
2430 005154 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2431 005162 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2432 005170 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2433 005176 012737 012031 003200      MOV      #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2434 005204 012737 071027 003226      MOV      #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2435 005212 012737 037760 003230      MOV      #37760,E.MR3 ;LOAD EXPECTED MR3
2436 005220 023737 003200 003140      CMP      E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2437 005226 001405                                     BEQ     25
2438 005230 104011                                     ERROR   11 ;YES, CHECK MESSAGE A
2439 005232 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CS1 INCORRECT
2440 005240 000412                                     BR      TST4 ;CLEAR RK611
2441                                     ;GO TO NEXT TEST

```

```

2442 005242 023737 003226 003166 25:    CMP      E.MR2,T.MR2 ;CHECK MESS A CORRECT
2443 005250 001401                                     BEQ     35 ;YES, CHECK MESSAGE B
2444 005252 104012                                     ERROR   12 ;MESS A INCORRECT
2445 005254 023737 003230 003170 35:    CMP      E.MR3,T.MR3 ;CHECK MESS B CORRECT
2446 005252 001401                                     BEQ     TST4 ;YES, GO ON TO NEXT TEST
2447 005264 104013                                     ERROR   13 ;MESS B INCORRECT

```

*TEST 4 READ DATA CLEAR MESSAGE

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE CONTROLLER ON DIAGNOSTIC MODE. ISSUE READ DATA TO AN RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND THE CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT.

*TST4: SCOPE

```

2460 005266 000004                                     TST4:  MOV      #50,%TIMES ;DO 50. ITERATIONS
2461 005270 012737 000062 001200      MOV      $BASE,R2 ;LOAD RK611 BASE
2462 005276 013702 001270                                     MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2463 005302 012762 100000 000000      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2464 005310 012762 000040 000026      MOV      #-1,RKWC(R2) ;LOAD WORD COUNT
2465 005316 012762 177777 000002      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2466 005324 012762 055220 000004      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2467 005332 012762 001777 000020      MOV      #3400,RKDA(R2) ;LOAD TRACK
2468 005340 012762 003400 000006      MOV      #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2469 005346 012762 000007 000010      MOV      #CDT!CFMT!RDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2470 005354 012762 012021 000000                                     ; 24 SECTOR FORMAT
2471 005362 012700 000156                                     ;LOAD COUNT TO LOAD DRIVE CLEAR
2472 005366 012762 000440 000026 15:    MOV      #DMD!MCLK,RKMR1(R2)
2473 005374 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2474 005402 005300                                     DEC      R0
2475 005404 001370                                     BNE     15

```

M04

2476	005406	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2477	005414	016237	000034	003166		MOV	RKMR2(R2),T.MR2	:STORE MAINT. REG. 2 (MESS A)
2478	005422	016237	000036	003170		MOV	RKMR3(R2),T.MR3	:STORE MAINT. REG. 3 (MESS B)
2479	005430	012737	012021	003200		MOV	#CDT!CFMT!RDATA,E.CS1	:LOAD EXPECTED CSI
2480	005436	012737	071407	003226		MOV	#S.CLR!S.FMT!70007,E.MR2	:LOAD EXPECTED MAINT REG. 2
2481	005444	005037	003230			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2482	005450	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
2483	005456	001405				BEQ	2\$:YES, CHECK MESSB
2484	005460	104014				ERROR	14	:CSI INCORRECT
2485	005462	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
2486	005470	000412				BR	TSTS	::GO TO NEXT TEST
2487								
2488	005472	023737	003226	003166	2\$:	CMP	E.MR2,T.MR2	:CHECK MESS A CORRECT
2489	005500	001401				BEQ	3\$:YES, CHECK MESS B
2490	005502	104015				ERROR	15	:MESS A INCORRECT
2491	005504	023737	003230	003170	3\$:	CMP	E.MR3,T.MR3	:CHECK MESS B CORRECT
2492	005512	001401				BEQ	TSTS	::YES, GO ON TO NEXT TEST
2493	005514	104016				ERROR	16	:MESS B INCORRECT
2494								
2495								
2496								
2497								
2498								
2499								
2500								
2501								
2502								
2503								
2504								
2505	005516	000004				TSTS:	SCOPE	
2506	005520	012737	000062	001200		MOV	#50,\$TIMES	:DO 50. ITERATIONS
2507	005528	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
2508	005532	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
2509	005540	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
2510	005546	012762	177777	000002		MOV	#-1,RKWC(R2)	:LOAD WORD COUNT
2511	005554	012762	055220	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
2512	005562	012762	001777	000020		MOV	#1777,RKDCYL(R2)	:LOAD CYLINDER ADDRESS REG.
2513	005570	012762	003400	000006		MOV	#3400,RKDA(R2)	:LOAD TRACK
2514	005576	012762	000007	000010		MOV	#7,RKCS2(R2)	:LOAD DRIVE NUMBER
2515	005604	012762	012023	000000		MOV	#CDT!CFMT!WRDATA,RKCS1(R2)	:ISSUE COMMAND WITH CDT SET IN 24 SECTOR FORMAT
2516								
2517	005612	012700	000156			MOV	#27.*4+2,R0	:LOAD COUNT TO LOAD DRIVE CLEAR
2518	005616	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2519	005624	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2520	005632	005300				DEC	R0	
2521	005634	001370				BNE	1\$	
2522	005636	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2523	005644	016237	000034	003166		MOV	RKMR2(R2),T.MR2	:STORE MAINT. REG. 2 (MESS A)
2524	005652	016237	000036	003170		MOV	RKMR3(R2),T.MR3	:STORE MAINT. REG. 3 (MESS B)
2525	005660	012737	012023	003200		MOV	#CDT!CFMT!WRDATA,E.CS1	:LOAD EXPECTED CSI
2526	005666	012737	071407	003226		MOV	#S.CLR!S.FMT!70007,E.MR2	:LOAD EXPECTED MAINT REG. 2
2527	005674	005037	003230			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2528	005700	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
2529	005706	001405				BEQ	2\$:YES, CHECK MESSB
2530	005710	104017				ERROR	17	:CSI INCORRECT
2531	005712	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611

```

*****
*TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT. CYLINDER 1777
* TRACK 7. DRIVE 7. CLOCK IN SEEK MESSAGE AND DRIVE CLEAR.
* CHECK IF PROPER DRIVE CLEAR MESSAGE IS ASSEMBLED.
*
*****

```

```

*****
TSTS: SCOPE
MOV #50,$TIMES ;DO 50. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #-1,RKWC(R2) ;LOAD WORD COUNT
MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
MOV #3400,RKDA(R2) ;LOAD TRACK
MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
MOV #CDT!CFMT!WRDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
; 24 SECTOR FORMAT
;LOAD COUNT TO LOAD DRIVE CLEAR
MOV #27.*4+2,R0
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
MOV #CDT!CFMT!WRDATA,E.CS1 ;LOAD EXPECTED CSI
MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
CLR E.MR3 ;LOAD EXPECTED MAINT REG.
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ 2$ ;YES, CHECK MESSB
ERROR 17 ;CSI INCORRECT
MOV #CLR,RKCS1(R2) ;CLEAR RK611

```

NO4

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 52
 DZR6DA.P11 28-SEP-76 15:50 T5 WRITE DATA AND DRIVE CLEAR MESSAGE

SEQ 0052

```

2532 005720 000412 BR TST6 ;;GO TO NEXT TEST
2533
2534 005722 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2535 005730 001401 BEQ 3$ ;YES, CHECK MESS B
2536 005732 104020 ERROR 20 ;MESS A INCORRECT
2537 005734 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2538 005742 001401 BEQ TST6 ;YES, GO ON TO NEXT TEST
2539 005744 104021 ERROR 21 ;MESS B INCORRECT
2540
2541
2542 *****
2543 *TEST 6 DRIVE CLEAR MESSAGE FOR WRITE CHECK
2544 *
2545 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2546 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
2547 * CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT.
2548 * CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
2549 * THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR.
2550 * MAKE SURE CORRECT MESSAGE IS ASSEMBLED.
2551 *
2552 *****
2553 †ST6: SCOPE
2554 MOV #50, $TIMES ;;DO 50. ITERATIONS
2555 MOV $BASE,R2 ;LOAD RK611 BASE
2556 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2557 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2558 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2559 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2560 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2561 MOV #3400,RKDA(R2) ;LOAD TRACK
2562 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2563 MOV #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2564 ; 24 SECTOR FORMAT
2565 MOV #27,*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2566 1$: MOV #DMD!MCLK,RKMR1(R2)
2567 MOV #DMD,RKMR1(R2)
2568 DEC R0
2569 BNE 1$
2570 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2571 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2572 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2573 MOV #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2574 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2575 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2576 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2577 BEQ 2$ ;YES, CHECK MESSB
2578 ERROR 22 ;CS1 INCORRECT
2579 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2580 BR TST7 ;;GO TO NEXT TEST
2581
2582 2581 006152 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2583 006160 001401 BEQ 3$ ;YES, CHECK MESS B
2584 006162 104023 ERROR 23 ;MESS A INCORRECT
2585 006164 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2586 006172 001401 BEQ TST7 ;YES, GO ON TO NEXT TEST
2587 006174 104024 ERROR 24 ;MESS B INCORRECT
    
```


.SBTTL **HEADER GENERATION

TEST 7 HEADER GENERATION (PART 1)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3 TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR CYLINDERS 1-1777.

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

006176 000004
006200 010000 000012 001200
006206 010000 001270
006212 005037 003220
006216 005037 003206
006222 012737 000021 003200
006230 012737 006236 001110

006236
006236 004737 036010
006242 012762 100000 000000
006250 012762 000040 000026
006256 012762 177777 000002
006264 012762 055220 000004
006272 013762 003220 000020
006300 013762 003206 000006
006306 013762 003200 000000
006314 012700 000426
006320 012762 000440 000026
006326 012762 000040 000026
006334 005300
006336 001370
006340 016237 000000 003140
006346 016237 000034 003166
006354 016237 000036 003170
006362 023737 003200 003140
006370 001405
006372 104025
006374 012762 100000 000000
006402 000412

006404 023737 003226 003166
006412 001401
006414 104026
006416 023737 003230 003170
006424 001401
006426 104027
006430 104415
006432 005237 003220
006436 022737 000633 003220
006444 001002

57: SCOPE
MOV #10, STIMES ; DO 10 ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
CLR E.DCYL ; INITIALIZE CYLINDER ADDR
CLR E.DA ; INITIALIZE TRACK AND SECTOR
MOV #RDDATA, E.CS1 ; INITIALIZE FORMAT
MOV #15, \$LPERR ; LOAD LOOP ON ERROR LOCATION FOR
; SUBTEST LOOP

15: JSR PC, CALHDR ; CALCULATE EXPECTED HEADER
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMRI(R2) ; PUT RK611 IN MAINTENANCE MODE
MOV #-1, RKWC(R2) ; LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV E.DCYL, RKDCYL(R2) ; LOAD CYLINDER NUMBER
MOV E.DA, RKDA(R2) ; LOAD TRACK AND SECTOR
MOV E.CS1, RKCS1(R2) ; LOAD COMMAND AND FORMAT
MOV #69, *4+2, R0 ; LOAD COUNT UNTIL HEADER GENERATION
55: MOV #DMD, MCLK, RKMRI(R2)
MOV #DMD, RKMRI(R2)
DEC R0
BNE 55
MOV RKCS1(R2), T.CS1 ; STORE COMMAND AND STATUS REG 1
MOV RKMRI(R2), T.MR2 ; STORE FIRST HEADER WORD
MOV RKMRI(R2), T.MR3 ; STORE SECOND WORD
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ 65 ; YES, CHECK FIRST LINE OF HEADER
ERROR 25 ; CS1 INCORRECT
MOV #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
BR 155 ; YES, CHECK SECOND WORD OF HEADER

65: CMP E.MR2, T.MR2 ; CHECK IF FIRST WORD OF HEADER CORRECT
BEQ 75 ; YES, CHECK IF SECOND WORD OF HEADER CORRECT
ERROR 26 ; FIRST WORD OF HEADER INCORRECT
75: CMP E.MR3, T.MR3 ; CHECK IF SECOND WORD OF HEADER CORRECT
BEQ 155 ; YES, CHECK IF LOOP ON ERROR
ERROR 27 ; SECOND WORD INCORRECT
155: SCOP1 ; LOOP ON ERROR
INC E.DCYL ; INCREMENT EXPECTED CYLINDER
CMP #633, E.DCYL ; CHECK IF VALUE OF CYLINDER OVERFLOW DETECTION
BNE 165 ; NO, CONTINUE

C05

```

2644 006446 005237 003220      INC      E.DCYL      :USE 634
2645 006452 022737 001777 003220 16$:  CMP      #1777,E.DCYL :CHECK IF FINISHED
2646 006460 103266      BHS      IS         :NO, GO TO NEXT CONFIGURATION
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659 006462 000004      *ST10: SCOPE
2660 006464 012737 000012 00120C  MOV      #10, $TIMES  ;;DO 10. ITERATIONS
2661 006472 013702 001270      MOV      $BASE,R2    :LOAD RK611 BASE
2662 006476 005037 003220      CLR      E.DCYL     :INITIALIZE CYLINDER ADD
2663 006502 005037 003206      CLR      E.DA       :INITIALIZE TRACK AND SECTOR
2664 006506 012737 000021 003200  MOV      #RDDATA,E.CS1 :INITIALIZE FORMAT
2665 006514 012737 006522 001110  MOV      #IS,$LPERR  :LOAD LOOP ON ERROR LOCATION FOR
2666                                     : SUBTEST LOOP
2667
2668
2669 006522      1$:
2670 006522 004737 036010      JSR      PC,CALHDR   :CALULATE EXPECTED HEADER
2671 006526 012762 100000 000000  MOV      #CCLR,RKCS1(R2) :CLEAR RK611
2672 006534 012762 000040 000026  MOV      #DMD,RKMR1(R2) :PUT RK611 IN MAINTENANCE MODE
2673 006542 012762 177777 000002  MOV      #-1,RKWC(R2)  :LOAD WORD COUNT
2674 006550 012762 055220 000004  MOV      #9UFF,RKBA(R2) :LOAD DUMMY BUS ADDRESS
2675 006556 013762 003220 000020  MOV      E.DCYL,RKDCYL(R2) :LOAD CYLINDER NUMBER
2676 006564 013762 003206 000006  MOV      E.DA,RKDA(R2)  :LOAD TRACK AND SECTOR
2677 006572 013762 003200 000000  MOV      E.CS1,RKCS1(R2) :LOAD COMMAND AND FORMAT
2678 006600 012700 000426      MOV      #69,*4+2,R0  :LOAD COUNT UNTIL HEADER GENERATION
2679 006604 012762 000440 000026  5$:  MOV      #DMD,$MCLK,RKMR1(R2)
2680 006612 012762 000040 000026  MOV      #DMD,RKMR1(R2)
2681 006622 001370      DEC      R0
2682 006624 016237 000000 003140  BNE      5$
2683 006632 016237 000034 003166  MOV      RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG 1
2684 006640 016237 000036 003170  MOV      RKMR2(R2),T.MR2 :STORE FIRST HEADER WORD
2685 006646 023737 003200 003140  MOV      RKMR3(R2),T.MR3 :STORE SECOND WORD
2686 006654 001405      CMP      E.CS1,T.CS1  :CHECK COMMAND AND STATUS REG 1 CORRECT
2687 006656 104030      BEQ      6$          :YES, CHECK FIRST LINE OF HEADER
2688 006660 012762 100000 000000  ERROR   30          :CS1 INCORRECT
2689 006666 000412      MOV      #CCLR,RKCS1(R2) :CLEAR CONTROLLER
2690                                     BF      15$          :YES, CHECK SECOND WORD OF HEADER
2691
2692 006670 023737 003226 003166  6$:  CMP      E.MR2,T.MR2  :CHECK IF FIRST WORD OF HEADER CORRECT
2693 006676 001401      BEQ      7$          :YES, CHECK IF SECOND WORD OF HEADER CORRECT
2694 006700 104031      ERROR   31          :FIRST WORD OF HEADER INCORRECT
2695 006702 023737 003230 003170  7$:  CMP      E.MR3,T.MR3  :CHECK IF SECOND WORD OF HEADER CORRECT
2696 006710 001401      BEQ      15$         :YES, CHECK IF LOOP ON ERROR
2697 006712 104032      ERROR   32          :SECOND WORD INCORRECT
2698 006714 104415      15$:  SCOPI
2699 006716 105237 003207      INCB     E.DA+1      :INCREMENT TRACK
2700 006722 122737 000007 003207  CMPB     #7,E.DA+1  :CHECK IF FINISHED
    
```

2700 006730 103274 BHIS 1\$;NO, GO TO NEXT CONFIGURATION

:TEST 11 HEADER GENERATION (PART 3)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
* ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
* 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
* CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
* TO MAKE SURE HEADER IS CORRECT. REPEAT FOR SECTORS 1-25.

2713 006732 000004
2714 006734 012737 000012 001200
2715 006742 013702 001270
2716 006746 005037 003220
2717 006752 005037 003206
2718 006756 012737 000021 003200
2719 006764 012737 006772 001110

TEST11: SCOPE
MOV #10, \$TIMES ;DO 10. ITERATIONS
MOV \$BASE, R2 ;LOAD RK611 BASE
CLR E.DCYL ;INITIALIZE CYLINDER ADD
CLR E.DA ;INITIALIZE TRACK AND SECTOR
MOV #RDDATA, E.CS1 ;INITIALIZE FORMAT
MOV #15, \$LPERR ;LOAD LOOP ON ERROR LOCATION FOR
; SUBTEST LOOP

2722 006772
2723 006772 004737 036010
2724 006776 012762 100000 000000
2725 007004 012762 000040 000026
2726 007012 012762 177777 000002
2727 007020 012762 055220 000004
2728 007026 013762 003220 000020
2729 007034 013762 003206 000006
2730 007042 013762 003200 000000
2731 007050 012700 000426
2732 007054 012762 000440 000026
2733 007062 012762 000040 000026
2734 007070 005300
2735 007072 001370
2736 007074 016237 000000 003140
2737 007102 016237 000034 003166
2738 007110 016237 000036 003170
2739 007116 023737 003200 003140
2740 007124 001405
2741 007126 104033
2742 007130 012762 100000 000000
2743 007136 000412

1\$: JSR PC, CALHDR ;CALCULATE EXPECTED HEADER
MOV #CLR, RKCS1(R2) ;CLEAR RK611
MOV #DMD, RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
MOV #-1, RKWC(R2) ;LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV E.DCYL, RKDCYL(R2) ;LOAD CYLINDER NUMBER
MOV E.DA, RKDA(R2) ;LOAD TRACK AND SECTOR
MOV E.CS1, RKCS1(R2) ;LOAD COMMAND AND FORMAT
MOV #69, *4+2, R0 ;LOAD COUNT UNTIL HEADER GENERATION
5\$: MOV #DMD, MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 5\$
MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG 1
MOV RKMR2(R2), T.MR2 ;STORE FIRST HEADER WORD
MOV RKMR3(R2), T.MR3 ;STORE SECOND WORD
CMP E.CS1, T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ 6\$;YES, CHECK FIRST LINE OF HEADER
ERROR 33 ;CS1 INCORRECT
MOV #CLR, RKCS1(R2) ;CLEAR CONTROLLER
BR 15\$;YES, CHECK SECOND WORD OF HEADER

2745 007140 023737 003226 003166 6\$:
2746 007146 001401
2747 007150 104034
2748 007152 023737 003230 003170 7\$:
2749 007160 001401
2750 007162 104035
2751 007164 104415
2752 007166 105237 003206
2753 007172 122737 000025 003206
2754 007200 103274
2755

6\$: CMP E.MR2, T.MR2 ;CHECK IF FIRST WORD OF HEADER CORRECT
BEQ 7\$;YES, CHECK IF SECOND WORD OF HEADER CORRECT
ERROR 34 ;FIRST WORD OF HEADER INCORRECT
7\$: CMP E.MR3, T.MR3 ;CHECK IF SECOND WORD OF HEADER CORRECT
BEQ 15\$;YES, CHECK IF LOOP ON ERROR
ERROR 35 ;SECOND WORD INCORRECT
15\$: SCOP1 ;LOOP ON ERROR
INCB E.DA ;INCREMENT SECTOR
CMPB #25, E.DA ;CHECK IF FINISHED
BHIS 1\$;NO, GO TO NEXT CONFIGURATION

E05

```

2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768 007202 000004
2769 007204 012737 000062 001200
2770 007212 013702 001270
2771 007216 005037 003220
2772 007222 005037 003206
2773 007226 012737 000021 003200
2774 007234 012737 007242 001110
2775
2776
2777 007242
2778 007242 004737 036010
2779 007246 012762 100000 000000
2780 007254 012762 000040 000026
2781 007262 012762 177777 000002
2782 007270 012762 055220 000004
2783 007276 013762 003220 000020
2784 007304 013762 003206 000006
2785 007312 013762 003200 000000
2786 007320 012700 000426
2787 007324 012762 000440 000026
2788 007332 012762 000040 000026
2789 007340 005300
2790 007342 001370
2791 007344 016237 000000 003140
2792 007352 016237 000034 003166
2793 007360 016237 000036 003170
2794 007366 023737 003200 003140
2795 007374 001405
2796 007376 104036
2797 007400 012762 100000 000000
2798 007406 000412
2799
2800 007410 023737 003226 003166
2801 007416 001401
2802 007420 104037
2803 007422 023737 003230 003170
2804 007430 001401
2805 007432 104040
2806 007434 104415
2807 007436 032737 010000 003200
2808 007444 001004
2809 007446 052737 010000 003200
2810 007454 000672
2811

```

```

*****
*TEST 12      HEADER GENERATION (PART 4)
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
* ONE WORD FOR AND RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
* HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
* MESSAGES, CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE
* REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR 24
* SECTOR FORMAT.
*****
TST12: SCOPE
MOV      #50, $TIMES      ;; DO 50 ITERATIONS
MOV      $BASE, R2        ;; LOAD RK611 BASE
CLR      E.DCYL           ;; INITIALIZE CYLINDER ADD
CLR      E.DA             ;; INITIALIZE TRACK AND SECTOR
MOV      $RCDATA, E.CS1   ;; INITIALIZE FORMAT
MOV      #15, $SLPERR     ;; LOAD LOOP ON ERROR LOCATION FOR
                          ;; SUBTEST LOOP

15:
JSR      PC, CALHDR       ;; CALCULATE EXPECTED HEADER
MOV      #0, RKC51(R2)    ;; CLEAR RK611
MOV      #0, RKMRI(R2)    ;; PUT RK611 IN MAINTENANCE MODE
MOV      #-1, RKWC(R2)    ;; LOAD WORD COUNT
MOV      $BUFF, RKBA(R2)  ;; LOAD DUMMY BUS ADDRESS
MOV      E.DCYL, RKDCYL(R2) ;; LOAD CYLINDER NUMBER
MOV      E.DA, RKDA(R2)   ;; LOAD TRACK AND SECTOR
MOV      E.CS1, RKC51(R2) ;; LOAD COMMAND AND FORMAT
MOV      #69, *4+2, R0     ;; LOAD COUNT UNTIL HEADER GENERATION
55:
MOV      #0, MCLK, RKMRI(R2)
MOV      #0, RKMRI(R2)
R0
DEC
BNE
55
MOV      RKC51(R2), T.CS1  ;; STORE COMMAND AND STATUS REG 1
MOV      RKMRI(R2), T.MR2  ;; STORE FIRST HEADER WORD
MOV      RKMRI(R2), T.MR3  ;; STORE SECOND WORD
CMP      E.CS1, T.CS1     ;; CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ      65
ERROR   36                ;; YES, CHECK FIRST LINE OF HEADER
;; CS1 INCORRECT
MOV      #0, RKC51(R2)    ;; CLEAR CONTROLLER
BR      155
;; YES, CHECK SECOND WORD OF HEADER

65:
CMP      E.MR2, T.MR2     ;; CHECK IF FIRST WORD OF HEADER CORRECT
BEQ      75
ERROR   37                ;; YES, CHECK IF SECOND WORD OF HEADER CORRECT
;; FIRST WORD OF HEADER INCORRECT
75:
CMP      E.MR3, T.MR3     ;; CHECK IF SECOND WORD OF HEADER CORRECT
BEQ      155
ERROR   40                ;; YES, CHECK IF LOOP ON ERROR
;; SECOND WORD INCORRECT
155:
SCOPI
BIT      #CFMT, E.CS1     ;; CHECK IF FINISHED
BNE
TST13
;; YES, GO TO NEXT TEST
BIS      #CFMT, E.CS1    ;; USE 24 SECTOR FORMAT
BR      15

```



F05

.SBTTL **NPR TRANSFER FOR WRITE DATA

*TEST 13 WRITE DATA NPR TRANSFER

* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* 67 WORDS, TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
* TRACK 0, DRIVE 0. CLOCK IN SEEK AND DRIVE CLEAR MESSAGES.
* GIVE ENOUGH CLOCK PULSE FOR 68 SILO WORDS. MAKE SURE DATA
* LATE DOES NOT OCCUR. READ BACK 66 WORDS AND VERIFY THEY
* ARE CORRECT.

```
TST13: SCOPE
MOV #50, $TIMES ;; DO 50. ITERATIONS
MOV $BASE, R2 ;; LOAD RK611 BASE
MOV #CLR, RKCS1(R2) ;; CLEAR RK611
MOV #DMD, RKMRI(R2) ;; PUT RK611 IN DIAGNOSTIC MODE
MOV #-67, RKWC(R2) ;; WORD COUNT=67
MOV #NPRBUF, RKBA(R2) ;; LOAD BUFFER ADDRESS
MOV #WRDATA, RKCS1(R2) ;; ISSUE WRDATA
MOV #68, *37, R0 ;; ISSUE ENOUGH CLOCKS FOR 68
1$: MOV #DMD, MCLK, RKMRI(R2) ; NPR TRANSFERS
MOV #DMD, RKMRI(R2)
DEC R0
BNE 1$
MOV RKCS1(R2), T.CS1 ;; STORE COMMAND AND STATUS REG. 1
MOV RKCS2(R2), T.CS2 ;; STORE COMMAND AND STATUS REG. 2
MOV RKWC(R2), T.WC ;; STORE WORD COUNT REG.
MOV RKBA(R2), T.BA ;; STORE BUS ADDRESS
MOV RKR(R2), T.ER ;; STORE ERROR REG.
MOV #WRDATA, E.CS1 ;; LOAD EXPECTED CS1
MOV #OR, E.CS2 ;; LOAD EXPECTED CS2
MOV #-1, E.WC ;; LOAD EXPECTED WORD COUNT
MOV #NPRBUF+(66.*2), E.BA ;; LOAD EXPECTED BUS ADDRESS
CLR E.ER ;; LOAD EXPECTED ERROR REG
CMP E.CS1, T.CS1 ;; CHECK CS1 CORRECT
BEQ 5$ ;; YES, CHECK CS2
ERROR 41 ;; CS1 INCORRECT
MOV #CLR, RKCS1(R2) ;; CLEAR RK611
BR TST14 ;; GO TO NEXT TEST

5$: CMP E.CS2, T.CS2 ;; CHECK CS2
BEQ 6$ ;; NO, CHECK BUS ADDRESS
ERROR 42 ;; CS2 INCORRECT
6$: CMP E.BA, T.BA ;; CHECK BUS ADDRESS
BEQ 7$ ;; YES, CHECK WORD COUNT
ERROR 44 ;; BUS ADDRESS INCORRECT
7$: CMP E.WC, T.WC ;; CHECK WORD COUNT
BEQ 8$ ;; YES, CONTINUE
ERROR 45 ;; WORD COUNT INCORRECT
8$: CMP E.ER, T.ER ;; CHECK ERROR REG CORRECT
BEQ 10$ ;; YES, CONTINUE
ERROR 43 ;; ERROR REG INCORRECT
10$: MOV #NPRBUF, R3 ;; LOAD ADDRESS OR START OF BUFFER
```

Table with 5 columns: Address (e.g., 2812, 2813, 2814), Hex Data (e.g., 007456, 012737), Hex Data (e.g., 000062, 001200), Hex Data (e.g., 001270, 000000), Hex Data (e.g., 000049, 000026). The table lists memory addresses and their corresponding data values for the diagnostic test.

```

2868 007744 012701 000101      MOV      #65.,R1      ;LOAD COUNT FOR SICC
2869 007750 005037 003264      CLR      WRDCNT      ;INITIALIZE WORD COUNT
2870 007754 012737 000300 003210  MOV      #IR,OR,E.CS2 ;LOAD EXPECTED CS2
2871 007762 012337 003222 15$:  MOV      (R3)+,E.DB   ;LOAD EXPECTED DATA
2872 007766 016237 000024 003162  MOV      RKDB(R2),T.DB ;GET DATA READ
2873 007774 012700 000020      MOV      #20,R0      ;SET COUNT TO WAIT FOR SILO
2874 010000 005300 16$:  DEC      R0          ;DEC COUNT
2875 010002 001376      BNE     16$         ;WAIT FOR 0
2876 010004 016237 000000 003140  MOV      RKCS1(R2),T.CS1 ;STORE CS1
2877 010012 016237 000010 003150  MOV      RKCS2(R2),T.CS2 ;STORE CS2
2878 010020 016237 000014 003154  MOV      RKR(R2),T.ER   ;STORE ERROR REG.
2879 010026 023737 003200 003140  CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
2880 010034 001405      BEQ     20$         ;YES, CONTINUE
2881 010036 104046      ERROR   46          ;CS1 INCORRECT
2882 010040 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2883 010046 000427      BR     TST14       ;GO ON TO NEXT TEST
2884
2885 010050 023737 003210 003150 20$:  CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
2886 010056 001401      BEQ     21$         ;YES, CONTINUE
2887 010060 104047      ERROR   47          ;CS2 INCORRECT
2888 010062 023737 003214 003154 21$:  CMP      E.ER,T.ER   ;CHECK ERROR REG CORRECT
2889 010070 001401      BEQ     22$         ;YES, CONTINUE
2890 010072 104050      ERROR   50          ;ERROR REG INCORRECT
2891 010074 023737 003222 003162 22$:  CMP      E.DB,T.DB   ;CHECK DATA CORRECT
2892 010102 001401      BEQ     25$         ;YES, CONTINUE
2893 010104 104051      ERROR   51          ;DATA INCORRECT
2894 010106 005237 003264 25$:  INC      WRDCNT      ;INCREMENT WORD COUNT
2895 010112 005301      DEC      R1          ;
2896 010114 001003      BNE     26$         ;CHECK IF LAST WORD
2897 010116 012737 000100 003210  MOV      #IR,E.CS2   ;UPDATE EXPECTED CS2
2898 010124 100316      BPL     15$         ;CHECK IF FINISHED
2899

```

.SBTTL **HEADER RECOGNITION TESTS

```

*****
*TEST 14 WRITE DATA HEADER RECOGNITION

```

```

*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE
* WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
* SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING
* DATA:

```

```

000000
140000
140000

```

MAKE SURE WRITE GATE SETS SHOWING CORRECT HEADER RECOGNITION.

```

*****

```

```

2919 010126 000004 TST14: SCOPE
2920 010130 012737 000012 001200  MOV      #10,$TIMES ;DO 10. ITERATIONS
2921 010136 013702 001270      MOV      $BASE,R2   ;LOAD RK611 BASE
2922 010142 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2923 010150 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE

```


H05

2924	010156	012762	177777	000002		MOV	#-1,RKWC(R2)	WORD COUNT=1
2925	010164	012762	055220	000004		MOV	#BUFF,RKBA(R2)	;LOAD DUMMY BUS ADDRESS
2926	010172	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA
2927	010200	012700	000426			MOV	#69,*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL READY
2928								;FOR SECTOR PULSE
2929	010204	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	
2930	010212	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2931	010220	005300				DEC	R0	
2932	010222	001370				BNE	15	
2933	010224	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
2934	010232	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2935	010240	012737	000023	003200		MOV	#WRDATA,E.CS1	;STORE EXPECTED CS1
2936	010246	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
2937	010252	005037	003256			CLR	M1.BIT	
2938	010256	005037	003262			CLR	BITCNT	
2939	010262	012700	000377			MOV	#255,R0	
2940	010266	004737	037030		55:	JSR	PC,RDBIT	
2941	010272	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2942	010300	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2943	010306	001112				BNE	635	;NO REPORT ERROR
2944	010310	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2945	010314	005300				DEC	R0	;CHECK IF SYNCH FINISHED
2946	010316	001363				BNE	55	
2947	010320	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
2948	010326	004737	037030			JSR	PC,RDBIT	
2949	010332	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2950	010340	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2951	010346	001072				BNE	635	;NO REPORT CS1
2952	010350	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2953	010354	012703	053366			MOV	#HEAD1,R3	;SIMULATE GOOD HEADER
2954	010360	012701	000003			MOV	#3,R1	
2955	010364	012304			125:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
2956	010366	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
2957	010372	013737	003254	003256	155:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
2958	010400	006004				ROR	R4	;GET NEXT BIT
2959	010402	103403				BCS	175	
2960	010404	005037	003254			CLR	PR.BIT	
2961	010410	000403				BR	185	
2962								
2963	010412	012737	000001	003254	175:	MOV	#1,PR.BIT	
2964	010420	004737	037030		185:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
2965	010424	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
2966	010432	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2967	010440	001035				BNE	635	;NO REPORT ERROR
2968	010442	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2969	010446	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
2970	010450	001350				BNE	155	;NO CONTINUE
2971	010452	005301				DEC	R1	;CHECK IF FINISHED WITH HEAD
2972	010454	001343				BNE	125	;NO CONTINUE
2973	010456	012700	000105			MOV	#69,R0	;LOAD COUNT FOR GAP
2974	010462	013737	003254	003256	255:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
2975	010470	005037	003254			CLR	PR.BIT	
2976	010474	004737	037030			JSR	PC,RDBIT	
2977	010500	005300				DEC	R0	
2978	010502	001367				BNE	255	
2979	010504	016237	000026	003164		MOV	RKMR1(R2),T.MR1	;GET MR1

I05

```

2980 010512 012737 062040 033224      MOV      #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;LOAD EXPECTED MR1
2981 010520 023737 003224 003164      CMP      E.MR1,T.MR1 ;CHECK IF WRITE GATE SET
2982 010526 001411                BEQ      TST15 ;:YES, GO ON TO NEXT TEST
2983 010530 104053                ERROR   53
2984 010532 000407                BR      TST15 ;:GO ON TO NEXT TEST
2985
2986 010534 016237 000010 003150 63$: MOV      RKCS2(R2),T.CS2 ;STORE CS2 FOR PRINT OUT
2987 010542 016237 000014 003154      MOV      RKER(R2),T.ER ;STORE ERROR REG FOR PRINT OUT
2988 010550 104052                ERROR   52 ;REPORT ERROR
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007 010552 000004                TST15: SCOPE
3008 010554 012737 000012 001200      MOV      #10, $TIMES ;:DO 10. ITERATIONS
3009 010562 013702 001270                MOV      $BASE,R2 ;LOAD RK611 BASE
3010 010566 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3011 010574 012700 000700                MOV      #700,RO ;SET STALL COUNT
3012 010600 005300                4$: DEC   RO ;DEC COUNT
3013 010602 001376                BNE     4$ ;LOOP UNTIL ZERO
3014 010604 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3015 010612 012762 177777 000002      MOV      #-1,RKWC(R2) ;WORD COUNT = 1
3016 010620 012762 055220 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3017 010626 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3018 010634 012700 000426                MOV      #69.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTILL READY
3019
3020 010640 012762 000440 000026 1$: MOV      #DMD!MCLK,RKMR1(R2)
3021 010646 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3022 010654 005300                DEC     RO
3023 010656 001370                BNE     1$
3024 010660 012700 000004                MOV      #4,RO ;SIMULATE INDEX PULSE
3025 010664 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3026 010672 012762 000640 000026 2$: MOV      #DMD!MIND!MCLK,RKMR1(R2)
3027 010700 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3028 010706 005300                DEC     RO
3029 010710 001370                BNE     2$
3030 010712 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3031 010720 005037 003254                CLR     PR.BIT ;GENERATE SYNCH
3032 010724 005037 003256                CLR     M1.BIT
3033 010730 012700 000377                MOV      #255,RO
3034 010734 004737 037030 5$: JSR     PC,ROBIT
3035 010740 005300                DEC     RO

```

```

*****
*TEST 15 SECTOR PULSE DETECTION FOR WRITE DATA
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE ISSUE A WRITE DATA OF ONE
* WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
* SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE AN INDEX PULSE AND A HEADER WITH THE FOLLOWING
* DATA:
*
* 000000
* 140000
* 140000
*
* MAKE SURE WRITE GATE DOES NOT SET.
*
*****

```



```

3036 010742 001374      BNE      5$
3037 010744 012737 000001 003254  MOV      #1,PR.BIT      ;SIMULATE SYNCH BIT
3038 010752 004737 037030      JSR      PC,RDBIT
3039 010756 012703 053366      MOV      #HEAD1,R3      ;SIMULATE GOOD HEADER
3040 010762 012701 000003      MOV      #3,R1
3041 010766 012304      MOV      (R3)+,R4      ;GET NEXT HEADER WORD
3042 010770 012700 000020      MOV      #16,R0      ;LOAD BITS PER WORDS
3043 010774 013737 003254 003256 15$:  MOV      PR.BIT,M1.BIT  ;STORE PREVIOUS BIT
3044 011002 006004      ROR      R4      ;GET NEXT BIT
3045 011004 103403      BCS      17$
3046 011006 005037 003254      CLR      PR.BIT
3047 011012 000403      BR       18$
3048
3049 011014 012737 000001 003254 17$:  MOV      #1,PR.BIT
3050 011022 004737 037030 18$:  JSR      PC,RDBIT      ;SIMULATE NEXT BIT
3051 011026 005300      DEC      R0      ;CHECK IF READY FOR NEXT HEADER WORD
3052 011030 001361      BNE      15$      ;NO,CONTINUE
3053 011032 005301      DEC      R1      ;CHECK IF FINISHED WITH HEADER
3054 011034 001354      BNE      12$      ;NO,CONTINUE
3055 011036 012700 000100      MOV      #64,R0      ;LOAD COUNT FOR GAP
3056 011042 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT  ;SIMULATE GAP
3057 011050 005037 003254      CLR      PR.BIT
3058 011054 004737 037030      JSR      PC,RDBIT
3059 011060 005300      DEC      R0
3060 011062 001367      BNE      25$
3061 011064 016237 000026 003164  MOV      RKMRI(R2),T.MRI ;GET MRI
3062 011072 012737 022040 003224  MOV      #DMD!ECCW!MEWD,E.MRI ;LOAD EXPECTED MRI
3063 011100 023737 003164 003224  CMP      T.MRI,E.MRI    ;CHECK IF WRITE GATE DID NOT SET
3064 011106 001401      BEQ      TST16        ;:YES, GO ON TO NEXT TEST
3065 011110 104053      ERROR    53

```

```

*****
*TEST 16      SECTOR INCREMENT
*
*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
*      CONTROLLER IN MAINTENANCE MODE.  ISSUE A WRITE DATA OF ONE WORD
*      TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0
*      SECTOR 0.  CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
*      SIMULATE SECTOR PULSE AND PROPER HEADER.  MAKE SURE
*      THAT WRITE GATE SETS,
*
*      REPEAT FOR SECTOR 1-24.

```

```

*****
TST16:  SCOPE
3080 011112 000004      MOV      #10,$TIMES      ;DO 10. ITERATIONS
3081 011114 012737 000012 001200  MOV      $BASE,R2      ;LOAD RK611 BASE
3082 011122 013702 001270      CLR      CYLN      ;INITIALIZE CYLINDER
3083 011126 005037 003300      CLR      SECTOR      ;INITIALIZE TRACK AND SECTOR
3084 011132 005037 003276      MOV      #1,$$LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
3085 011136 012737 011144 001110  ; SUBTEST LOOP
3086
3087
3088 011144      1$:  JSR      PC,INCHDR      ;GENERATE HEADER WORDS
3089 011144 004737 035626      MOV      #CLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3090 011150 012762 100000 000000  MOV      #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3091 011156 012762 000040 000026  MOV

```


3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203

011504 000004
011506 012737 000012 001200
011514 013702 001270
011520 005037 003300
011524 012737 000025 003276
011532 012737 011540 001110

011540
011540 004737 035626
011544 012762 100000 000000
011552 012762 000040 000026
011560 013762 003300 000020
011566 013762 003276 000006
011574 012762 177777 000002
011602 012762 055220 000004
011610 012762 000023 000000
011616 012700 000426

011622 012762 000440 000026
011630 012762 000040 000026
011636 005300
011640 001370
011642 012762 000140 000026
011650 012762 000040 000026
011656 005037 003254
011662 005037 003256
011666 012700 000377
011672 004737 037030
011676 005300
011700 001374
011702 012737 000001 003254
011710 004737 037030
011714 012703 003302
011720 012701 000003
011724 012304
011726 012700 000020
011732 013737 003254 003256
011740 006004
011742 103403
011744 005037 003254
011750 000403

```
*****  
:TEST 17 TRACK INCREMENT  
:  
: CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT  
: CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD  
: TO AN RK06 IN 26 SECTOR FORMAT. CYLINDER 0, HEAD 0,  
: SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.  
: SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE  
: THAT WRITE GATE SETS.  
:  
: REPEAT FOR TRACK = 1.  
:  
*****  
↑ST17: SCOPE  
MOV #10, $TIMES ;DO 10. ITERATIONS  
MOV $BASE, R2 ;LOAD RK611 BASE  
CLR CYLN ;INITIALIZE CYLINDER  
MOV #25, SECTOR ;INITIALIZE TRACK AND SECTOR  
MOV #1$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR  
; SUBTEST LOOP  
  
1$: JSR PC, INCHDR ;GENERATE HEADER WORDS  
MOV #CLR, RKCS1(R2) ;CLEAR RK611 CONTROLLER  
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
MOV CYLN, RKDCYL(R2) ;LOAD DESIRED CYLINDER  
MOV SECTOR, RKDA(R2) ;LOAD DESIRED TRACK/SECTOR  
MOV #-1, RKWC(R2) ;WORD COUNT=1  
MOV #BUFF, RKBA(R2) ;LOAD DUMMY ADDRESS  
MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA  
MOV #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY  
; FOR SECTOR PULSE  
  
5$: MOV #DMD!MCLK, RKMR1(R2)  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 5$  
MOV #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE  
MOV #DMD, RKMR1(R2)  
CLR PR.BIT ;GENERATE SYNCH  
CLR M1.BIT  
MOV #255, R0  
10$: JSR PC, RDBIT  
DEC R0  
BNE 10$  
MOV #1, PR.BIT ;SIMULATE SYNCH BIT  
JSR PC, RDBIT  
MOV #HEADER, R3 ;SIMULATE HEADER  
MOV #3, R1  
12$: MOV (R3)+, R4 ;GET NEXT HEADER WORD  
MOV #16, R0 ;LOAD BITS PER WORD  
15$: MOV PR.BIT, M1.BIT  
ROR R4  
BCS 17$  
CLR PR.BIT  
BR 18$
```

M05

```

3204 011752 012737 000001 003254 17$: MOV #1,PR.BIT
3205 011760 004737 037030 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3206 011764 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
3207 011766 001361 BNE 15$ ;NO, CONTINUE
3208 011770 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3209 011772 001354 BNE 12$ ;NO, CONTINUE
3210 011774 012700 000100 MOV #64,RO ;LOAD COUNT FOR GAP
3211 012000 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3212 012006 005037 003254 CLR PR.BIT
3213 012012 004737 037030 JSR PC,RDBIT
3214 012016 005300 DEC RO ;CHECK IF GAP FINISHED
3215 012020 001367 BNE 25$ ;NO, CONTINUE
3216 012022 016237 000006 003146 MOV RKDA(R2),T.DA ;GET DISK ADDRESS REG
3217 012030 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3218 012036 023737 003206 003146 CMP E.DA,T.DA ;CHECK DISK ADD CORRECT
3219 012044 001401 BEQ 30$ ;YES, CONTINUE
3220 012046 104056 ERROR 56 ;DISK ADDRESS INCORRECT
3221 012050 023737 003220 003160 30$: CMP E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3222 012056 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3223 012060 104002 ERROR R2 ;CYLINDER INCORRECT
3224 012062 104415 32$: SCOP1 ;CHECK IF LOOP ON ERROR
3225 012064 105237 003277 INCB TRACK ;INCREMENT TRACK
3226 012070 122737 000003 003277 CMPB #3,TRACK ;CHECK IF ALL TRACKS TRIED
3227 012076 001220 BNE 1$ ;NO, TRY NEXT VALUE

```

```

*****
*TEST 20 CYLINDER INCREMENT
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD
* TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2,
* SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
* THAT WRITE GATE SETS.
*
* REPEAT FOR CYLINDER = 1-632.
*****

```

```

3241 *****
3242 012100 000004 TST20: SCOP1
3243 012102 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
3244 012110 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3245 012114 005037 003300 CLR CYLN ;INITIALIZE CYLINDER
3246 012120 012737 001025 003276 MOV #1025,SECTOR ;INITIALIZE TRACK AND SECTOR
3247 012126 012737 012134 001110 MCV #15,$LPERR ;LOAD LOOP ON ERROR LOCATION FOR
3248 ; SUBTEST LOOP
3249
3250 012134 1$: JSR PC,INCHDP ;GENERATE HEADER WORDS
3251 012134 004737 035626 MOV #CLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3252 012140 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3253 012146 012762 000040 000026 MOV CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3254 012154 013762 003300 000020 MOV SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3255 012162 013762 003276 000006 MOV #-1,RKWC(R2) ;WORD COUNT=1
3256 012170 012762 177777 000002 MOV #BUFF,RKEA(R2) ;LOAD DUMMY ADDRESS
3257 012176 012762 055220 000004 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3258 012204 012762 000023 000000 MOV #69,*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL READY
3259 012212 012700 000426

```

N05

```

3260                                     ; FOR SECTOR PULSE
3261 012216 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3262 012224 012762 000040 000026 MOV #DMD,RKMR1(R2)
3263 012232 005300 DEC RO
3264 012234 001370 BNE 5$
3265 012236 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3266 012244 012762 000040 000026 MOV #DMD,RKMR1(R2)
3267 012252 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3268 012256 005037 003256 CLR M1.BIT
3269 012262 012700 000377 MOV #255,RO
3270 012266 004737 037030 10$: JSR PC,RDBIT
3271 012272 005300 DEC RO
3272 012274 001374 BNE 10$
3273 012276 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3274 012304 004737 037030 JSR PC,RDBIT
3275 012310 012703 003302 MOV #HEADER,R3 ;SIMULATE HEADER
3276 012314 012701 000003 MOV #3,R1
3277 012320 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3278 012322 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
3279 012326 013737 003254 003256 15$: MOV PR.BIT,M1.BIT
3280 012334 006004 ROR R4
3281 012336 103403 BCS 17$
3282 012340 005037 003254 CLR PR.BIT
3283 012344 000403 BR 18$
3284
3285 012346 012737 000001 003254 17$: MOV #1,PR.BIT
3286 012354 004737 037030 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3287 012360 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
3288 012362 001361 BNE 15$ ;NO, CONTINUE
3289 012364 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3290 012366 001354 BNE 12$ ;NO, CONTINUE
3291 012370 012700 000100 MOV #64,RO ;LOAD COUNT FOR GAP
3292 012374 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3293 012402 005037 003254 CLR PR.BIT
3294 012406 004737 037030 JSR PC,RDBIT
3295 012412 005300 DEC RO ;CHECK IF GAP FINISHED
3296 012414 001367 BNE 25$ ;NO, CONTINUE
3297 012416 016237 000006 003146 MOV RKDA(R2),T.DA ;GET DISK ADDRESS REG
3298 012424 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3299 012432 023737 003206 003146 CMP E.DA,T.DA ;CHECK DISK ADD CORRECT
3300 012440 001401 BEQ 30$ ;YES, CONTINUE
3301 012442 104060 ERROR 60 ;DISK ADDRESS INCORRECT
3302 012444 023737 003220 003160 30$: CMP E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3303 012452 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3304 012454 104002 ERROR R2 ;CYLINDER INCORRECT
3305 012456 104415 32$: SCOP1 ;CHECK IF LOOP ON ERROR
3306 012460 005237 003300 INC CYLN ;INCREMENT CYLINDER
3307 012464 022737 000633 003300 CMP #633,CYLN ;CHECK IF ALL CYLINDER TRIED
3308 012472 001220 BNE 1$ ;NO, TRY NEXT VALUE

```

```

*****
*TEST 21 BAD SECTOR ERROR (PART 1)
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0.

```

```

3309
3310
3311
3312
3313
3314
3315

```

HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR
MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH
THE FOLLOWING DATA:

000000
040000
040000

MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS
IS NOT INCREMENTED.

3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371

ST21: SCOPE
MOV #10, \$TIMES ; DO 10 ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #0, RKDCYL(R2) ; LOAD CYLINDER
MOV #0, RKDA(R2) ; LOAD TRACK AND SECTOR
MOV #-1, RKWC(R2) ; WORD COUNT=1
MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV #WRDATA, RKCS1(R2) ; ISSUE WRITE DATA
MOV #69, *4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE
15: MOV #DMD, MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 15
MOV #DMD, MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
55: JSR PC, R0BIT
DEC R0 ; CHECK IF SYNCH FINISHED
BNE 55
MOV #1, PR.BIT ; SIMULATE SYNCH BIT
JSR PC, R0BIT
MOV #HEAD2, R3 ; SIMULATE HEADER ERROR
; #3, R1
125: MOV (R3)+, R4 ; GET NEXT HEADER WORD
MOV #16, R0 ; LOAD BITS PER WORD
155: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 175
CLR PR.BIT
BR 185
175: MOV #1, PR.BIT
185: JSR PC, R0BIT ; SIMULATE NEXT BIT
DEC R0 ; CHECK IF READY FOR NEXT HEADER WORD
BNE 155 ; NO, CONTINUE
DEC R1 ; CHECK IF FINISHED WITH HEAD2
BNE 125 ; NO, CONTINUE
MOV #64, R0 ; LOAD COUNT FOR GAP
255: MOV PR.BIT, M1.BIT ; SIMULATE GAP

3372	012752	005037	003254	CLR	PR.BIT	
3373	012756	004737	037030	JSR	PC,ROBIT	
3374	012762	005300		DEC	RO	:CHECK IF GAP FINISHED
3375	012764	001367		BNE	25\$:NO CONTINUE
3376	012766	016237	000000 003140	MOV	RKCS1(R2),T.CS1	:GET CS1
3377	012774	016237	000010 003150	MOV	RKCS2(R2),T.CS2	:GET CS2
3378	013002	016237	000014 003154	MOV	RKER(R2),T.ER	:GET ERROR REG
3379	013010	012737	000023 003200	MOV	#WRDATA,E.CS1	:LOAD EXPECTED CS1
3380	013016	012737	000300 003210	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3381	013024	012737	000200 003214	MOV	#BSE,E.ER	:LOAD EXPECTED ERROR REG.
3382	013032	023737	003200 003140	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
3383	013040	001401		BEQ	30\$:YES, CHECK CS2
3384	013042	104063		ERROR	63	:CS1 INCORRECT
3385	013044	023737	003210 003150 30\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
3386	013052	001401		BEQ	32\$:YES, CHECK ERROR REG.
3387	013054	104064		ERROR	64	:CS2 INCORRECT
3388	013056	023737	003214 003154 32\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3389	013064	001401		BEQ	TST22	:YES, GO ON TO NEXT TEST
3390	013066	104065		ERROR	65	:ERROR REG INCORRECT

:TEST 22 BAD SECTOR ERROR (PART 2)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT CYLINDER C,
* HEAD D, SECTOR D. CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH
* THE FOLLOWING DATA:

000000
100000
100000

* MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS
* IS NOT INCREMENTED.

TST22: SCOPE

3410	013070	000004		MOV	#10,\$TIMES	:DO 10. ITERATIONS
3411	013072	012737	000012 001200	MOV	\$BASE,R2	:LOAD RK611 BASE
3412	013100	013702	001270	MOV	#CLR,RKCS1(R2)	:CLEAR RK611
3413	013104	012762	100000 000000	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3414	013112	012762	000040 000026	MOV	#0,RKDCYL(R2)	:LOAD CYLINDER
3415	013120	012762	000000 000020	MOV	#0,RKDA(R2)	:LOAD TRACK AND SECTOR
3416	013126	012762	000000 000006	MOV	#-1,RKWC(R2)	:WORD COUNT=1
3417	013134	012762	177777 000002	MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
3418	013142	012762	055220 000004	MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
3419	013150	012762	000023 000000	MOV	#69.*4+2,RO	:ISSUE ENOUGH CLOCKS UNTIL READY
3420	013156	012700	000426	MOV		:FOR SECTOR PULSE
3421						
3422	013162	012762	000440 000026 1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3423	013170	012762	000040 000026	MOV	#DMD,RKMR1(R2)	
3424	013176	005300		DEC	RO	
3425	013200	001370		BNE	1\$	
3426	013202	012762	000140 000026	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
3427	013210	012762	000040 000026	MOV	#DMD,RKMR1(R2)	


```

03428 013216 005037 003254 CLR PR.BIT ;GENERATE SYNCH
03429 013222 005037 003256 CLR M1.BIT
03430 013226 012700 000377 MOV #255,RC
03431 013232 004737 037030 55: JSR PC,RDBIT
03432 013236 005300 DEC RO ;CHECK IF SYNCH FINISHED
03433 013240 001374 BNE 55
03434 013242 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
03435 013250 004737 037030 JSR PC,RDBIT
03436 013254 012703 053402 MOV #HEAD3,R3 ;SIMULATE HEADER ERROR
03437 013260 012701 000003 MOV #3,R1
03438 013264 012304 125: MOV (R3)+,R4 ;GET NEXT HEADER WORD
03439 013266 012700 000020 MOV #16,RC ;LOAD BITS PER WORD
03440 013272 013737 003254 003256 155: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
03441 013300 006004 ROR R4 ;GET NEXT BIT
03442 013302 103403 BCS 175
03443 013304 005037 003254 CLR PR.BIT
03444 013310 000403 BR 185
03445
03446 013312 012737 000001 003254 175: MOV #1,PR.BIT
03447 013320 004737 037030 195: JSR PC,RDBIT ;SIMULATE NEXT BIT
03448 013324 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
03449 013326 001361 BNE 155 ;NO, CONTINUE
03450 013330 005301 DEC R1 ;CHECK IF FINISHED WITH HEAD3
03451 013332 001354 BNE 125 ;NO, CONTINUE
03452 013334 012700 000100 MOV #64,RO ;LOAD COUNT FOR GAP
03453 013340 013737 003254 003256 255: MOV PR.BIT,M1.BIT ;SIMULATE GAP
03454 013346 005037 003254 CLR PR.BIT
03455 013352 004737 037030 JSR PC,RDBIT
03456 013356 005300 DEC RO ;CHECK IF GAP FINISHED
03457 013360 001367 BNE 255 ;NO, CONTINUE
03458 013362 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
03459 013370 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
03460 013376 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
03461 013404 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
03462 013412 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
03463 013420 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG.
03464 013426 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
03465 013434 001401 BEQ 305 ;YES, CHECK CS2
03466 013436 104063 ERROR 63 ;CS1 INCORRECT
03467 013440 023737 003210 003150 305: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
03468 013446 001401 BEQ 325 ;YES, CHECK ERROR REG.
03469 013450 104064 ERROR 64 ;CS2 INCORRECT
03470 013452 023737 003214 003154 325: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
03471 013460 001401 BEQ TST23 ;YES, GO ON TO NEXT TEST
03472 013462 104065 ERROR 65 ;ERROR REG INCORRECT

```

```

*****
*TEST 23 OPERATION INCOMPLETE
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253,
* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH
* 1 BIT DIFFERENT IN 30 BITS OF OPT DETERMINATION. ALL
* SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY

```


E06

* OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS
* THAT SET.

```

3484
3485
3486
3487
3488 013464 000004
3489 013466 012737 000012 001200
3490 013474 013702 001270
3491 013500 012762 100000 000000
3492 013506 012700 002500
3493 013512 005300
3494 013514 001376
3495 013516 012762 000040 000026
3496 013524 012762 055220 000004
3497 013532 012762 177777 000002
3498 013540 012762 001253 000020
3499 013546 012762 001023 000006
3500 013554 012762 010023 000000
3501 013562 012700 000426
3502
3503 013566 012762 000440 000026 15:
3504 013574 012762 000040 000026
3505 013602 005300
3506 013604 001370
3507 013606 012705 000040
3508 013612 012703 053576
3509 013616 012737 010023 003200
3510 013624 012737 000300 003210
3511 013632 005037 003214
3512 013636 012737 022040 003224
3513 013644 005037 003310
3514 013650 012762 000140 000026 55:
3515 013656 012762 000040 000026
3516 013664 022737 000037 003310
3517
3518 013672 001460
3519 013674 005037 003254
3520 013700 005037 003256
3521 013704 012700 000377
3522 013710 004737 037030 105:
3523 013714 005300
3524 013716 001374
3525 013720 012737 000001 003254
3526 013726 004737 037030
3527 013732 012701 000003
3528 013736 012304 125:
3529 013740 012700 000020
3530 013744 013737 003254 003256 155:
3531 013752 006004
3532 013754 103403
3533 013756 005037 003254
3534 013762 000403
3535
3536 013764 012737 000001 003254 175:
3537 013772 004737 037030 185:
3538 013776 005300
3539 014000 001361
3540 014002 005301

```

```

+ST23: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;CLEAR RK611
MOV #2500, R0 ;SET COUNT FOR STALL
25: DEC R0 ;DEC COUNT
BNE 25 ;LOOP UNTIL 0
MOV #DMD, RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
MOV #-1, RKWC(R2) ;WORD COUNT = 1
MOV #1253, RKDCYL(R2) ;LOAD CYLINDER
MOV #1023, RKDA(R2) ;LOAD TRACK AND SECTOR.
MOV #WRDATA!CFMT, RKCS1(R2) ;ISSUE WRITE DATA
MOV #69, #4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE
15: MOV #DMD!MCLK, RKMRI(R2)
MOV #DMD, RKMRI(R2)
DEC R0
BNE 15
MOV #32, R5 ;LOAD HEADER COUNT
MOV #OPI1, R3 ;LOAD ADDRESS OF HEADERS
MOV #WRDATA!CFMT, E.CS1 ;LOAD EXPECTED CS1
MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
CLR E.ER ;LOAD EXPECTED ERROR REG
MOV #DMD!MEWD!ECCW, E.MRI ;LOAD EXPECTED MRI
CLR HDRCNT ;INITIALIZE HEADER COUNT
55: MOV #DMD!MSP, RKMRI(R2) ;SIMULATE SECTOR PULSE
MOV #DMD, RKMRI(R2)
CMP #31, HDRCNT ;CHECK IF ALL HEADERS DONE
BEQ 265 ;YES - SKIP TO ERROR TEST
CLR PR.BIT ;GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
105: JSR PC, RDBIT
DEC R0 ;CHECK IF SYNCH FINISHED
BNE 105
MOV #1, PR.BIT ;SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #3, R1 ;SIMULATE OPI
MOV (R3)+, R4 ;GET NEXT HEADER WORD
MOV #16, R0 ;LOAD BITS PER WORD
155: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT
175: BCS 175
CLR PR.BIT
BR 185
185: MOV #1, PR.BIT
JSR PC, RDBIT ;SIMULATE NEXT BIT
DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
BNE 155 ;NO, CONTINUE
DEC R1 ;CHECK IF FINISHED WITH HEADER

```

```

3540 014004 001354          BNE      125          :NO CONTINUE
3541 014006 012700 000100      MOV      #64.,R0     :LOAD CCJNT FOR GAP
3542 014012 013737 003254 0C3256 25S:  MOV      PR.BIT,M1.BIT :SIMULATE GAP
3543 014020 005037 003254      CLR      PR.BIT
3544 014024 004737 037030      JSR      PC,R0BIT
3545 014030 005300          DEC      R0          :CHECK IF GAP IS FINISHED
3546 014032 001367          BNE      25S         :NO CONTINUE
3547 014034 016237 000000 003140 26S:  MOV      RKCS1(R2),T.CS1 :GET CS1
3548 014042 016237 000010 003153      MOV      RKCS2(R2),T.CS2 :GET CS2
3549 014050 016237 000014 003154      MOV      RKER(R2),T.ER   :GET ERROR REG.
3550 014056 023737 003200 003140      CMP      E.CS1,T.CS1     :CHECK CS1 CORRECT
3551 014064 001401          BEQ      30S         :YES, CONTINUE
3552 014066 104074          ERROR    74          :CS1 INCORRECT
3553 014070 023737 003210 003150 30S:  CMP      E.CS2,T.CS2     :CHECK CS2 CORRECT
3554 014076 001401          BEQ      31S         :YES, CONTINUE
3555 014100 104075          ERROR    75          :CS2 INCORRECT
3556 014102 023737 003214 003154 31S:  CMP      E.ER,T.ER       :CHECK ERROR REG CORRECT
3557 014110 001401          BEQ      32S         :YES, CONTINUE
3558 014112 104076          ERROR    76          :ERROR REG INCORRECT
3559 014114 016237 000026 003164 32S:  MOV      RKMRI(R2),T.MRI :GET MRI
3560 014122 023737 003224 0C3164      CMP      E.MRI,T.MRI     :CHECK TO MAKE SURE WRITE GATE DID NOT SET
3561 014130 001401          BEQ      34S         :YES, CONTINUE
3562 014132 104077          ERROR    77          :MRI INCORRECT
3563 014134 005237 003310          INC      HRCNT        :INCREMENT HEADER COUNT
3564 014140 022737 000037 003310 34S:  CMP      #31.,HRCNT     :CHECK IF LAST HEADER
3565 014146 001011          BNE      35S         :NO, CHECK IF FINISHED
3566 014150 012737 020000 003214      MOV      #0PI,E.ER      :LOAD ERROR BIT
3567 014156 042737 000001 003200      BIC      #00,E.CS1      :ADJUST E.CS1 FOR END OF OP CONTENTS
3568 014164 052737 100200 003200      BIS      #RDY!CERR,E.CS1
3569 014172 005305          DEC      RS           :CHECK IF FINISHED
3570 014174 0C1402          BEQ      37S         :YES - SKIP
3571 014176 000137 013650          JMP      5S           :DO NEXT SECTOR
3572 014202 012762 100000 000000 37S:  MOV      #CLR,RKCS1(R2) :CLEAR CONTROLLER
3573 014210 016237 000000 0C3140      MOV      RKCS1(R2),T.CS1 :GET CS1
3574 014216 016237 000010 003150      MOV      RKCS2(R2),T.CS2 :CS2
3575 014224 016237 000014 003154      MOV      RKER(R2),T.ER   :ER
3576 014232 012737 00020C 003200      MOV      #RDY,E.CS1     :SET EXPECTED CS1
3577 014240 023737 003140 003200      CMP      T.CS1,E.CS1     :CHECK IF CORRECT
3578 014246 001401          BEQ      TST24        :GO TO NEXT TEST
3579 014250 104153          ERROR    153

```

```

*****
*TEST 24      HEADER VRC
*

```

```

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1253,
* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES SIMULATE A SECTOR PULSE AND HEADER WITH BIT 0
* OF THE VRC INCORRECT. MAKE SURE ONLY HEADER VRC AND
* CONTROLLER ERROR ARE THE ONLY ERRORS SET. REPEAT FOR
* BITS 1-15 OF VRC.
*

```

```

*****

```

```

3594 014252 000004      TST24:  SCOPE
3595 014254 012737 000C12 001200      MOV      #10.,$TIMES   ;;DO 10. ITERATIONS

```

G06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MC-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 71
 DZR6DA.P11 28-SEP-76 15:50 T24 HEADER VRC

SEG 007:

3596	014262	013702	001270		MOV	\$BASE,R2	:LOAD RK611 BASE
3597	014266	012737	012023	003200	MOV	#CFMT!CDT!WRDATA,E.CS1	:LOAD EXPECTED CS1
3598	014274	012737	000300	003210	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3599	014302	012737	000400	003214	MOV	#HVRC,E.ER	:LOAD EXPECTED ERROR REGISTER
3600	014310	012737	000001	001170	MOV	#1,STMP4	:INITIALIZE BIT FOR VRC ERROR
3601	014316	012737	014324	001110	MOV	#1\$,SLPERR	:LOAD LOOP ON ERROR LOCATION FOR
3602							: SUBTEST LOOP
3603							
3604	014324				15:		
3605	014324	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3606	014332	012700	002000		MOV	#2000,R0	:SET COUNT FOR STALL
3607	014336	005300			25:	DEC	R0
3608	014340	001376			BNE	25	:LOOP UNTIL 0
3609	014342	012762	000040	000026	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN MAINTENANCE MODE
3610	014350	012737	140370	055202	MOV	#140370,VRCHDR+4	:LOAD VRC
3611	014356	033737	001170	055202	BIT	STMP4,VRCHDR+4	:MAKE ONE BIT BAD
3612	014364	001404			BEQ	35	
3613	014366	043737	001170	055202	BIC	STMP4,VRCHDR+4	
3614	014374	000403			BR	55	
3615							
3616	014376	053737	001170	055202	35:	BIS	STMP4,VRCHDR+4
3617	014404	012762	055220	000004	55:	MOV	#BUFF,RKBA(R2)
3618	014412	012762	177777	000002	MOV	#-1,RKWC(R2)	:WORD COUNT = 1
3619	014420	012762	001023	000006	MOV	#1023,RKDA(R2)	:LOAD TRACK 2 SECTOR 23
3620	014426	012762	001253	000020	MOV	#1253,RKDCYL(R2)	:LOAD CYLINDER 1253
3621	014434	012762	012023	000000	MOV	#CFMT!CDT!WRDATA,RKCS1(R2)	:ISSUE COMMAND
3622	014442	012700	000426		MOV	#69,*4+2,R0	:LOAD COUNT UNTIL READY FOR SECTOR
3623	014446	012762	000440	000026	105:	MOV	#DMD!MCLK,RKMR1(R2)
3624	014454	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3625	014462	005300			DEC	R0	
3626	014464	001370			BNE	105	
3627	014466	012762	000140	000026	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
3628	014474	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3629	014502	005037	003254		CLR	PR.BIT	:INITIAL BITS
3630	014506	005037	003256		CLR	M1.BIT	
3631	014512	012700	000377		MOV	#255,R0	:SIMULATE SYNCH
3632	014516	004737	037030		155:	JSR	PC,R0BIT
3633	014522	005300			DEC	R0	
3634	014524	001374			BNE	155	
3635	014526	012737	000001	003254	MOV	#1,PR.BIT	:GENERATE SYNCH BIT
3636	014534	004737	037030		JSR	PC,R0BIT	:SIMULSTE SYNC 1 BIT
3637	014540	012703	055176		MOV	#VRCHDR,R3	:LOAD ADDRESS OF HEADER
3638	014544	012701	000003		MOV	#3,R1	:LOAD HEADER COUNT
3639	014550	012304			175:	MOV	(R3)+,R4
3640	014552	012700	000020		MOV	#16,R0	:LOAD BITS PER WORD
3641	014556	013737	003254	003256	205:	MOV	PR.BIT,M1.BIT
3642	014564	006004			ROR	R4	
3643	014566	103403			BCS	235	
3644	014570	005037	003254		CLR	PR.BIT	
3645	014574	000403			BR	255	
3646							
3647	014576	012737	000001	003254	235:	MOV	#1,PR.BIT
3648	014604	004737	037030		255:	JSR	PC,R0BIT
3649	014610	005300			DEC	R0	:CHECK IF FINISHED WITH WORD
3650	014612	001361			BNE	205	:NO, CONTINUE
3651	014614	005301			DEC	R1	:CHECK IF HEADER FINISHED

H06

```

3652 014616 001354 BNE 17$ ;NO, CONTINUE
3653 014620 012700 000100 MOV #64, R0 ;SIMULATE GAP
3654 014624 013737 003254 003256 30$: MOV PR.BIT, M1.BIT
3655 014632 005037 003254 CLR PR.BIT
3656 014636 004737 037030 JSR PC, RDBIT
3657 014642 005300 DEC R0
3658 014644 001367 BNE 30$
3659 014646 016237 000000 003140 MOV RKCS1(R2), T.CS1 ;STORE CS1
3660 014654 016237 000010 003150 MOV RKCS2(R2), T.CS2 ;STORE CS2
3661 014662 016237 000014 003154 MOV RKR(R2), T.ER ;STORE ERROR REG
3662 014670 023737 003200 003140 CMP E.CS1, T.CS1 ;CHECK CS1 CORRECT
3663 014676 001401 BEQ 35$ ;YES, CHECK CS2
3664 014700 104071 ERROR 71 ;CS1 INCORRECT
3665 014702 023737 003210 003150 35$: CMP E.CS2, T.CS2 ;CHECK CS2 CORRECT
3666 014710 001401 BEQ 37$ ;YES, CHECK ERROR REG
3667 014712 104072 ERROR 72 ;CS2 INCORRECT
3668 014714 023737 003214 003154 37$: CMP E.ER, T.ER ;CHECK ERROR REG CORRECT
3669 014722 001401 BEQ 40$ ;YES, CHECK IF LOOP ON ERROR
3670 014724 104073 ERROR 73 ;ERROR REG INCORRECT
3671 014726 104415 SCOP1 ;CHECK IF LOOP ON ERROR
3672 014730 006337 001170 ASL $TMP4 ;GET NEXT PATTERN
3673 014734 103402 BCS TST25 ;CHECK IF FINISHED
3674 014736 000137 014324 JMP 1$ ;NO, CONTINUE

```

```

*****
*TEST 25 BAD SECTOR ERROR AND HEADER VRC

```

```

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300.
* HEAD 1, SECTOR 17, CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES. SIMULATE THE FOLLOWING HEADER:

```

```

*
* 000300
* 040057
* 040356

```

```

* MAKE SURE ONLY HEADER VRC ERROR SETS.

```

```

*****
TST25: SCOP1

```

```

3692 014742 000004 MOV #10, $TIMES ;DO 10. ITERATIONS
3693 014744 012737 000012 001200 MOV $BASE, R2 ;LOAD RK611 BASE
3694 014752 013702 001270 MOV #CCLR, RKCS1(R2) ;CLEAR RK611
3695 014756 012762 100000 000000 MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3696 014764 012762 000040 000026 MOV #300, RKDCYL(R2) ;LOAD CYLINDER
3697 014772 012762 000300 000020 MOV #417, RKDA(R2) ;LOAD TRACK AND SECTOR
3698 015000 012762 000417 000006 MOV #-1, RKWC(R2) ;WORD COUNT=1
3699 015006 012762 177777 000002 MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3700 015014 012762 055220 000004 MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
3701 015022 012762 000023 000000 MOV #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3702 015030 012700 000426 ; FOR SECTOR PULSE
3703
3704 015034 012762 000440 000026 1$: MOV #DMD!MCLK, RKMR1(R2)
3705 015042 012762 000040 000026 MOV #DMD, RKMR1(R2)
3706 015050 005300 DEC R0
3707 015052 001370 BNE 1$

```

```

3708 015054 012762 000140 000026      MOV      #DMD!MSP,RKMRI(R2) ;SIMULATE SECTOR PULSE
3709 015062 012762 000040 000026      MOV      #DMD,RKMRI(R2)
3710 015070 005037 003254      CLR      PR.BIT ;GENERATE SYNCH
3711 015074 005037 003256      CLR      M1.BIT
3712 015100 012700 000377      MOV      #255,R0
3713 015104 004737 037030      55:     JSR      PC,RDBIT
3714 015110 005300      DEC      R0 ;CHECK IF SYNCH FINISHED
3715 015112 001374      BNE      55
3716 015114 012737 000001 003254      MOV      #1,PR.BIT ;SIMULATE SYNCH BIT
3717 015122 004737 037030      JSR      PC,RDBIT
3718 015126 012703 053410      MOV      #HEAD4,R3 ;SIMULATE HEADER ERROR
3719 015132 012701 000003      MOV      #3,R1
3720 015136 012304      125:    MOV      (R3)+,R4 ;GET NEXT HEADER WORD
3721 015140 012700 000020      MOV      #16,R0 ;LOAD BITS PER WORD
3722 015144 013737 003254 003256 155:    MOV      PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3723 015152 006004      ROR      R4 ;GET NEXT BIT
3724 015154 103403      BCS      175
3725 015156 005037 003254      CLR      PR.BIT
3726 015162 000403      BR       185
3727
3728 015164 012737 000001 003254 175:    MOV      #1,PR.BIT
3729 015172 004737 037030 185:    JSR      PC,RDBIT ;SIMULATE NEXT BIT
3730 015176 005300      DEC      R0 ;CHECK IF READY FOR NEXT HEADER WORD
3731 015200 001361      BNE      155 ;NO, CONTINUE
3732 015202 005301      DEC      R1 ;CHECK IF FINISHED WITH HEAD4
3733 015204 001354      BNE      125 ;NO, CONTINUE
3734 015206 012700 000100      MOV      #64,R0 ;LOAD COUNT FOR GAP
3735 015212 013737 003254 003256 255:    MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3736 015220 005037 003254      CLR      PR.BIT
3737 015224 004737 037030      JSR      PC,RDBIT
3738 015230 005300      DEC      R0 ;CHECK IF GAP FINISHED
3739 015232 001367      BNE      255 ;NO, CONTINUE
3740 015234 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
3741 015242 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
3742 015250 016237 000014 003154      MOV      RKR(R2),T.ER ;GET ERROR REG
3743 015256 012737 000023 003200      MOV      #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3744 015264 012737 000300 003210      MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3745 015272 012737 000400 003214      MOV      #HVRC,E.ER ;LOAD EXPECTED ERROR REG.
3746 015300 023737 003200 003140      CMP      E.CS1,T.CS1 ;CHECK CS1 CORRECT
3747 015306 001401      BEQ      305 ;YES, CHECK CS2
3748 015310 104066      ERROR    66 ;CS1 INCORRECT
3749 015312 023737 003210 003150 305:    CMP      E.CS2,T.CS2 ;CHECK CS2 CORRECT
3750 015320 001401      BEQ      325 ;YES, CHECK ERROR REG.
3751 015322 104067      ERROR    67 ;CS2 INCORRECT
3752 015324 023737 003214 003154 325:    CMP      E.ER,T.ER ;CHECK ERROR REG CORRECT
3753 015332 001401      BEQ      TST26 ;YES, GO ON TO NEXT TEST
3754 015334 104070      ERROR    70 ;ERROR REG INCORRECT

```

```

*****
*TEST 26 GOOD HEADER AND PREVIOUS BSE
*
* CLEAR RK611 CONTROLLER WITH CONTROLLER CLEAR. PUT
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 100,
* HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
* MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE

```

```

3755
3756
3757
3758
3759
3760
3761
3762
3763

```

```

3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782 015336 000004
3783 015340 012737 000012 001200
3784 015346 013702 001270
3785 015352 012762 100000 000000
3786 015360 012762 000040 000026
3787 015366 012762 055220 000004
3788 015374 012762 177777 000002
3789 015402 012762 000001 000006
3790 015410 012762 000100 000023
3791 015416 012762 000023 000000
3792 015424 012700 000426
3793
3794 015430 012762 000440 000026 1$:
3795 015436 012762 000040 000026
3796 015444 005300
3797 015446 001370
3798 015450 012705 000002
3799 015454 012703 053416
3800 015460 012737 000023 003200
3801 015466 012737 000300 003210
3802 015474 005037 003214
3803 015500 012737 022040 003224
3804 015506 005037 003310
3805 015512 012762 000140 000026 5$:
3806 015520 012762 000040 000026
3807 015526 005037 003254
3808 015532 005037 003256
3809 015536 012700 000377
3810 015542 004737 037030 10$:
3811 015546 005300
3812 015550 001374
3813 015552 012737 000001 003254
3814 015560 004737 037030
3815 015564 012701 000003
3816 015570 012304 12$:
3817 015572 012700 000020
3818 015576 013737 003254 15$:
3819 015604 006004

```

```

: * FOLLOWING 3 WORDS WITH A BAD SECTOR INDICATION:
: *
: * 000100
: * 040000
: * 040100
: *
: * MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
: * SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING
: * 3 WORDS:
: *
: * 000100
: * 140001
: * 140101
: *
: * MAKE SURE WRITE GATE SETS INDICATING THAT HEADER HAS
: * BEEN RECOGNIZED.

```

```

*****
TST26: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;CLEAR RK611
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #-1, RKWC(R2) ;WORD COUNT -1
MOV #1, RKDA(R2) ;LOAD TRACK AND SECTOR
MOV #100, RKDCYL(R2) ;LOAD CYLINDER
MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
MOV #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE
MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV #2, R5 ;LOAD HEADER COUNT
MOV #HEADS, R3 ;LOAD ADDRESS OF HEADERS
MOV #WRDATA, E.CS1 ;LOAD EXPECTED CS1
MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
CLR E.ER ;LOAD EXPECTED MR1
MOV #DMD!MEWD!ECC.W.E, MR1 ;LOAD EXPECTED MR1
CLR HDRCNT ;INITIALIZE HEADER COUNT
MOV #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ;GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
JSR PC, RDBIT
DEC R0 ;CHECK IF SYNCH FINISHED
BNE 10$
MOV #1, PR.BIT ;SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #3, R1 ;SIMULATE HEADER
MOV (R3)+, R4 ;GET NEXT HEADER WORD
MOV #16, R0 ;LOAD BITS PER WORD
MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT

```

K06

```

3820 015606 10340C          BCS      17$
3821 015610 005037 003254 CLR      PR.BIT
3822 015614 000403          BR       18$
3823
3824 015616 012737 000001 003254 17$: MOV     #1,PR.BIT
3825 015624 004737 037030 18$: JSR   PC,RDBIT      ;SIMULATE NEXT BIT
3826 015630 005300          DEC     RO              ;CHECK IF READY FOR NEXT HEADER WORD
3827 015632 001361          SNE    15$             ;NO, CONTINUE
3828 015634 005301          DEC     R1              ;CHECK IF FINISHED WITH HEADER
3829 015636 001354          BNE    12$             ;NO, CONTINUE
3830 015640 012700 000102 MOV     #66.,RO        ;LOAD COUNT FOR GAP
3831                                     ;PLUS 2 COUNTS FOR WRTGAT TO SET
3832 015644 013737 003254 003256 25$: MOV     PR.BIT,M1.BIT ;SIMULATE GAP
3833 015652 005037 003254 CLR      PR.BIT
3834 015656 004737 037030 JSR   PC,RDBIT
3835 015662 005300          DEC     RO              ;CHECK IF GAP IS FINISHED
3836 015664 001367          BNE    25$             ;NO, CONTINUE
3837 015666 016237 000000 003140 MOV     RKCS1(R2),T.CS1 ;GET CS1
3838 015674 016237 000010 003150 MOV     RKCS2(R2),T.CS2 ;GET CS2
3839 015702 016237 000014 003154 MOV     RKER(R2),T.ER   ;GET ERROR REG
3840 015710 023737 003200 003140 CMP     E.CS1,T.CS1    ;CHECK CS1 CORRECT
3841 015716 001401          BEQ    30$             ;YES, CONTINUE
3842 015720 104114          ERROR  114            ;CS1 INCORRECT
3843 015722 023737 003210 003.   30$: CMP     E.CS2,T.CS2    ;CHECK CS2 CORRECT
3844 015730 001401          BEQ    31$             ;YES, CONTINUE
3845 015732 104115          ERROR  115            ;CS2 INCORRECT
3846 015734 023737 003214 003154 31$: CMP     E.ER,T.ER    ;CHECK ERROR REG CORRECT
3847 015742 001401          BEQ    32$             ;YES, CONTINUE
3848 015744 104116          ERROR  116            ;ERROR REG INCORRECT
3849 015746 016237 000026 003164 32$: MOV     RKMRI(R2),T.MRI ;GET MRI
3850 015754 023737 003224 003164 CMP     E.MRI,T.MRI    ;CHECK WRITE GATE CORRECT
3851 015762 001401          BEQ    34$             ;YES, CONTINUE
3852 015764 104117          ERROR  117            ;MRI INCORRECT
3853 015766 005237 003310 34$: INC     HDRCNT        ;INCREMENT HEADER COUNT
3854 015772 012737 062040 003224 MOV     #WRTGAT!DMD!MEWD ;ECCW,E.MRI ;LOAD EXPECTED MRI
3855 016000 005305          DEC     R5              ;CHECK IF FINISHED
3856 016002 001402          BEQ    TST27           ;YES, GO ON TO NEXT TEST
3857 016004 000137 015512 JMP     5$

```

```

3858
3859 *****
3860 :*TEST 27      GOOD HEADER AND PREVIOUS HVRC
3861 :*
3862 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3863 :* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF
3864 :* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 200,
3865 :* HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
3866 :* MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE
3867 :* FOLLOWING 3 WORDS WITH A BAD HEADER VRC:
3868 :*
3869 :*          000200
3870 :*          140000
3871 :*          140000
3872 :*
3873 :* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
3874 :* SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE
3875 :* FOLLOWING 3 WORDS:

```


L06

```

3876
3877
3878
3879
3880
3881
3882
3883
3884
3885 016010 000004
3886 016012 012737 000012 001200
3887 016020 013702 001270
3888 016024 012762 100000 000000
3889 016032 012762 000040 000026
3890 016040 012762 055220 000004
3891 016046 012762 177777 000002
3892 016054 012762 000001 000006
3893 016062 012762 000200 000020
3894 016070 012762 000023 000000
3895 016076 012700 000426
3896
3897 016102 012762 000440 000026 15: MOV #DMD!MCLK,RKMR1(R2)
3898 016110 012762 000040 000026 MOV #DMD,RKMR1(R2)
3899 016116 005300 DEC RO
3900 016120 001370 BNE 15
3901 016122 012705 000002 MOV #2,R5 ;LOAD HEADER COUNT
3902 016126 012703 053432 MOV #HEAD6,R3 ;LOAD ADDRESS OF HEADERS
3903 016132 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3904 016140 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3905 016146 005037 003214 CLR E.ER ;LOAD EXPECTED MRI
3906 016152 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MRI
3907 016160 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT
3908 016164 012762 000140 000026 55: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3909 016172 012762 000040 000026 MOV #DMD,RKMR1(R2)
3910 016200 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3911 016204 005037 003256 CLR M1.BIT
3912 016210 012700 000377 MOV #255,RO
3913 016214 004737 037030 105: JSR PC,RDBIT
3914 016220 005300 DEC RO ;CHECK IF SYNCH FINISHED
3915 016222 001374 BNE 105
3916 016224 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3917 016232 004737 037030 JSR PC,RDBIT
3918 016236 012701 000003 MOV #3,R1 ;SIMULATE HEADER
3919 016242 012304 125: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3920 016244 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
3921 016250 013737 003254 155: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3922 016256 006004 ROR R4 ;GET NEXT BIT
3923 016260 103403 BCS 175
3924 016262 005037 003254 CLR PR.BIT
3925 016266 000403 BR 185
3926
3927 016270 012737 000001 003254 175: MOV #1,PR.BIT
3928 016276 004737 037030 185: JSR PC,RDBIT ;SIMULATE NEXT BIT
3929 016302 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
3930 016304 001361 BNE 155 ;NO, CONTINUE
3931 016306 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
    
```

MAKE SURE WRITE GATE SEES INDICATING THAT HEADER HAS BEEN RECOGNIZED.

†ST27: SCOPE

M06

```

3932 016310 001354          BNE 12$          :NO, CONTINUE
3933 016312 012700 000102  MOV #66.,R0      :LOAD COUNT FOR GAP
3934                                     : PLUS 2 COUNTS FOR WRTGAT TO SET
3935 016316 013737 003254 003256 25$: MOV PR.BIT,M1.BIT :SIMULATE GAP
3936 016324 005037 003254          CLR PR.BIT
3937 016330 004737 037030          JSR PC,RDBIT
3938 016334 005300          DEC R0          :CHECK IF GAP IS FINISHED
3939 016336 001367          BNE 25$        :NO, CONTINUE
3940 016340 016237 000000 003140  MOV RKCS1(R2),T.CS1 :GET CS1
3941 016346 016237 000010 003150  MOV RKCS2(R2),T.CS2 :GET CS2
3942 016354 016237 000014 003154  MOV RKER(R2),T.ER   :GET ERROR REG
3943 016362 023737 003200 003140  CMP E.CS1,T.CS1    :CHECK CS1 CORRECT
3944 016370 001401          BEQ 30$        :YES, CONTINUE
3945 016372 104120          ERROR 120       :CS1 INCORRECT
3946 016374 023737 003210 003150 30$: CMP E.CS2,T.CS2    :CHECK CS2 CORRECT
3947 016402 001401          BEQ 31$        :YES, CONTINUE
3948 016404 104121          ERROR 121       :CS2 INCORRECT
3949 016406 023737 003214 003154 31$: CMP E.ER,T.ER     :CHECK ERROR REG CORRECT
3950 016414 001401          BEQ 32$        :YES, CONTINUE
3951 016416 104122          ERROR 122       :ERROR REG INCORRECT
3952 016420 016237 000026 003164 32$: MOV RKMR1(R2),T.MR1 :GET MR1
3953 016426 023737 003224 003164  CMP E.MR1,T.MR1    :CHECK WRITE GATE CORRECT
3954 016434 001401          BEQ 34$        :YES, CONTINUE
3955 016436 104123          ERROR 123       :MR1 INCORRECT
3956 016440 005237 003310 34$: INC HDRCNT      :INCREMENT HEADER COUNT
3957 016444 012737 062040 003224  MOV #WRTGAT!DMD!MEWD :ECCW,E.MR1 :LOAD EXPECTED MR1
3958 016452 005305          DEC R5          :CHECK IF FINISHED
3959 016454 001402          BEQ TST30      :;YES, GO ON TO NEXT TEST
3960 016456 000137 016164          JMP 5$

```

```

3961
3962 ;*****
3963 ;TEST 30      BAD SECTOR ERROR AND PREVIOUS HVRC
3964 ;
3965 ; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3966 ; PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A
3967 ; WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR
3968 ; FORMAT, CYLINDER 400, HEAD 0, SECTOR 1.  CLOCK THROUGH
3969 ; SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
3970 ; AND A HEADER OF THE FOLLOWING 3 WORDS WITH A
3971 ; BAD HEADER VRC:
3972 ;
3973 ;          000400
3974 ;          140000
3975 ;          140000
3976 ;
3977 ; MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
3978 ; DOES NOT SET.  SIMULATE A SECTOR PULSE AND A
3979 ; HEADER CONSISTING OF THE FOLLOWING 3 WORDS:
3980 ;
3981 ;          000400
3982 ;          040001
3983 ;          040401
3984 ;
3985 ; MAKE SURE BAD SECTOR ERROR SETS AND HEADER VRC
3986 ; ERROR DOES NOT SET.
3987 ;

```

N06

```

3988
3989 016462 000004
3990 016464 012737 000012 001200
3991 016472 013702 001270
3992 016476 012762 100000 000000
3993 016504 012762 000040 000026
3994 016512 012762 055220 000004
3995 016520 012762 177777 000002
3996 016526 012762 000001 000006
3997 016534 012762 000400 000020
3998 016542 012762 000023 000000
3999 016550 012700 000426
4000
4001 016554 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4002 016562 012762 000040 000026 MOV #DMD,RKMR1(R2)
4003 016570 005300 DEC R0
4004 016572 001370 BNE 1$
4005 016574 012705 000002 MOV #2,R5 ;LOAD HEADER COUNT
4006 016600 012703 053446 MOV #HEAD7,R3 ;LOAD ADDRESS OF HEADERS
4007 016604 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
4008 016612 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4009 016620 005037 003214 CLR E.ER ;LOAD EXPECTED MR1
4010 016624 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
4011 016632 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT
4012 016636 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4013 016644 012762 000040 000026 MOV #DMD,RKMR1(R2)
4014 016652 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4015 016656 005037 003256 CLR M1.BIT
4016 016662 012700 000377 MOV #255,R0
4017 016666 004737 037030 10$: JSR PC,RDBIT
4018 016672 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4019 016674 001374 BNE 10$
4020 016676 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4021 016704 004737 037030 JSR PC,RDBIT
4022 016710 012701 000003 MOV #3,R1 ;SIMULATE HEADER
4023 016714 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4024 016716 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4025 016722 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4026 016730 006004 ROR R4 ;GET NEXT BIT
4027 016732 103403 BCS 17$
4028 016734 005037 003254 CLR PR.BIT
4029 016740 000403 BR 18$
4030
4031 016742 012737 000001 003254 17$: MOV #1,PR.BIT
4032 016750 004737 037030 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4033 016754 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
4034 016756 001361 BNE 15$ ;NO, CONTINUE
4035 016760 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4036 016762 001354 BNE 12$ ;NO, CONTINUE
4037 016764 012700 000102 MOV #66,R0 ;LOAD COUNT FOR GAP
4038 ; PLUS 2 COUNTS FOR WRTGAT TO SET
4039 016770 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4040 016776 005037 003254 CLR PR.BIT
4041 017002 004737 037030 JSR PC,RDBIT
4042 017006 005300 DEC R0 ;CHECK IF GAP IS FINISHED
4043 017010 001367 BNE 25$ ;NO, CONTINUE
    
```

4074	017012	016237	000000	003140	MOV	RKCS1(R2),T.CS1	:GET CS1
4075	017020	016237	000010	003150	MOV	RKCS2(R2),T.CS2	:GET CS2
4076	017026	016237	000014	003154	MOV	RHER(R2),ER	:GET ERROR REG
4077	017034	023737	003200	003140	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4078	017042	001401			BEQ	308	:YES, CONTINUE
4079	017044	104124			ERROR	124	:CS1 INCORRECT
4080	017046	023737	003210	003150	308:	CMP	E.CS2,T.CS2
4081	017054	001401			BEQ	318	:CHECK CS2 CORRECT
4082	017056	124125			ERROR	125	:YES, CONTINUE
4083	017058	023737	003214	003154	318:	CMP	E.ER,T.ER
4084	017066	001401			BEQ	328	:CHECK ERROR REG CORRECT
4085	017070	104126			ERROR	126	:YES, CONTINUE
4086	017072	016237	000026	003164	328:	MOV	RKMR1(R2),T.MR1
4087	017100	023737	003224	003164	CMP	E.MR1,T.MR1	:GET MR1
4088	017106	001401			BEQ	348	:CHECK WRITE GATE CORRECT
4089	017110	104127			ERROR	127	:YES, CONTINUE
4090	017112	005237	003310		348:	INC	HDRCNT
4091	017116	012737	000200	003214	MOV	#BSE,E.ER	:INCREMENT HEADER COUNT
4092	017124	005305			DEC	R5	:LOAD EXPECTER ERROR REG
4093	017126	001402			BEQ	TST31	:CHECK IF FINISHED
4094	017130	000137	016636		JMP	58	::YES, GO ON TO NEXT TEST

*TEST 31 HEADER VRC AND PREVIOUS BSE

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A
* WRITE DATA OR ONE WORD TO AN RK06 IN 26 SECTOR
* FORMAT, CYLINDER 140, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A HEADER OF THE FOLLOWING 3 WORDS WITH A HEADER
* VRC ERROR:

* 000140
* 040000
* 040140

* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND A
* HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

* 000140
* 140001
* 140101

* MAKE SURE HEADER VRC ERROR SETS AND BAD SECTOR
* ERROR DOES NOT SET.

*TST31: SCOPE

4095	017134	000004			MOV	#10,\$TIMES	::DO 10. ITERATIONS
4096	017136	012737	000012	001200	MOV	\$BASE,R2	:LOAD RK611 BASE
4097	017144	013702	001270		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4098	017150	012762	100000	000000	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4099	017156	012762	000040	000026	MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
4100	017164	012762	055220	000004	MOV	#-1,RKWC(R2)	:WORD COUNT -1
4101	017172	012762	177777	000002			

C07

4100	017200	012762	000001	000006		MOV	#1,RKDA(R2)	:LOAD TRACK AND SECTOR
4101	017206	012762	000140	000320		MOV	#140,RKDCYL(R2)	:LOAD CYLINDER
4102	017214	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4103	017222	012700	000426			MOV	#69,*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL READ / FOR SECTOR PULSE
4104	017226	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	
4105	017234	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4106	017242	005300				DEC	R0	
4107	017244	001370				BNE	15	
4108	017246	012705	000002			MOV	#2,R5	:LOAD HEADER COUNT
4109	017252	012703	053462			MOV	#HEADR,R3	:LOAD ADDRESS OF HEADERS
4110	017256	012737	000023	003200		MOV	#WRDATA,E,CS1	:LOAD EXPECTED CS1
4111	017264	012737	000300	003210		MOV	#R!OR,E,CS2	:LOAD EXPECTED CS2
4112	017272	005037	003214			CLR	E,ER	:LOAD EXPECTED MRI
4113	017276	012737	022040	003224		MOV	#DMD!MEND!ECCW,E,MRI	:LOAD EXPECTED MRI
4114	017304	005037	003310			CLR	HDRCNT	:INITIALIZE HEADER COUNT
4115	017310	012762	000140	000026	55:	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
4116	017316	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4117	017324	005037	003254			CLR	PR,BIT	:GENERATE SYNCH
4118	017330	005037	003256			CLR	M1,BIT	
4119	017334	012700	000277			MOV	#255,R0	
4120	017340	004737	037030		105:	JSR	PC,RDBIT	
4121	017344	005300				DEC	R0	:CHECK IF SYNCH FINISHED
4122	017346	001374				BNE	105	
4123	017350	012737	000001	003254		MOV	#1,PR,BIT	:SIMULATE SYNCH BIT
4124	017356	004737	037030			JSR	PC,RDBIT	
4125	017362	012701	000003			MOV	#3,R1	:SIMULATE HEADER
4126	017366	012304			125:	MOV	(R3)+,R4	:GET NEXT HEADER WORD
4127	017370	012700	000020			MOV	#16,R0	:LOAD BI 5 PER WORD
4128	017374	013737	003254	003256	155:	MOV	PR,BIT,M1,BIT	:STORE PREVIOUS BIT
4129	017402	006004				ROR	R4	:GET NEXT BIT
4130	017404	103403				BCS	175	
4131	017406	005037	003254			CLR	PR,BIT	
4132	017412	000403				BR	185	
4133	017414	012737	000001	003254	175:	MOV	#1,PR,BIT	
4134	017422	004737	037030		185:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
4135	017426	005300				DEC	R0	:CHECK IF READY FOR NEXT HEADER WORD
4136	017430	001361				BNE	155	:NO, CONTINUE
4137	017432	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER
4138	017434	001354				BNE	125	:NO, CONTINUE
4139	017436	012700	000102			MOV	#66.,R0	:LOAD COUNT FOR GAP PLUS 2 COUNTS FOR WRTGAT TO SET
4140	017442	013737	003254	003256	255:	MOV	PR,BIT,M1,BIT	:SIMULATE GAP
4141	017450	005037	003254			CLR	PR,BIT	
4142	017454	004737	037030			JSR	PC,RDBIT	
4143	017460	005300				DEC	R0	:CHECK IF GAP IS FINISHED
4144	017462	001367				BNE	255	:NO, CONTINUE
4145	017464	016237	000000	003140		MOV	RKCS1(R2),T,CS1	:GET CS1
4146	017472	016237	000010	003150		MOV	RKCS2(R2),T,CS2	:GET CS2
4147	017500	016237	000014	003154		MOV	RKER(R2),T,ER	:GET ERRCR REG
4148	017506	023737	003200	003140		CMP	E,CS1,T,CS1	:CHECK CS1 CORRECT
4149	017514	001401				BEQ	305	:YES, CONTINUE
4150	017516	104130				ERROR	130	:CS1 INCORRECT
4151	017520	023737	003210	003150	305:	CMP	E,CS2,T,CS2	:CHECK CS2 CORRECT
4152	017526	001401				BEQ	315	:YES, CONTINUE

4154	017530	104131				ERROR	131	:CS2 INCORRECT
4155	017532	023737	003214	003154	31S:	CMP	E.E.R.T.ER	:CHECK ERROR REG CORRECT
4156	017540	001401				BEQ	32S	:YES, CONTINUE
4157	017542	104132				ERROR	132	:ERROR REG INCORRECT
4158	017544	016237	000026	003164	32S:	MOV	RKMR1(R2),T.MR1	:GET MR1
4159	017552	023737	003224	003164		CMP	E.MR1,T.MR1	:CHECK WRITE GATE CORRECT
4160	017560	001401				BEQ	34S	:YES, CONTINUE
4161	017562	104133				ERROR	133	:MR1 INCORRECT
4162	017564	005237	003310		34S:	INC	HDRCNT	:INCREMENT HEADER COUNT
4163	017570	012737	000400	003214		MOV	#HVRC,E.ER	:LOAD EXPECTER ERROR REG
4164	017576	005305				DEC	R5	:CHECK IF FINISHED
4165	017600	001402				BEQ	TST32	:YES, GO ON TO NEXT TEST
4166	017602	000137	017310			JMP	5S	

*TEST 32 OPI AND HVRC ON LAST HEADER

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT.
CYLINDER 240, HEAD 0, SECTOR 1. - CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:

000240
140000
140240

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000240
140000
140040

MAKE SURE HEADER VRC AND OPI ERROR SET.

*TST32: SCOPE

4195	017606	000004				MOV	#10,SIMES	:DO 10. ITERATIONS
4196	017610	012737	000012	001200		MOV	\$BASE,R2	:LOAD RK611 BASE
4197	017616	013702	001270			MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4198	017622	012762	100000	000000		MOV	#2500,R0	:SET COUNT FOR STALL
4199	017630	012700	002500			MOV		
4200	017634	005300			2S:	DEC	R0	:DEC COUNT
4201	017636	001376				BNE	2S	:LOOP UNTIL 0
4202	017640	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4203	017646	012762	055220	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUFFER ADDRESS
4204	017654	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT = 1
4205	017662	012762	000240	000020		MOV	#240,RKDCYL(R2)	:LOAD CYLINDER
4206	017670	012762	000001	000006		MOV	#1,RKDA(R2)	:LOAD TRACK AND SECTOR
4207	017676	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4208	017704	012700	000426			MOV	#69,*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL READY FOR SECTOR PULSE
4209								
4210	017710	012762	000440	000026	1S:	MOV	#DMD!MCLK,RKMR1(R2)	
4211	017716	012762	000040	000026		MOV	#DMD,RKMR1(R2)	

E07

4212	017724	005300				DEC	RO	
4213	017726	001370				BNE	15	
4214	017730	012705	000040			MOV	#32, R5	:LOAD HEADER COUNT
4215	017734	012703	054076			MOV	#OPI2, R3	:LOAD ADDRESS OF HEADERS
4216	017740	012737	000023	003200		MOV	#WRDATA, E.CS1	:LOAD EXPECTED CS1
4217	017746	012737	000300	003210		MOV	#IR!OR, E.CS2	:LOAD EXPECTED CS2
4218	017754	005037	003214			CLR	E.ER	:LOAD EXPECTED ERROR REG
4219	017760	012737	022040	003224		MOV	#DMD!MEWD!ECCW, E.MR1	:LOAD EXPECTED MR1
4220	017766	005037	003310			CLR	HDRCNT	:INITIALIZE HEADER COUNT
4221	017772	012762	000140	000026	55:	MOV	#DMD!MSP, RKMRI(R2)	:SIMULATE SECTOR PULSE
4222	020000	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
4223	020006	022737	000037	003310		CMP	#31., HDRCNT	:CHECK IF ALL HEADERS DONE
4224	020014	001460				BEQ	265	:YES - SKIP TO ERROR TEST
4225	020016	005037	003254			CLR	PR.BIT	:GENERATE SYNCH
4226	020022	005037	003256			CLR	M1.BIT	
4227	020026	012700	000377			MOV	#255., RO	
4228	020032	004737	037030		105:	JSR	PC, RDBIT	
4229	020036	005300				DEC	RO	:CHECK IF SYNCH FINISHED
4230	020040	001374				BNE	105	
4231	020042	012737	000001	003254		MOV	#1, PR.BIT	:SIMULATE SYNCH BIT
4232	020050	004737	037030			JSR	PC, RDBIT	
4233	020054	012701	000003			MOV	#3, R1	:SIMULATE OPI
4234	020060	012304			125:	MOV	(R3)+, R4	:GET NEXT HEADER WORD
4235	020062	012700	000020			MOV	#16., RO	:LOAD BITS PER WORD
4236	020066	013737	003254	003256	155:	MOV	PR.BIT, M1.BIT	:STORE PREVIOUS BIT
4237	020074	006004				ROR	R4	:GET NEXT BIT
4238	020076	103403				BCS	175	
4239	020100	005037	003254			CLR	PR.BIT	
4240	020104	000403				BR	185	
4241								
4242	020106	012737	000001	003254	175:	MOV	#1, PR.BIT	
4243	020114	004737	037030		185:	JSR	PC, RDBIT	:SIMULATE NEXT BIT
4244	020120	005300				DEC	RO	:CHECK IF READY FOR NEXT HEADER WORD
4245	020122	001361				BNE	155	:NO, CONTINUE
4246	020124	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER
4247	020126	001354				BNE	125	:NO, CONTINUE
4248	020130	012700	000100			MOV	#64., RO	:LOAD COUNT FOR GAP
4249	020134	013737	003254	003256	255:	MOV	PR.BIT, M1.BIT	:SIMULATE GAP
4250	020142	005037	003254			CLR	PR.BIT	
4251	020146	004737	037030			JSR	PC, RDBIT	
4252	020152	005300				DEC	RO	:CHECK IF GAP IS FINISHED
4253	020154	001367				BNE	255	:NO, CONTINUE
4254	020156	016237	000000	003140	265:	MOV	RKCS1(R2), T.CS1	:GET CS1
4255	020164	016237	000010	003150		MOV	RKCS2(R2), T.CS2	:GET CS2
4256	020172	016237	000014	003154		MOV	RKER(R2), T.ER	:GET ERROR REG.
4257	020200	023737	000200	003140		CMP	E.CS1, T.CS1	:CHECK CS1 CORRECT
4258	020206	001401				BEQ	305	:YES, CONTINUE
4259	020210	104100				ERROR	100	:CS1 INCORRECT
4260	020212	023737	000210	003150	305:	CMP	E.CS2, T.CS2	:CHECK CS2 CORRECT
4261	020220	001401				BEQ	315	:YES, CONTINUE
4262	020222	104101				ERROR	101	:CS2 INCORRECT
4263	020224	023737	003214	003154	315:	CMP	E.ER, T.ER	:CHECK ERROR REG CORRECT
4264	020232	001401				BEQ	325	:YES, CONTINUE
4265	020234	104102				ERROR	102	:ERROR REG INCORRECT
4266	020236	016237	000026	003164	325:	MOV	RKMRI(R2), T.MR1	:GET MR1
4267	020244	023737	003224	003164		CMP	E.MR1, T.MR1	:CHECK TO MAKE SURE WRITE GATE DID NOT SET

4268	020252	001401			BEG	34\$: YES, CONTINUE
4269	020254	104103			ERROR	103	: MRI INCORRECT
4270	020256	005237	003310		INC	HDRCNT	: INCREMENT HEADER COUNT
4271	020262	022737	000037	003310	34\$: CMP	#31.,HDCNT	: CHECK IF LAST HEADER
4272	020270	001011			BNE	35\$: NO, CHECK IF FINISHED
4273	020272	012737	020400	003214	MOV	#OPI!HVRC,E.ER	: LOAD ERROR BIT
4274	020300	042737	000001	003200	BIC	#GO,E.CS1	: ADJUST E.CS1 FOR END OF OP CONTENTS
4275	020306	052737	100200	003200	BIS	#RDY!CERR,E.CS1	
4276	020314	005305			35\$: DEC	RS	: CHECK IF FINISHED
4277	020316	001402			BEG	37\$: YES - SKIP
4278	020320	000137	017772		JMP	5\$: DO NEXT SECTOR
4279	020324	012762	100000	000000	37\$: MOV	#CLR,RKCS1(R2)	: CLEAR CONTROLLER
4280	020332	016237	000000	003140	MOV	RKCS1(R2),T.CS1	: GET CS1
4281	020340	016237	000010	003150	MOV	RKCS2(R2),T.CS2	: CS2
4282	020346	016237	000014	003154	MOV	RKER(R2),T.ER	: ER
4283	020354	012737	000200	003200	MOV	#RDY,E.CS1	: SET EXPECTED CS1
4284	020362	023737	003140	003200	CMP	T.CS1,E.CS1	: CHECK IF CORRECT
4285	020370	001401			BEG	TST33	: GO TO NEXT TEST
4286	020372	104153			ERROR	153	

*TEST 33 OPI AND PREVIOUS HEADER VRC (PART 1)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 300, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A
* SECTOR PULSE AND 30 HEADERS CONSISTING OF THE
* FOLLOWING 3 WORDS:

000300
140000
140300

* THEN SIMULATE A 3 WORD HEADER CONSISTING ON
* THE FOLLOWING DATA:

000300
140000
140200

* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND A
* HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000300
140000
140300

* MAKE SURE HEADER VRC AND OPI ERRORS SET.

4321	020374	000004			TST33: SCOPE		
4322	020376	012737	000012	001200	MOV	#10,\$TIMES	: DO 10. ITERATIONS
4323	020404	013702	001270		MOV	\$BASE,R2	: LOAD RK611 BASE

G07

4324	020410	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
4325	020416	012700	002500			MOV	#2500,RO	:SET COUNT FOR STALL
4326	020422	005300			25:	DEC	RO	:DEC COUNT
4327	020424	001376				BNE	25	:LOOP UNTIL 0
4328	020426	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4329	020434	012762	055220	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUFFER ADDRESS
4330	020442	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT = 1
4331	020450	012762	000300	000020		MOV	#300,RKDCYL(R2)	:LOAD CYLINDER
4332	020456	012762	000001	000006		MOV	#1,RKDA(R2)	:LOAD TRACK AND SECTOR
4333	020464	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4334	020472	012700	000426			MOV	#69.*4+2,RO	:ISSUE ENOUGH CLOCKS UNTIL READY
4335								:FOR SECTOR PULSE
4336	020476	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	
4337	020504	012762	000040	000026		MOV	#CMD,RKMR1(R2)	
4338	020512	005300				DEC	RO	
4339	020514	001370				BNE	15	
4340	020516	012705	000040			MOV	#32,R5	:LOAD HEADER COUNT
4341	020522	012703	054376			MOV	#OPI3,R3	:LOAD ADDRESS OF HEADERS
4342	020526	012737	000023	003200		MOV	#WRDATA,E.CS1	:LOAD EXPECTED CS1
4343	020534	012737	000300	003210		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
4344	020542	005037	003214			CLR	E.ER	:LOAD EXPECTED ERROR REG
4345	020546	012737	022040	003224		MOV	#DMD!MEWD!ECCW,E.MR1	:LOAD EXPECTED MRI
4346	020554	005037	003310			CLR	HDRCNT	:INITIALIZE HEADER COUNT
4347	020560	012762	000140	000026	55:	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
4348	020566	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4349	020574	022737	000037	003310		CMP	#31.,HDRCNT	:CHECK IF ALL HEADERS DONE
4350	020602	001460				BEQ	265	:YES - SKIP TO ERROR TEST
4351	020604	005037	003254			CLR	PR.BIT	:GENERATE SYNCH
4352	020610	005037	003256			CLR	M1.BIT	
4353	020614	012700	000377			MOV	#255,RO	
4354	020620	004737	037030		105:	JSR	PC,RDBIT	
4355	020624	005300				DEC	RO	:CHECK IF SYNCH FINISHED
4356	020626	001374				BNE	105	
4357	020630	012737	000001	003254		MOV	#1,PR.BIT	:SIMULATE SYNCH BIT
4358	020636	004737	037030			JSR	PC,RDBIT	
4359	020642	012701	000003			MOV	#3,R1	:SIMULATE OPI
4360	020646	012304			125:	MOV	(R3)+,R4	:GET NEXT HEADER WORD
4361	020650	012700	000020			MOV	#16,RO	:LOAD BITS PER WORD
4362	020654	013737	003254	003256	155:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4363	020662	006004				ROR	R4	:GET NEXT BIT
4364	020664	103403				BCS	175	
4365	020666	005037	003254			CLR	PR.BIT	
4366	020672	000403				BR	185	
4367								
4368	020674	012737	000001	003254	175:	MOV	#1,PR.BIT	
4369	020702	004737	037030		185:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
4370	020706	005300				DEC	RO	:CHECK IF READY FOR NEXT HEADER WORD
4371	020710	001361				BNE	155	:NO, CONTINUE
4372	020712	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER
4373	020714	001354				BNE	125	:NO, CONTINUE
4374	020716	012700	000100			MOV	#64,RO	:LOAD COUNT FOR GAP
4375	020722	013737	003254	003256	255:	MOV	PR.BIT,M1.BIT	:SIMULATE GAP
4376	020730	005037	003254			CLR	PR.BIT	
4377	020734	004737	037030			JSR	PC,RDBIT	
4378	020740	005300				DEC	RO	:CHECK IF GAP IS FINISHED
4379	020742	001367				BNE	255	:NO, CONTINUE

H07

4380	020744	016237	000000	003140	26%:	MOV	RKCS1(R2),T.CS1	:GET CS1
4381	020752	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:GET CS2
4382	020760	016237	000014	003154		MOV	RKER(R2),↑.ER	:GET ERROR REG.
4383	020766	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4384	020774	001401				BEQ	30%	:YES, CONTINUE
4385	020776	104104				ERROR	104	:CS1 INCORRECT
4386	021000	023737	003210	003150	30%:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4387	021006	001401				BEQ	31%	:YES, CONTINUE
4388	021010	104105				ERROR	105	:CS2 INCORRECT
4389	021012	023737	003214	003154	31%:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
4390	021020	001401				BEQ	32%	:YES, CONTINUE
4391	021022	104106				ERROR	106	:ERROR REG INCORRECT
4392	021024	016237	000026	003164	32%:	MOV	RKMRI(R2),T.MRI	:GET MRI
4393	021032	023737	003224	003164		CMP	E.MRI,T.MRI	:CHECK TO MAKE SURE WRITE GATE DID NOT SET
4394	021040	001401				BEQ	34%	:YES, CONTINUE
4395	021042	104107				ERROR	107	:MRI INCORRECT
4396	021044	005237	003310		34%:	INC	HDRCNT	:INCREMENT HEADER COUNT
4397	021050	022737	000037	003310		CMP	#31.,HDRCNT	:CHECK IF LAST HEADER
4398	021056	001011				BNE	35%	:NO, CHECK IF FINISHED
4399	021060	012737	020400	003214		MOV	#OPI!HVRC,E.ER	:LOAD ERROR BIT
4400	021066	042737	000001	003200		BIC	#GO,E.CS1	:ADJUST E.CS1 FOR END OF OP CONTENTS
4401	021074	052737	100200	003200		BIS	#RDY!CERR,E.CS1	
4402	021102	005305			35%:	DEC	R5	:CHECK IF FINISHED
4403	021104	001402				BEQ	37%	:YES - SKIP
4404	021106	000137	020560			JMP	5%	:DO NEXT SECTOR
4405	021112	012762	100000	000000	37%:	MOV	#CCLR,RKCS1(R2)	:CLEAR CONTROLLER
4406	021120	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
4407	021126	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:CS2
4408	021134	016237	000014	003154		MOV	RKER(R2),↑.ER	:ER
4409	021142	012737	000200	003200		MOV	#RDY,E.CS1	:SET EXPECTED CS1
4410	021150	023737	003140	003200		CMP	T.CS1,E.CS1	:CHECK IF CORRECT
4411	021156	001401				BEQ	TST34	:GO TO NEXT TEST
4412	021160	104153				ERROR	153	

*TEST 34 OPI AND PREVIOUS HEADER VRC (PART 2)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 40, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
* PULSE AND A HEADER CONSISTING OF THE FOLLOWING
* THREE WORDS HAVING A BAD HEADER VRC:

* 000040
* 140000
* 140000

* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND 31
* HEADERS CONSISTING OF THE FOLLOWING THREE WORDS:

* 000040
* 140000
* 140040

4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435

```

4436      *
4437      * MAKE SURE HEADER VRC AND OPI ERRORS SET.
4438      *
4439      * *****
4440 021162 000004 15:34: SCOPE
4441 021164 012737 000012 001200 MOV #10, $TIMES ;; DO 10. ITERATIONS
4442 021172 013702 001270 MOV $BASE, R2 ;; LOAD RK611 BASE
4443 021176 012762 100000 000000 MOV #CCLR, RKCS1(R2) ;; CLEAR RK611
4444 021204 012700 002500 MOV #2500, R0 ;; SET COUNT FOR STALL
4445 021210 005300 25: DEC R0 ;; DEC COUNT
4446 021212 001376 BNE 25 ;; LOOP UNTIL 0
4447 021214 012762 000040 000026 MOV #DMD, RKMRI(R2) ;; PUT RK611 IN DIAGNOSTIC MODE
4448 021222 012762 055220 000004 MOV #BUFF, RKBA(R2) ;; LOAD DUMMY BUFFER ADDRESS
4449 021230 012762 177777 000002 MOV #-1, RKWC(R2) ;; WORD COUNT = 1
4450 021236 012762 000040 000020 MOV #40, RKDCYL(R2) ;; LOAD CYLINDER
4451 021244 012762 000001 000006 MOV #1, RKDA(R2) ;; LOAD TRACK AND SECTOR
4452 021252 012762 000023 000000 MOV #WRDATA, RKCS1(R2) ;; ISSUE WRITE DATA
4453 021260 012700 000426 MOV #69.*4+2, R0 ;; ISSUE ENOUGH CLOCKS UNTIL READY
4454 ;; FOR SECTOR PULSE
4455 021264 012762 000440 000026 15: MOV #DMD!MCLK, RKMRI(R2)
4456 021272 012762 000040 000026 MOV #DMD, RKMRI(R2)
4457 021300 005300 DEC R0
4458 021302 001370 BNE 15
4459 021304 012705 000040 MOV #32, R5 ;; LOAD HEADER COUNT
4460 021310 012703 054676 MOV #OPI4, R3 ;; LOAD ADDRESS OF HEADERS
4461 021314 012737 000023 003200 MOV #WRDATA, E.CS1 ;; LOAD EXPECTED CS1
4462 021322 012737 000300 003210 MOV #IR!OR, E.CS2 ;; LOAD EXPECTED CS2
4463 021330 005037 003214 CLR E.ER ;; LOAD EXPECTED ERROR REG
4464 021334 012737 022040 003224 MOV #DMD!MEWD!ECCW, E.MRI ;; LOAD EXPECTED MRI
4465 021342 005037 003310 CLR HDRCNT ;; INITIALIZE HEADER COUNT
4466 021346 012762 000140 000026 55: MOV #DMD!MSP, RKMRI(R2) ;; SIMULATE SECTOR PULSE
4467 021354 012762 000040 000026 MOV #DMD, RKMRI(R2)
4468 021362 022737 000037 003310 CMP #31., HDRCNT ;; CHECK IF ALL HEADERS DONE
4469 021370 001460 BEQ 265 ;; YES - SKIP TO ERROR TEST
4470 021372 005037 003254 CLR PR.BIT ;; GENERATE SYNCH
4471 021376 005037 003256 CLR M1.BIT
4472 021402 012700 000377 MOV #255., R0
4473 021406 004737 037030 105: JSR PC, RDBIT
4474 021412 005300 DEC R0 ;; CHECK IF SYNCH FINISHED
4475 021414 001374 BNE 105
4476 021416 012737 000001 003254 MOV #1, PR.BIT ;; SIMULATE SYNCH BIT
4477 021424 004737 037030 JSR PC, RDBIT
4478 021430 012701 000003 MOV #3, R1 ;; SIMULATE OPI
4479 021434 012304 125: MOV (R3)+, R4 ;; GET NEXT HEADER WORD
4480 021436 012700 000020 MOV #16., R0 ;; LOAD BITS PER WORD
4481 021442 013737 003254 155: MOV PR.BIT, M1.BIT ;; STORE PREVIOUS BIT
4482 021450 006004 ROR R4 ;; GET NEXT BIT
4483 021452 103403 BCS 175
4484 021454 005037 003254 CLR PR.BIT
4485 021460 000403 BR 185
4486
4487 021462 012737 000001 003254 175: MOV #1, PR.BIT
4488 021470 004737 037030 185: JSR PC, RDBIT ;; SIMULATE NEXT BIT
4489 021474 005300 DEC R0 ;; CHECK IF READY FOR NEXT HEADER WORD
4490 021476 001361 BNE 155 ;; NO, CONTINUE
4491 021500 005301 DEC R1 ;; CHECK IF FINISHED WITH HEADER

```

```

4492 021502 001354 BNE 12$ :NO CONTINUE
4493 021504 012700 000100 MOV #64.,RO :LOAD COUNT FOR GAP
4494 021510 013737 003254 003256 25$: MOV PR.BIT,M1.BIT :SIMULATE GAP
4495 021516 005037 003254 CLR PR.BIT
4496 021522 004737 037030 JSR PC,RDBIT
4497 021526 005300 DEC RO ;CHECK IF GAP IS FINISHED
4498 021530 001367 BNE 25$ :NO CONTINUE
4499 021532 016237 000000 003140 26$: MOV RKCS1(R2),T.CS1 :GET CS1
4500 021540 016237 000010 003150 MOV RKCS2(R2),T.CS2 :GET CS2
4501 021546 016237 000014 003154 MOV RKER(R2),T.ER :GET ERROR REG.
4502 021554 023737 003200 003140 CMP E.CS1,T.CS1 :CHECK CS1 CORRECT
4503 021562 001401 BEQ 30$ :YES, CONTINUE
4504 021564 104110 ERROR 110 :CS1 INCORRECT
4505 021566 023737 003210 003150 30$: CMP E.CS2,T.CS2 :CHECK CS2 CORRECT
4506 021574 001401 BEQ 31$ :YES, CONTINUE
4507 021576 104111 ERROR 111 :CS2 INCORRECT
4508 021600 023737 003214 003154 31$: CMP E.ER,T.ER :CHECK ERROR REG CORRECT
4509 021606 001401 BEQ 32$ :YES, CONTINUE
4510 021610 104112 ERROR 112 :ERROR REG INCORRECT
4511 021612 016237 000026 003164 32$: MOV RKMRI(R2),T.MRI :GET MRI
4512 021620 023737 003224 003164 CMP E.MRI,T.MRI :CHECK TO MAKE SURE WRITE GATE DID NOT SET
4513 021626 001401 BEQ 34$ :YES, CONTINUE
4514 021630 104113 ERROR 113 :MRI INCORRECT
4515 021632 005237 003310 003310 34$: INC HDRCNT :INCREMENT HEADER COUNT
4516 021636 022737 000037 003310 CMP #31.,HDRCNT :CHECK IF LAST HEADER
4517 021644 001011 BNE 35$ :NO, CHECK IF FINSHED
4518 021646 012737 020400 003214 MOV #OPI!HVRC,E.ER :LOAD ERROR BIT
4519 021654 042737 000001 003200 BIC #GO,E.CS1 :ADJUST E.CS1 FOR END OF OP CONTENTS
4520 021662 052737 100200 003200 BIS #RDY!CERR,E.CS1
4521 021670 005305 DEC R5 :CHECK IF FINISHED
4522 021672 001402 BEQ 37$ :YES - SKIP
4523 021674 000137 021346 JMP 5$ :DO NEXT SECTOR
4524 021700 012762 100000 000000 37$: MOV #CCLR,RKCS1(R2) :CLEAR CONTROLLER
4525 021706 016237 000000 003140 MOV RKCS1(R2),T.CS1 :GET CS1
4526 021714 016237 000010 003150 MOV RKCS2(R2),T.CS2 : CS2
4527 021722 016237 000014 003154 MOV RKER(R2),T.ER : ER
4528 021730 012737 000200 003200 MOV #RDY,E.CS1 :SET EXPECTED CS1
4529 021736 023737 003140 003200 CMP T.CS1,E.CS1 :CHECK IF CORRECT
4530 021744 001401 BEQ TST35 ;GO TO NEXT TEST
4531 021746 104153 ERROR 153

```

```

*****
*TEST 35 BSE AND CONTROLLER ERROR

```

```

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
* OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A HEADER WITH A BSE ERROR. MAKE SURE CONTROLLER
* ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
* CONTROLLER ERROR RESETS.

```

```

*****
TST35: SCOPE

```

```

4547 021750 000004

```



K07

4548	021752	012737	000010	001200		MOV	#10, \$TIMES	;DC 10 ITERATIONS
4549	021760	013702	001270			MOV	\$BASE, R2	;LOAD RK611 BASE
4550	021764	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	;CLEAR RK611
4551	021772	012762	000040	000026		MOV	#DMD, RKMRI(R2)	;PUT RK611 IN DIAGNOSTIC MODE
4552	022000	012762	055220	000004		MOV	#BUFF, RKBA(R2)	;LOAD DUMMY BUS ADDRESS
4553	022006	012762	177777	000002		MOV	#-1, RKWC(R2)	;WORD COUNT=1
4554	022014	012762	000000	000020		MOV	#0, RKDCYL(R2)	;LOAD CYLINDER 11DMS
4555	022022	012762	000000	000006		MOV	#0, RKDA(R2)	;LOAD TRACK AND SECTOR
4556	022030	012762	000023	000000		MOV	#WRDATA, RKCS1(R2)	;ISSUE COMMAND
4557	022036	012700	000426			MOV	#69.*4+2, R0	;ISSUE ENOUGH CLOCKS UNTIL ;READY FOR SECTOR PULSE.
4558								
4559	022042	012762	000440	000026	15:	MOV	#DMD!MCLK, RKMRI(R2)	
4560	022050	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
4561	022056	005300				DEC	R0	
4562	022060	001370				BNE	15	
4563	022062	012762	000140	000026		MOV	#DMD!MSP, RKMRI(R2)	;SIMULATE SECTOR PULSE
4564	022070	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
4565	022076	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
4566	022102	005037	003256			CLR	M1.BIT	
4567	022106	012700	000377			MOV	#255, R0	
4568	022112	004737	037030		55:	JSR	PC, RDBIT	
4569	022116	005300				DEC	R0	;CHECK IF SYNCH FINISHED
4570	022120	001374				BNE	55	
4571	022122	012737	000001	003254		MOV	#1, PR.BIT	;SIMULATE SYNCH
4572	022130	004737	037030			JSR	PC, RDBIT	
4573	022134	012703	053374			MOV	#HEAD2, R3	;LOAD HEADER
4574	022140	012701	000003			MOV	#3, R1	;LOAD WORDS PER HEADER
4575	022144	012304			105:	MOV	(R3)+, R4	;GET NEXT WORD
4576	022146	012700	000020			MOV	#16, R0	;LOAD BITS PER WORD
4577	022152	013737	003254	003256	125:	MOV	PR.BIT, M1.BIT	;STORE PREVIOUS BIT
4578	022160	006004				ROR	R4	;GET NEXT BIT
4579	022162	103403				BCS	155	
4580	022164	005037	003254			CLR	PR.BIT	
4581	022170	000403				BR	165	
4582								
4583	022172	012737	000001	003254	155:	MOV	#1, PR.BIT	
4584	022200	004737	037030		165:	JSR	PC, RDBIT	;SIMULATE NEXT BIT
4585	022204	005300				DEC	R0	;CHECK IF READY FOR NEXT WORD
4586	022206	001361				BNE	125	;NO, CONTINUE
4587	022210	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
4588	022212	001354				BNE	105	;NO, CONTINUE
4589	022214	012700	000101			MOV	#65, R0	
4590	022220	013737	003254	003256	205:	MOV	PR.BIT, M1.BIT	;SIMULATE GAP
4591	022226	005037	003254			CLR	PR.BIT	
4592	022232	004737	037030			JSR	PC, RDBIT	
4593	022236	005300				DEC	R0	;CHECK IF GAP FINISHED
4594	022240	001367				BNE	205	;NO, CONTINUE
4595	022242	012700	021200			MOV	#2*(256.<+16.*256.>+64.), R0	;LOAD COUNT UNTIL POSTAMBLE
4596	022246	012762	000440	000026	255:	MOV	#DMD!MCLK, RKMRI(R2)	
4597	022254	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
4598	022262	005300				DEC	R0	
4599	022264	001370				BNE	255	
4600	022266	016237	000000	003140		MOV	RKCS1(R2), T.CS1	;GET CS1
4601	022274	016237	000010	003150		MOV	RKCS2(R2), T.CS2	;GET CS2
4602	022302	016237	000014	003154		MOV	RKER(R2), T.ER	;GET ERROR REG
4603	022310	012737	100222	003200		MOV	#CERR!RDY!WRDATA<+C<GO>>, E.CS1	;LOAD EXPECTED CS1

```

4604 022316 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4605 022324 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG
4606 022332 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4607 022340 001401 BEQ 30$ ;YES, CONTINUE
4608 022342 104136 ERROR 136 ;CS1 INCORRECT
4609 022344 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4610 022352 001401 BEQ 32$ ;YES, CONTINUE
4611 022354 104137 ERROR 137 ;CS2 INCORRECT
4612 022356 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4613 022364 001401 BEQ 35$ ;YES - SKIP
4614 022366 104140 ERROR 140 ;ERROR REG INCORRECT
4615 022370 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4616 022376 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4617 022404 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4618 022412 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4619 022420 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4620 022426 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4621 022434 001401 BEQ TST36 ;GO TO NEXT TEST
4622 022436 104153 ERROR 153 ;ERROR DID NOT CLEAR

```

```

*****
*TEST 36 HVRC AND CONTROLLER ERROR

```

```

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
* OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
* ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
* CONTROLLER ERROR RESETS.

```

```

*****
*TEST 36: SCOPE

```

```

4637 022440 000004 5*36: SCOPE
4638 022442 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
4639 022450 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4640 022454 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4641 022462 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4642 022470 012762 055220 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
4643 022476 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
4644 022504 012762 000300 000020 MOV #300,RKDCYL(R2) ;LOAD CYLINDER 11DMS
4645 022512 012762 000000 000006 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
4646 022520 012762 000023 000000 MOV #WADATA,RKCS1(R2) ;ISSUE COMMAND
4647 022526 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE.
4648
4649 022532 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4650 022540 012762 000040 000026 MOV #DMD,RKMR1(R2)
4651 022546 005300 DEC R0
4652 022550 001370 BNE 1$
4653 022552 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4654 022560 012762 000040 000026 MOV #DMD,RKMR1(R2)
4655 022566 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4656 022572 005037 003256 CLR M1.BIT
4657 022576 012700 000377 MOV #255,R0
4658 022602 004737 037030 5$: JSR PC,R0BIT
4659 022606 005300 DEC R0 ;CHECK IF SYNCH FINISHED

```

M07

```

4660 022610 001374 BNE 5$
4661 022612 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4662 022620 004737 037030 JSR PC,RDBIT
4663 022624 012703 053476 MOV #HEAD10,R3 ;LOAD HEADER
4664 022630 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4665 022634 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4666 022636 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4667 022642 013737 003254 003256 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4668 022650 006004 ROR R4 ;GET NEXT BIT
4669 022652 103403 BCS 15$
4670 022654 005037 003254 CLR PR.BIT
4671 022660 000403 BR 16$
4672
4673 022662 012737 000001 003254 15$: MOV #1,PR.BIT
4674 022670 004737 037030 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4675 022674 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4676 022676 001361 BNE 12$ ;NO, CONTINUE
4677 022700 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4678 022702 001354 BNE 10$ ;NO, CONTINUE
4679 022704 012700 000101 MOV #65,R0
4680 022710 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4681 022716 005037 003254 CLR PR.BIT
4682 022722 004737 037030 JSR PC,RDBIT
4683 022726 005300 DEC R0 ;CHECK IF GAP FINISHED
4684 022730 001367 BNE 20$ ;NO, CONTINUE
4685 022732 012700 021200 MOV #2*(256.+<16.*256.>+64.),R0 ;LOAD COUNT UNTIL POSTAMBLE
4686 022736 012762 000440 000026 25$: MOV #DMD!MCLK,RKMRI(R2)
4687 022744 012762 000040 000026 MOV #DMD,RKMRI(R2)
4688 022752 005300 DEC R0
4689 022754 001370 BNE 25$
4690 022756 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4691 022764 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4692 022772 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4693 023000 012737 100222 003200 MOV #CERR!RDY!WRDATA<1C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4694 023006 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4695 023014 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4696 023022 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4697 023030 001401 BEQ 30$ ;YES, CONTINUE
4698 023032 104141 ERROR 141 ;CS1 INCORRECT
4699 023034 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4700 023042 001401 BEQ 32$ ;YES, CONTINUE
4701 023044 104142 ERROR 142 ;CS2 INCORRECT
4702 023046 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4703 023054 001401 BEQ 35$ ;YES - SKIP
4704 023056 104143 ERROR 143 ;ERROR REG INCORRECT
4705 023060 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4706 023066 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4707 023074 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;CS2
4708 023102 016237 000014 003154 MOV RKER(R2),T.ER ;ER
4709 023110 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4710 023116 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4711 023124 001401 BEQ TST37 ;GO TO NEXT TEST
4712 023126 104153 ERROR 153 ;ERROR DID NOT CLEAR

```

```

*****
; *TEST 37 READ DATA AND HVRC ERROR

```

4713
4714
4715

```

4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771

```

```

:*
:* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
:* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
:* OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
:* CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
:* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
:* AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
:* ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
:* CONTROLLER ERROR RESETS.

```

```

T37: SCOPE
MOV #10, $TIMES ;; DO 10. ITERATIONS
MOV $BASE, R2 ;; LOAD RK611 BASE
MOV #CLR, RKCS1(R2) ;; CLEAR RK611
MOV #DMD, RKMR1(R2) ;; PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF, RKBA(R2) ;; LOAD DUMMY BUS ADDRESS
MOV #-1, RKWC(R2) ;; WORD COUNT=1
MOV #300, RKDCYL(R2) ;; LOAD CYLINDER 110MS
MOV #0, RKDA(R2) ;; LOAD TRACK AND SECTOR
MOV #RODATA, RKCS1(R2) ;; ISSUE COMMAND
MOV #69.*4+2, R0 ;; ISSUE ENOUGH CLOCKS UNTIL
;; READY FOR SECTOR PULSE.
1$: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
5$: JSR PC, R0BIT
DEC R0 ; CHECK IF SYNCH FINISHED
BNE 5$
MOV #1, PR.BIT ; SIMULATE SYNCH
JSR PC, R0BIT
MOV #HEAD10, R3 ; LOAD HEADER
MOV #3, R1 ; LOAD WORDS PER HEADER
10$: MOV (R3)+, R4 ; GET NEXT WORD
MOV #16, R0 ; LOAD BITS PER WORD
12$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 15$
CLR PR.BIT
BR 16$
15$: MOV #1, PR.BIT
16$: JSR PC, R0BIT ; SIMULATE NEXT BIT
DEC R0 ; CHECK IF READY FOR NEXT WORD
BNE 12$ ; NO, CONTINUE
DEC R1 ; CHECK IF FINISHED WITH HEADER
BNE 10$ ; NO, CONTINUE
MOV #65, R0
20$: MOV PR.BIT, M1.BIT ; SIMULATE GAP
CLR PR.BIT

```


477	023412	004737	037030			JSR	PC,ROBIT	
478	023416	005300				DEC	R0	:CHECK IF GAP FINISHED
479	023420	001367				BNE	20\$:NO CONTINUE
480	023420	012700	021200			MOV	#2*(256.+(16.*256.)+64.)R0	:LOAD COUNT UNTIL POSTAMBLE
481	023420	012762	000440	000026	25\$:	MOV	#DMD,MCLK,RKMR1(R2)	
482	023420	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
483	023442	005300				DEC	R0	
484	023444	001370				BNE	25\$	
485	023446	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
486	023450	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:GET CS2
487	023460	016237	000014	003154		MOV	RKER(R2),↑.ER	:GET ERROR REG
488	023470	012737	100220	003200		MOV	#CERR!RDY!RDATA<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
489	023470	012737	000100	003210		MOV	#↑R,E.CS2	:LOAD EXPECTED CS2
490	023470	012737	000400	003214		MOV	#HVRC,E.ER	:LOAD EXPECTED ERROR REG
491	023470	023737	003200	003140		COMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
492	023520	001401				BEQ	30\$:YES, CONTINUE
493	023520	104141				ERROR	141	:CS1 INCORRECT
494	023524	023737	003210	003150	30\$:	COMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
495	023530	001401				BEQ	32\$:YES, CONTINUE
496	023534	104142				ERROR	142	:CS2 INCORRECT
497	023536	023737	003214	003154	32\$:	COMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
498	023544	001401				BEQ	35\$:YES - SKIP
499	023546	104143				ERROR	143	:ERROR REG INCORRECT
500	023550	012762	100000	000000	35\$:	MOV	#CCLR,RKCS1(R2)	:CLEAR CONTROLLER
501	023556	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
502	023564	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:CS2
503	023570	016237	000014	003154		MOV	RKER(R2),↑.ER	:ER
504	023600	012737	000200	003200		MOV	#200,E.CS1	:SET EXPECTED CS1
505	023606	023737	003140	003200		COMP	T.CS1,E.CS1	:CHECK IF CORRECT
506	023614	001401				BEQ	TST40	:GO TO NEXT TEST
507	023616	104153				ERROR	153	:ERROR DID NOT CLEAR

```

*****
*TEST 40 WRITE CHECK AND HVRC
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
* A HEADER WITH A HEADER VRC ERROR. MAKE SURE
* HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
* AND MAKE SURE CONTROLLER ERROR RESETS.
*****

```

4817	023620	000004				TST40:	SCOPE	
4818	023622	012737	000012	001200		MOV	#10,\$TIMES	:DO 10. ITERATIONS
4819	023630	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
4820	023634	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4821	023642	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4822	023650	012762	055220	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
4823	023656	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT=1
4824	023664	012762	000300	000020		MOV	#300,RKDCYL(R2)	:LOAD CYLINDER 110MS
4825	023672	012762	000000	000006		MOV	#0,RKDA(R2)	:LOAD TRACK AND SECTOR
4826	023700	012762	000031	000000		MOV	#WRTCHK,RKCS1(R2)	:ISSUE COMMAND
4827	023706	012700	000426			MOV	#69.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL

C08

READY FOR SECTOR PULSE.

4828									
4829	023712	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)		
4830	023720	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4831	023726	005300				DEC	RO		
4832	023730	001370				BNE	15		
4833	023732	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE	
4834	023740	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4835	023746	005037	003254			CLR	PR.BIT	:GENERATE SYNCH	
4836	023752	005037	003256			CLR	M1.BIT		
4837	023756	012700	000377			MOV	#255,RO		
4838	023762	004737	037030		55:	JSR	PC,RDBIT		
4839	023766	005300				DEC	RO	:CHECK IF SYNCH FINISHED	
4840	023770	001374				BNE	55		
4841	023772	012737	000001	003254		MOV	#1,PR.BIT	:SIMULATE SYNCH	
4842	024000	004737	037030			JSR	PC,RDBIT		
4843	024004	012703	053476			MOV	#HEAD10,R3	:LOAD HEADER	
4844	024010	012701	000003			MOV	#3,R1	:LOAD WORDS PER HEADER	
4845	024014	012304			105:	MOV	(R3)+,R4	:GET NEXT WORD	
4846	024016	012700	000020			MOV	#16,RO	:LOAD BITS PER WORD	
4847	024022	013737	003254	003256	125:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT	
4848	024030	006004				ROR	R4	:GET NEXT BIT	
4849	024032	103403				BCS	155		
4850	024034	005037	003254			CLR	PR.BIT		
4851	024040	000403				BR	165		
4852									
4853	024042	012737	000001	003254	155:	MOV	#1,PR.BIT		
4854	024050	004737	037030		165:	JSR	PC,RDBIT	:SIMULATE NEXT BIT	
4855	024054	005300				DEC	RO	:CHECK IF READY FOR NEXT WORD	
4856	024056	001361				BNE	125	:NO, CONTINUE	
4857	024060	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER	
4858	024062	001354				BNE	105	:NO, CONTINUE	
4859	024064	012700	000101			MOV	#65,RO		
4860	024070	013737	003254	003256	205:	MOV	PR.BIT,M1.BIT	:SIMULATE GAP	
4861	024076	005037	003254			CLR	PR.BIT		
4862	024102	004737	037030			JSR	PC,RDBIT		
4863	024106	005300				DEC	RO	:CHECK IF GAP FINISHED	
4864	024110	001367				BNE	205	:NO, CONTINUE	
4865	024112	012700	021200			MOV	#2*(256+(16*256)+64),RO	:LOAD COUNT UNTIL POSTAMBLE	
4866	024116	012762	000440	000026	255:	MOV	#DMD!MCLK,RKMR1(R2)		
4867	024124	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4868	024132	005300				DEC	RO		
4869	024134	001370				BNE	255		
4870	024136	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1	
4871	024144	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:GET CS2	
4872	024152	016237	000014	003154		MOV	RKER(R2),T.ER	:GET ERROR REG	
4873	024160	012737	100230	003200		MOV	#CERR!RDY!WRTCHK<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1	
4874	024166	012737	000100	003210		MOV	#IR,E.CS2	:LOAD EXPECTED CS2	
4875	024174	012737	000400	003214		MOV	#HVRC,E.ER	:LOAD EXPECTED ERROR REG	
4876	024202	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT	
4877	024210	001401				BEQ	305	:YES, CONTINUE	
4878	024212	104141				ERROR	141	:CS1 INCORRECT	
4879	024214	023737	003210	003150	305:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT	
4880	024222	001401				BEQ	325	:YES, CONTINUE	
4881	024224	104142				ERROR	142	:CS2 INCORRECT	
4882	024226	023737	003214	003154	325:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT	
4883	024234	001401				BEQ	355	:YES - SKIP	

4884	024236	104143				ERROR	143	:ERROR REG INCORRECT
4885	024240	012762	100000	000000	35:	MOV	#CCLR,RKCS1(R2)	:CLEAR CONTROLLER
4886	024246	016237	000000	003140		MOV	RKCS1(R2),T,CS1	:GET CS1
4887	024254	016237	000010	003150		MOV	RKCS2(R2),T,CS2	:CS2
4888	024262	016237	000014	003154		MOV	RKER(R2),T,ER	:ER
4889	024270	012737	000200	003200		MOV	#200,E,CS1	:SET EXPECTED CS1
4890	024276	023737	003140	003200		CMP	T,CS1,E,CS1	:CHECK IF CORRECT
4891	024304	001401				BEQ	TS41	:GO TO NEXT TEST
4892	024306	104153				ERROR	153	:ERROR DID NOT CLEAR

.SBTTL **ECC GENERATION TESTS

:TEST 41 ECC INITIALIZATION

:CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
:PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
:DATA OF 10 WORDS TO AN RK06, IN 26 SECTOR
:FORMAT, CYLINDER 0, HEAD 0, SECTOR 3. CLOCK THROUGH
:SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
:A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY FOUR
:DATA WORDS OF ZEROES. MAKE SURE THE ECC PATTERN
:REGISTER REMAINS ZERO.

4910	024310	000004				TS41:	SCOPE	
4911	024312	012737	000012	001200		MOV	#10,STIMES	:DO 10 ITERATIONS
4912	024320	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
4913	024324	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4914	024332	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4915	024340	012762	053512	000004		MOV	#ECC1,RKBA(R2)	:LOAD ADDRESS OF ECC DATA
4916	024346	012762	177770	000002		MOV	#-10,RKWC(R2)	:WORD COUNT =10
4917	024354	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4918	024362	012700	000450			MOV	#8,*37.,R0	:ISSUE ENOUGH CLOCKS UNTIL :READY FOR SECTOR PULSE
4920	024366	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	
4921	024374	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4922	024402	005300				DEC	R0	
4923	024404	001370				BNE	15	
4924	024406	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
4925	024414	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4926	024422	005037	003254			CLR	PR.BIT	:GENERATE SYNCH
4927	024426	005037	003256			CLR	M1.BIT	
4928	024432	012700	000377			MOV	#255,R0	
4929	024436	004737	037030		55:	JSR	PC,R0BIT	
4930	024442	005300				DEC	R0	:CHECK IF SYNCH FINISHED
4931	024444	001374				BNE	55	
4932	024446	012737	000001	003254		MOV	#1,PR.BIT	:SIMULATE SYNCH BIT
4933	024454	004737	037030			JSR	PC,R0BIT	
4934	024460	012701	000003			MOV	#3,R1	:SIMULATE HEADER
4935	024464	012703	053366			MOV	#HEAD1,R3	:CYL 0, TRK 0, SECTOR 0
4936	024470	012304			125:	MOV	(R3)+,R4	:GET NEXT HEADER WORD
4937	024472	012700	000020			MOV	#16,R0	:LOAD BITS PER WORD
4938	024476	013737	003254	003256	155:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4939	024504	006004				ROR	R4	:GET NEXT BIT

4940	024506	103403				BCS	17\$		
4941	024510	005037	003254			CLR	PR.BIT		
4942	024514	000403				BR	18\$		
4943									
4944	024516	012737	000001	003254	17\$:	MOV	#1.PR.BIT		
4945	024524	004737	037030		18\$:	JSR	PC,RDBIT		;SIMULATE NEXT BIT
4946	024530	005300				DEC	R0		;CHECK IF READY FOR NEXT HEADER WORD
4947	024532	001361				BNE	15\$;NO, CONTINUE
4948	024534	005301				DEC	R1		;CHECK IF FINISHED WITH HEADER
4949	024536	001354				BNE	12\$;NO, CONTINUE
4950	024540	012700	000101			MOV	#255,R0		;SIMULATE GAP +1 FOR SWITCH FROM READ TO WRITE
4951	024544	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT		
4952	024552	005037	003254			CLR	PR.BIT		
4953	024556	004737	037030			JSR	PC,RDBIT		
4954	024562	005300				DEC	R0		;CHECK IF GAP FINISHED
4955	024564	001367				BNE	20\$;NO, CONTINUE
4956	024566	012700	000400			MOV	#256,R0		;LOAD COUNT FOR SYNCH FIELD
4957	024572	012737	050672	001310		MOV	#EM327,EMW		;LOAD ERROR MESSAGE
4958	024600	005037	003252			CLR	P1.BIT		;INITIALIZE BITS
4959	024604	005037	003254			CLR	PR.BIT		
4960	024610	005037	003256			CLR	M1.BIT		
4961	024614	005037	003260			CLR	M2.BIT		
4962	024620	005037	003262			CLR	BITCNT		;INITIALIZE BIT COUNT
4963	024624	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WRTGAT.E.MR1		;LOAD EXPECTED MR1
4964	024632	004737	036302		25\$:	JSR	PC,WRTBIT		;WRITE BIT
4965	024636	104002				ERROR	2		;DATA INCORRECT
4966	024640	005237	003262			INC	BITCNT		;INCREMENT BIT COUNT
4967	024644	005300				DEC	R0		;CHECK IF FINISHED
4968	024645	001371				BNE	25\$;NO, CONTINUE
4969	024650	012737	000001	003252		MOV	#1,P1.BIT		;SIMULATE SYNCH BIT
4970	024656	004737	036302			JSR	PC,WRTBIT		
4971	024662	104002				ERROR	2		
4972	024664	005037	003262			CLR	BITCNT		;CLEAR BIT COUNT
4973	024670	012737	050744	001310		MOV	#EM328,EMW		;LOAD ERROR MESSAGE
4974	024676	005037	003234			CLR	E.ECPT		;CLEAR EXPECTED ECC PATTERN
4975	024702	012703	053512			MOV	#ECC1,R3		;LOAD START OF DATA
4976	024706	012701	000004			MOV	#4,R1		;LOAD COUNT
4977	024712	012304			30\$:	MOV	(R3)+,R4		;GET NEXT WORD
4978	024714	012700	000020			MOV	#16,R0		;LOAD BITS PER WORD
4979	024720	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT		;SHIFT BITS
4980	024726	013737	003254	003256		MOV	PR.BIT,M1.BIT		
4981	024734	013737	003252	003254		MOV	P1.BIT,PR.BIT		
4982	024742	006004				ROR	R4		;GET NEXT BIT
4983	024744	103403				BCS	34\$		
4984	024746	005037	003252			CLR	P1.BIT		
4985	024752	000403				BR	35\$		
4986									
4987	024754	012737	000001	003252	34\$:	MOV	#1,P1.BIT		
4988	024762	004737	036302		35\$:	JSR	PC,WRTBIT		;SIMULATE BIT
4989	024766	104002				ERROR	2		
4990	024770	016237	000032	003174		MOV	RKECPT(R2),T.ECPT		;STORE ECC WORD
4991	024776	023737	003234	003174		CMP	E.ECPT,T.ECPT		;CHECK ECC PATTERN CORRECT
4992	025004	001401				BEG	37\$;YES, CONTINUE
4993	025006	104134				ERROR	134		;ECC PATTERN NOT ZERO
4994	025010	005237	003262		37\$:	INC	BITCNT		;INCREMENT BIT COUNT
4995	025014	005300				DEC	R0		;CHECK IF FINISHED WITH WORD

4996	025016	001340	BNE	32S	:NO, CONTINUE
4997	025020	005301	DEC	R1	:CHECK IF FINISHED WITH TEST
4998	025022	001333	BNE	30S	:NO, CONTINUE

```

5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051

```

```

*****
*TEST 42      ECC GENERATION (PART 1)
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
* DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
* FORMAT, CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR
* PULSE AND A GOOD HEADER.  CLOCK THROUGH ONLY THE
* FOLLOWING SIX WORDS OF DATA:
*
*          005001
*          040040
*          020004
*          000064
*          000000
*          000000
*
* CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
* GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.
*****

```

```

*****
*ST42: SCOPE
*MOV      #10, $TIMES      ;; DO 10. ITERATIONS
*MOV      $BASE, R2       ;; LOAD RK611 BASE
*MOV      @CLR, RKCS1(R2) ; CLEAR RK611
*MOV      @DMD, RKMRI(R2) ; PUT RK611 IN DIAGNOSTIC MODE
*MOV      @ECC2, RKBA(R2) ; LOAD ECC DATA
*MOV      #-12, RKWC(R2)  ; WORD COUNT=12
*MOV      @WRDATA, RKCS1(R2) ; ISSUE WRITE DATA
*MOV      #10, *37., R0   ; ISSUE ENOUGH CLOCKS UNTIL
*                               ; READY FOR SECTOR PULSE
1$: MOV      @DMD!MCLK, RKMRI(R2)
*MOV      @DMD, RKMRI(R2)
*DEC      R0
*BNE     1$
*MOV      @DMD!MSP, RKMRI(R2) ; SIMULATE SECTOR PULSE
*MOV      @DMD, RKMRI(R2)
*CLR      PR.BIT          ; GENERATE SYNCH
*CLR      M1.BIT
*MOV      #255., R0
5$: JSR     PC, R0BIT
*DEC      R0              ; CHECK IF SYNCH FINISHED
*BNE     5$
*MOV      #1, PR.BIT     ; SIMULATE SYNCH BIT
*JSR     PC, R0BIT
*MOV      #3, R1         ; SIMULATE HEADER
*MOV      @HEAD1, R3     ; CYL 0, TRK 0, SECTOR 0
*MOV      (R3)+, R4      ; GET NEXT HEADER WORD
12$: MOV      #16., R0   ; LOAD BITS PER WORD
*MOV      PR.BIT, M1.BIT ; STORE PREVIOUS BIT
15$: MOV      R0
*RROR

```

5052	025222	103403				BCS	17\$		
5053	025224	005037	003254			CLR	PR.BIT		
5054	025230	000403				BR	18\$		
5055									
5056	025232	012737	000001	003254	17\$:	MOV	#1,PR.BIT		
5057	025240	004737	037030		18\$:	JSR	PC,RD'IT		:SIMULATE NEXT BIT
5058	025244	005300				DEC	RO		:CHECK IF READY FOR NEXT HEADER WORD
5059	025246	001361				BNE	15\$:NO, CONTINUE
5060	025250	005301				DEC	R1		:CHECK IF FINISHED WITH HEADER
5061	025252	001354				BNE	12\$:NO, CONTINUE
5062	025254	012700	000101			MOV	#65,RO		:SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5063	025260	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT		
5064	025266	005037	003254			CLR	PR.BIT		
5065	025272	004737	037030			JSR	PC,RDBIT		
5066	025276	005300				DEC	RO		:CHECK IF GAP FINISHED
5067	025300	001367				BNE	20\$:NO, CONTINUE
5068	025302	012700	000400			MOV	#256,RO		:LOAD COUNT FOR SYNCH FIELD
5069	025306	012737	050672	001310		MOV	#EM327,EMW		:LOAD ERROR
5070	025314	005037	003252			CLR	P1.BIT		:INITIALIZE BITS
5071	025320	005037	003254			CLR	PR.BIT		
5072	025324	005037	003256			CLR	M1.BIT		
5073	025330	005037	003260			CLR	M2.BIT		
5074	025334	005037	003262			CLR	BITCNT		:INITIALIZE BIT COUNT
5075	025340	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WRTGAT,E.MR1		:SET EXPECTED MR1
5076	025346	004737	036302		25\$:	JSR	PC,WRTBIT		:WRITE BIT
5077	025352	104002				ERROR	2		:DATA INCORRECT
5078	025354	005237	003262			INC	BITCNT		:INCREMENT BIT COUNT
5079	025360	005300				DEC	RO		:CHECK IF FINISHED
5080	025362	001371				BNE	25\$:NO, CONTINUE
5081	025364	012737	000001	003252		MOV	#1,P1.BIT		:SIMULATE SYNCH BIT
5082	025372	004737	036302			JSR	PC,WRTBIT		
5083	025376	104002				ERROR	2		
5084	025400	005037	003262			CLR	BITCNT		:CLEAR BIT COUNT
5085	025404	012737	050744	001310		MOV	#EM328,EMW		:LOAD ERROR MESSAGE
5086	025412	005037	003266			CLR	ECCHI		:INITIALIZE ECC
5087	025416	005037	003270			CLR	ECCLO		
5088	025422	012703	053532			MOV	#ECC2,R3		:LOAD START OF DATA
5089	025426	012701	000006			MOV	#6,R1		:LOAD COUNT
5090	025432	012304			30\$:	MOV	(R3)+,R4		:GET NEXT BIT
5091	025434	012700	000020			MOV	#16,RO		:LOAD BITS PER WORD
5092	025440	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT		:SHIFT BITS
5093	025446	013737	003254	003256		MOV	PR.BIT,M1.BIT		
5094	025454	013737	003252	003254		MOV	P1.BIT,PR.BIT		
5095	025462	006004				ROR	R4		:GET NEXT BIT
5096	025464	103403				BCS	34\$		
5097	025466	005037	003252			CLR	P1.BIT		
5098	025472	000403				BR	35\$		
5099									
5100	025474	012737	000001	003252	34\$:	MOV	#1,P1.BIT		
5101	025502	004737	036302		35\$:	JSR	PC,WRTBIT		:SIMULATE BIT
5102	025506	104002				ERROR	2		
5103	025510	016237	000032	003174		MOV	RKECPT(R2),T.ECPT		:STORE ECC WORD
5104	025516	004737	036134			JSR	PC,ECCGEN		:GENERATE EXPECTED ECC PAT
5105	025522	023737	003234	003174		CMP	E.ECPT,T.ECPT		:CHECK ECC PATTERN CORRECT
5106	025530	001401				BEG	37\$:YES, CONTINUE
5107	025532	104135				ERROR	13\$:ECC PATTERN INCORRECT

H08

5108	025534	005237	003262	37\$:	INC	BITCNT	: INCREMENT BIT COUNT
5109	025540	005300			DEC	RO	: CHECK IF FINISHED WITH WORD
5110	025542	001336			BNE	32\$: NO, CONTINUE
5111	025544	005301			DEC	R1	: CHECK IF FINISHED WITH TEST
5112	025546	001331			BNE	30\$: NO, CONTINUE

 *TEST 43 ECC GENERATION (PART 2)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 * PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 * DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
 * FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
 * SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
 * AND A GOOD HEADER. CLOCK THROUGH ONLY THE FOLLOWING
 * SIX WORDS OF DATA:

177777
 177777
 177777
 177777
 177777
 177777

* CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
 * GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

 *ST43: SCOPE

5136	025550	000004			MOV	#10, \$TIMES	: DO 10. ITERATIONS
5137	025552	012737	000012	001200	MOV	\$BASE, R2	: LOAD RK611 BASE
5138	025556	013702	001270		MOV	#CCLR, RKCS1(R2)	: CLEAR RK611
5139	025564	012762	100000	000000	MOV	#DMD, RKMR1(R2)	: PUT RK611 IN DIAGNOSTIC MODE
5140	025572	012762	000040	000026	MOV	#ECC3, RKBA(R2)	: LOAD ECC DATA
5141	025600	012762	053552	000004	MOV	#-12, RKWC(R2)	: WORD COUNT=12
5142	025606	012762	177766	000002	MOV	#WRDATA, RKCS1(R2)	: ISSUE WRITE DATA
5143	025614	012762	000023	000000	MOV	#10.*37., RO	: ISSUE ENOUGH CLOCKS UNTIL : READY FOR SECTOR PULSE
5145							
5146	025626	012762	000440	000026	1\$: MOV	#DMD!MCLK, RKMR1(R2)	
5147	025634	012762	000040	000026	MOV	#DMD, RKMR1(R2)	
5148	025642	005300			DEC	RO	
5149	025644	001370			BNE	1\$	
5150	025646	012762	000140	000026	MOV	#DMD!MSP, RKMR1(R2)	: SIMULATE SECTOR PULSE
5151	025654	012762	000040	000026	MOV	#DMD, RKMR1(R2)	
5152	025662	005037	003254		CLR	PR.BIT	: GENERATE SYNCH
5153	025666	005037	003256		CLR	M1.BIT	
5154	025672	012700	000377		MOV	#255, RO	
5155	025676	004737	037030		5\$: JSR	PC, RDBIT	
5156	025702	005300			DEC	RO	: CHECK IF SYNCH FINISHED
5157	025704	001374			BNE	5\$	
5158	025706	012737	000001	003254	MOV	#1, PR.BIT	: SIMULATE SYNCH BIT
5159	025714	004737	037030		JSR	PC, RDBIT	
5160	025720	012701	000003		MOV	#3, R1	: SIMULATE HEADER
5161	025724	012703	053366		MOV	#HEAD1, R3	: CYL 0, TRK 0, SECTOR 0
5162	025730	012304			12\$: MOV	(R3)+, R4	: GET NEXT HEADER WORD
5163	025732	012700	000020		MOV	#16., RO	: LOAD BITS PER WORD

```

5164 025736 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5165 025744 006004 ROR R4 ;GET NEXT BIT
5166 025746 103403 BCS 17$
5167 025750 005037 003254 CLR PR.BIT
5168 025754 000403 BR 18$
5169
5170 025756 012737 000001 003254 17$: MOV #1,PR.BIT
5171 025764 004737 037030 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5172 025770 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
5173 025772 001361 BNE 15$ ;NO, CONTINUE
5174 025774 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5175 025776 001354 BNE 12$ ;NO, CONTINUE
5176 026000 012700 000101 MOV #65,RO ;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5177 026004 013737 003254 003256 20$: MOV PR.BIT,M1.BIT
5178 026012 005037 003254 CLR PR.BIT
5179 026016 004737 037030 JSR PC,RDBIT
5180 026022 005300 DEC RO ;CHECK IF GAP FINISHED
5181 026024 001367 BNE 20$ ;NO, CONTINUE
5182 026026 012700 000400 MOV #256,RO ;LOAD COUNT FOR SYNCH FIELD
5183 026032 012737 050672 001310 MOV #EM327,EMW ;LOAD ERROR
5184 026040 005037 003252 CLR P1.BIT ;INITIALIZE BITS
5185 026044 005037 003254 CLR PR.BIT
5186 026050 005037 003256 CLR M1.BIT
5187 026054 005037 003260 CLR M2.BIT
5188 026060 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5189 026064 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5190 026072 004737 036302 25$: JSR PC,WRTBIT ;WRITE BIT
5191 026076 104002 ERROR 2 ;DATA INCORRECT
5192 026100 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5193 026104 005300 DEC RO ;CHECK IF FINISHED
5194 026106 001371 BNE 25$ ;NO, CONTINUE
5195 026110 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
5196 026116 004737 036302 JSR PC,WRTBIT
5197 026122 104002 ERROR 2
5198 026124 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5199 026130 012737 050744 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5200 026136 005037 003266 CLR ECCHI ;INITIALIZE ECC
5201 026142 005037 003270 CLR ECCL0
5202 026146 012703 053552 MOV #ECC3,R3 ;LOAD START OF DATA
5203 026152 012701 000006 MOV #6,R1 ;LOAD COUNT
5204 026156 012304 30$: MOV (R3)+,R4 ;GET NEXT BIT
5205 026160 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
5206 026164 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5207 026172 013737 003254 003256 MOV PR.BIT,M1.BIT
5208 026200 013737 003252 003254 MOV P1.BIT,PR.BIT
5209 026206 006004 ROR R4 ;GET NEXT BIT
5210 026210 103403 BCS 34$
5211 026212 005037 003252 CLR P1.BIT
5212 026216 000403 BR 35$
5213
5214 026220 012737 000001 003252 34$: MOV #1,P1.BIT
5215 026226 004737 036302 35$: JSR PC,WRTBIT ;SIMULATE BIT
5216 026232 104002 ERROR 2
5217 026234 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
5218 026242 004737 036134 JSR PC,ECCGEN ;GENERATE EXPECTED ECC PAT
5219 026246 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
    
```


5220	026254	001401			BEQ	37\$:YES, CONTINUE
5221	026256	104135			ERROR	135	:ECC PATTERN INCORRECT
5222	026260	005237	003262	37\$:	INC	BITCNT	:INCREMENT BIT COUNT
5223	026264	005300			DEC	R0	:CHECK IF FINISHED WITH WORD
5224	026266	001336			BNE	32\$:NO, CONTINUE
5225	026270	005301			DEC	R1	:CHECK IF FINISHED WITH TEST
5226	026272	001331			BNE	30\$:NO, CONTINUE

 :TEST 44 ECC WRITING
 *

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 DATA OF 400 WORDS. TO AN RK06 IN 25 SECTOR FORMAT,
 CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
 AND DRIVE CLEAR MESSAGE. SIMULATE A SECTOR PULSE
 AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS
 AND THE TWO ECC WORDS. MAKE SURE THE TWO ECC WORDS
 ARE CORRECT AND WRITTEN PROPERLY. CHECK BUS ADDRESS,
 WORD COUNT, CYLINDER, TRACK, AND SECTOR.

5242	026274	000004			ST44:	SCOPE		
5243	026276	012737	000012	001200	MOV	#10., \$TIMES	::DO 10. ITERATIONS	
5245	026304	013702	001270		MOV	\$BASE, R2	:LOAD RK611 BASE	
5246	026310	012762	100000	000000	MOV	#CCLR, RKCS1(R2)	:CLEAR RK611	
5247	026316	012762	000040	000026	MOV	#DMD, RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE	
5248	026324	012762	056220	000004	MOV	#ECCBUF, RKBA(R2)	:LOAD ADDRESS	
5249	026332	012762	177400	000002	MOV	#-400, RKWC(R2)	:WORD COUNT=400	
5250	026340	012762	000023	000000	MOV	#WRDATA, RKCS1(R2)	:ISSUE WRITE DATA	
5251	026346	012700	004612		MOV	#66.*37., R0	:ISSUE ENOUGH CLOCKS UNTIL : READY FOR SECTOR PULSE	
5253	026352	012762	000440	000026	15:	MOV	#DMD!MCLK, RKMR1(R2)	
5254	026360	012762	000040	000026	MOV	#DMD, RKMR1(R2)		
5255	026366	005300			DEC	R0		
5256	026370	001370			BNE	15		
5257	026372	012762	000140	000026	MOV	#DMD!MSP, RKMR1(R2)	:SIMULATE SECTOR PULSE	
5258	026400	012762	000040	000026	MOV	#DMD, RKMR1(R2)		
5259	026406	005037	003254		CLR	PR.BIT		
5260	026412	005037	003256		CLR	M1.BIT		
5261	026416	012700	000377		MOV	#255., R0		
5262	026422	004737	037030		5\$:	JSR	PC, RDBIT	
5263	026426	005300			DEC	R0	:CHECK IF SYNCH FINISHED	
5264	026430	001374			BNE	5\$		
5265	026432	012737	000001	003254	MOV	#1. PR.BIT	:SIMULATE SYNCH	
5266	026440	004737	037030		JSR	PC, RDBIT		
5267	026444	012703	053366		MOV	#HEAD1, R3	:LOAD HEADER	
5268	026450	012701	000003		MOV	#3, R1	:LOAD WORDS PER HEADER	
5269	026454	012304			10\$:	MOV	(R3)+, R4	:GET NEXT WORD
5270	026456	012700	000020		MOV	#16., R0	:LOAD BITS PER	
5271	026462	013737	003254	003256	12\$:	MOV	PR.BIT, M1.BIT	:STORE PREVIOUS BIT
5272	026470	006004			ROR	R4	:GET NEXT BIT	
5273	026472	103403			BCS	15\$		
5274	026474	005037	003254		CLR	PR.BIT		
5275	026500	000403			BR	16\$		


```

5276
5277 026502 012737 000001 003254 15$: MOV #1,PR.BIT
5278 026510 004737 037030 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5279 026514 005300 DEC RO ;CHECK IF READY FOR NEXT WORD
5280 026516 001361 BNE 12$ ;NO, CONTINUE
5281 026520 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5282 026522 001354 BNE 10$ ;NO, CONTINUE
5283 026524 012700 000101 MOV #65,RO
5284 026530 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5285 026536 005037 003254 CLR PR.BIT
5286 026542 004737 037030 JSR PC,RDBIT
5287 026546 005300 DEC RO ;CHECK IF GAP FINISHED
5288 026550 001367 BNE 20$ ;NO, CONTINUE
5289 026552 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH
5290 026556 005037 003254 CLR PR.BIT
5291 026562 005037 003256 CLR M1.BIT
5292 026566 005037 003260 CLR M2.BIT
5293 026572 012737 050672 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5294 026600 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5295 026604 012700 000400 MOV #256,RO ;LOAD SYNCH COUNT AND WRITE SYNCH
5296 026610 012737 062040 003224 1 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5297 026616 004737 036302 25$: JSR PC,WRTBIT
5298 026622 104002 ERROR 2
5299 026624 005237 003262 INC BITCNT ;CLEAR BIT COUNT
5300 026630 005300 DEC RO ;CHECK IF SYNCH FINISHED
5301 026632 001371 BNE 25$ ;NO, CONTINUE
5302 026634 012737 000001 003252 MOV #1,P1.BIT ;WRITE SYNCH BIT
5303 026642 004737 036302 JSR PC,WRTBIT
5304 026646 104002 ERROR 2
5305 026650 005037 003266 CLR ECCHI ;INITIALIZE ECC GENERATOR
5306 026654 005037 003270 CLR ECCLO
5307 026660 012737 050744 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5308 026666 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5309 026672 012701 000400 MOV #256,R1 ;LOAD WORDS PER SECTOR
5310 026676 012703 056220 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5311 026702 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5312 026704 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
5313 026710 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5314 026716 013737 003254 003256 MOV PR.BIT,M1.BIT
5315 026724 013737 003252 003254 MOV P1.BIT,PR.BIT
5316 026732 006004 ROR R4 ;DETERMINE NEXT BIT
5317 026734 103403 BCS 34$
5318 026736 005037 003252 CLR P1.BIT
5319 026742 000403 BR 35$
5320
5321 026744 012737 000001 003252 34$: MOV #1,P1.BIT
5322 026752 004737 036302 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5323 026756 104002 ERROR 2
5324 026760 004737 036134 JSR PC,ECCGEN ;GENERATE ECC
5325 026764 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET PATTERN
5326 026772 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5327 027000 001401 BEQ 37$ ;YES, CONTINUE
5328 027002 104135 ERROR 135 ;ECC PATTERN INCORRECT
5329 027004 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5330 027010 022700 000002 CMP #2,RO ;IF THE NEXT BIT IS THE LAST BIT OF
5331 027014 001006 BNE 28$ ;THE LAST DATA WORD, ECCW MUST BE RESET
    
```

```

5332 027016 022701 000001      CMP      #1,R1      ;IN THE EXPECTED MRI TO INDICATE ECC
5333 027022 001003      BNE      28$      ;BEING WRITTEN
5334 027024 042737 020000 003224      BIC      #ECCW,E.MR1
5335 027032 005300      DEC      R0      ;CHECK IF THROUGH WITH WORD
5336 027034 001325      BNE      28$      ;NO, CONTINUE
5337 027036 005301      DEC      R1      ;CHECK IF AT END OF SECTOR
5338 027040 001320      BNE      30$      ;NO, CONTINUE
5339 027042 012737 051311 001310      MOV      #EM333,EMW ;LOAD ERROR MESSAGE
5340 027050 005037 003262      CLR      BITCNT   ;INITIALIZE BIT COUNT
5341 027054 012701 000002      MOV      #2,R1    ;LOAD NUMBER OF ECC WORDS
5342 027060 012703 003266      MOV      #ECC1,R3 ;LOAD ADDRESS OF ECC
5343 027064 012304      MOV      (R3)+,R4 ;GET NEXT ECC WORD
5344 027066 012700 000020      MOV      #16,R0   ;LOAD BITS PER WORD
5345 027072 013737 003256 003260 42$:      MOV      M1,BIT,M2,BIT ;SHIFT BITS
5346 027100 013737 003254 003256      MOV      PR,BIT,M1,BIT
5347 027106 013737 003252 003254      MOV      P1,BIT,PR,BIT
5348 027114 006004      ROR      R4      ;DETERMINE NEXT BIT
5349 027116 103403      BCS      44$
5350 027120 005037 003252      CLR      P1,BIT
5351 027124 000403      BR
5352
5353 027126 012737 000001 003252 44$:      MOV      #1,P1,BIT
5354 027134 004737 036302 45$:      JSR      PC,WRTBIT ;WRITE NEXT BIT
5355 027140 104002      ERROR     2
5356 027142 005237 003262      INC      BITCNT   ;INCREMENT BIT COUNT
5357 027146 022700 000002      CMP      #2,R0   ;IF THE LAST BIT OF THE LAST ECC WORD
5358 027152 001006      BNE      46$      ;IS BEING WRITTEN, ECCW MUST BE SET
5359 027154 022701 000001      CMP      #1,R1   ;IN EXPECTED MRI TO INDICATE ECC
5360 027160 001003      BNE      46$      ;WRITING IS DONE
5361 027162 051311 020000 003224 46$:      BIS      #ECCW,E.MR1
5362 027170 005300      DEC      R0      ;CHECK IF THROUGH WITH WORD
5363 027172 001337      BNE      42$      ;NO, CONTINUE
5364 027174 005301      DEC      R1      ;CHECK IF FINISH WITH ECC
5365 027176 001332      BNE      40$      ;NO, CONTINUE
5366 027200 012737 051347 001310      MOV      #EM334,EMW ;LOAD ERROR MESSAGE
5367 027206 005037 003262      CLR      BITCNT   ;CLEAR BIT COUNT
5368 027212 012700 000017      MOV      #15,R0   ;LOAD POSTAMBLE BIT COUNT
5369 027216 013737 003256 003260 47$:      MOV      M1,BIT,M2,BIT ;SHIFT BIT
5370 027224 013737 003254 003256      MOV      PR,BIT,PR,BIT
5371 027232 013737 003252 003254      MOV      P1,BIT,PR,BIT
5372 027240 005037 003252      CLR      P1,BIT
5373 027244 004737 036302      JSR      PC,WRTBIT ;WRITE NEXT BIT
5374 027250 104002      ERROR     2
5375 027252 005237 003262      INC      BITCNT   ;INCREMENT BIT COUNT
5376 027256 005300      DEC      R0      ;CHECK IF THROUGH WITH POSTAMBLE
5377 027260 001356      BNE      47$      ;NO, CONTINUE
5378 027262 012700 000014      MOV      #3*4,R0  ;ISSUE CLOCKS TO COMPLETE COMMAND
5379 027266 012762 000440 000026 50$:      MOV      #DMD!MCLK,RKMR1(R2) ;ISSUE CLOCK PULSES
5380 027274 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5381 027302 005300      DEC      R0
5382 027304 001370      BNE      50$
5383 027306 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;SAVE CS1
5384 027314 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;SAVE CS2
5385 027322 016237 000014 003154      MOV      RKER(R2),T.ER ;SAVE ERROR REG
5386 027330 012737 000222 003200      MOV      #RDY!WRDATA&<↑<GO>>,E.CS1 ;LOAD EXPECTED CS1
5387 027336 012737 000100 003210      MOV      #IR,E.CS2 ;LOAD EXPECTED CS2

```

M08

```

5388 027344 005037 003214 CLR E.ER ;LOAD EXPECTED ERROR REG.
5389 027350 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;SAVE CYLINDER ADDR REG
5390 027356 016237 000006 003146 MOV RKDA(R2),T.DA ;SAVE DISK ADDR REG
5391 027364 016237 000004 003144 MOV RKBA(R2),T.BA ;SAVE BUS ADDR REG
5392 027372 016237 000002 003142 MOV RKWC(R2),T.WC ;SAVE WORD COUNT
5393 027400 005037 003220 CLR E.DCYL ;LOAD EXPECTED CYLINDER AND REG.
5394 027404 012737 000001 003206 MOV #1,E.DA ;LOAD EXPECTED DISK
5395 027412 012737 057220 003204 MOV #ECCBUF+<400*2>,E.BA ;LOAD EXPECTED
5396 027420 005037 003202 CLR E.WC ;LOAD EXPECTED ECC WORD
5397 027424 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1
5398 027432 001401 BEQ 55$
5399 027434 104144 ERROR 144
5400 027436 023737 003210 003150 55$: CMP E.CS2,T.CS2 ;CHECK CS2
5401 027444 001401 BEQ 56$
5402 027446 104145 ERROR 145
5403 027450 023737 003214 003154 56$: CMP E.ER,T.ER ;CHECK ERROR REG
5404 027456 001401 BEQ 57$
5405 027460 104146 ERROR 146
5406 027462 023737 003204 003144 57$: CMP E.BA,T.BA ;CHECK BUS ADDR
5407 027470 001401 BEQ 58$
5408 027472 104147 ERROR 147
5409 027474 023737 003202 003142 58$: CMP E.WC,T.WC ;CHECK WORD COUNT
5410 027502 001401 BEQ 59$
5411 027504 104150 ERROR 150
5412 027506 023737 003220 003160 59$: CMP E.DCYL,T.DCYL ;CHECK CYLINDER ADDR
5413 027514 001401 BEQ 60$
5414 027516 104151 ERROR 151
5415 027520 023737 003206 003146 60$: CMP E.DA,T.DA ;CHECK DISK ADDR.
5416 027526 001401 BEQ TST45 ;:YES, GO ON TO NEXT TEST
5417 027530 104152 ERROR 152

.SBTTL **PARTIAL WRITE DATA
:*****
: *TEST 45 ZERO FILL ON WRITE DATA
: *
: * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
: * PLT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
: * DATA OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
: * CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
: * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
: * AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS AND
: * THE TWO ECC WORDS. CHECK THE SECTOR FOR ZERO FILL
: * AND MAKE SURE THE TWO ECC WORDS ARE WRITTEN PROPERLY.
: * CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECTOR.
:*****
TST45: SCOPE
MOV #10, $TIMES ;;DO 10. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #ECCBUF,RKBA(R2) ;LOAD ADDRESS
MOV #-103,RKWC(R2) ;WORD COUNT=400
MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
    
```

N08

```

5444 027604 012700 004612      MOV      #66.*37.,R0      ;ISSUE ENOUGH CLOCKS UNTIL
5445                                     ;READY FOR SECTOR PULSE
5446 027610 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMRI(R2)
5447 027616 012762 000040 000026      MOV      #DMD,RKMRI(R2)
5448 027624 005300      DEC      R0
5449 027626 001370      BNE     1$
5450 027630 012762 000140 000026      MOV      #DMD!MSP,RKMRI(R2) ;SIMULATE SECTOR PULSE
5451 027636 012762 000040 000026      MOV      #DMD,RKMRI(R2)
5452 027644 005037 003254      CLR     PR.BIT
5453 027650 005037 003256      CLR     M1.BIT
5454 027654 012700 000377      MOV     #255.,R0
5455 027660 004737 037030      5$:    JSR     PC,RDBIT
5456 027664 005300      DEC     R0                ;CHECK IF SYNCH FINISHED
5457 027666 001374      BNE     5$
5458 027670 012737 000001 003254      MOV     #1,PR.BIT        ;SIMULATE SYNCH
5459 027676 004737 037030      JSR     PC,RDBIT
5460 027702 012703 053366      MOV     #HEAD1,R3        ;LOAD HEADER
5461 027706 012701 000003      MOV     #3,R1            ;LOAD WORDS PER HEADER
5462 027712 012304      10$:   MOV     (R3)+,R4          ;GET NEXT WORD
5463 027714 012700 000020      MOV     #16.,R0          ;LOAD BITS PER
5464 027720 013737 003254 003256 12$:   MOV     PR.BIT,M1.BIT    ;STORE PREVIOUS BIT
5465 027726 006004      ROR     R4                ;GET NEXT BIT
5466 027730 103403      BCS     15$
5467 027732 005037 003254      CLR     PR.BIT
5468 027736 000403      BR      16$
5469
5470 027740 012737 000001 003254 15$:   MOV     #1,PR.BIT
5471 027746 004737 037030 16$:   JSR     PC,RDBIT        ;SIMULATE NEXT BIT
5472 027752 005300      DEC     R0                ;CHECK IF READY FOR NEXT WORD
5473 027754 001361      BNE     12$              ;NO, CONTINUE
5474 027756 005301      DEC     R1                ;CHECK IF FINISHED WITH HEADER
5475 027760 001354      BNE     10$              ;NO, CONTINUE
5476 027762 012700 000101      MOV     #65.,R0
5477 027766 013737 003254 003256 20$:   MOV     PR.BIT,M1.BIT    ;SIMULATE GAP
5478 027774 005037 003254      CLR     PR.BIT
5479 030000 004737 037030      JSR     PC,RDBIT
5480 030004 005300      DEC     R0                ;CHECK IF GAP FINISHED
5481 030006 001367      BNE     20$              ;NO, CONTINUE
5482 030010 005037 003252      CLR     P1.BIT          ;INITIALIZE BITS FOR SYNCH
5483 030014 005037 003254      CLR     PR.BIT
5484 030020 005037 003256      CLR     M1.BIT
5485 030024 005037 003260      CLR     M2.BIT
5486 030030 012737 050672 001310      MOV     #EM327,EMW        ;LOAD ERROR MESSAGE
5487 030036 005037 003262      CLR     BITCNT          ;INITIALIZE BIT COUNT
5488 030042 012700 000400      MOV     #256.,R0          ;LOAD SYNCH COUNT AND WRITE SYNCH
5489 030046 012737 062040 003224      MOV     #DMD!ECCW!MEWD!WRTGAT,E.MRI ;SET EXPECTED MRI
5490 030054 004737 036302      25$:   JSR     PC,WRTBIT
5491 030060 104002      ERROR  2
5492 030062 005237 003262      INC     BITCNT          ;CLEAR BIT COUNT
5493 030066 005300      DEC     R0                ;CHECK IF SYNCH FINISHED
5494 030070 001371      BNE     25$              ;NO, CONTINUE
5495 030072 012737 000001 003252      MOV     #1,P1.BIT        ;WRITE SYNCH BIT
5496 030100 004737 036302      JSR     PC,WRTBIT
5497 030104 104002      ERROR  2
5498 030106 005037 003266      CLR     ECCHI            ;INITIALIZE ECC GENERATOR
5499 030112 005037 003270      CLR     ECCLO
    
```

0001	030116	012737	050744	001310		MOV	#EM328,EMW	:LOAD ERROR MESSAGE
0002	030124	005037	003262			CLR	BITCNT	:INITIALIZE BIT COUNT
0003	030130	012701	000400			MOV	#256,R1	:LOAD WORDS PER SECTOR
0004	030134	012703	056220			MOV	#ECCBUF,R3	:LOAD ADDRESS OF BUFFER
0005	030140	012304			30\$:	MOV	(R3)+,R4	:GET NEXT WORD
0006	030142	012700	000020		31\$:	MOV	#16,R0	:LOAD BITS PER WORD
0007	030146	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
0008	030154	013737	003254	003256		MOV	PR.BIT,M1.BIT	
0009	030162	013737	003252	003254		MOV	P1.BIT,PR.BIT	
0010	030170	006004				ROR	R4	:DETERMINE NEXT BIT
0011	030172	103403				BCS	34\$	
0012	030174	005037	003252			CLR	P1.BIT	
0013	030200	000403				BR	35\$	
0014	030202	012737	000001	003252	34\$:	MOV	#1,P1.BIT	
0015	030210	004737	036302		35\$:	JSR	PC,WRTBIT	:WRITE NEXT BIT
0016	030214	104002				ERROR	2	
0017	030216	004737	036134			JSR	PC,ECCGEN	:GENERATE ECC
0018	030222	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	:GET PATTERN
0019	030230	023737	003234	003174		CMP	E.ECPT,T.ECPT	:CHECK ECC PATTERN CORRECT
0020	030236	001401				BEQ	37\$:YES, CONTINUE
0021	030240	104135				ERROR	13\$:ECC PATTERN INCORRECT
0022	030242	005237	003262		37\$:	INC	BITCNT	:INCREMENT BIT COUNT
0023	030246	022700	000002			CMP	#2,R0	:IF THE NEXT BIT IS THE LAST BIT OF
0024	030252	001006				BNE	28\$:THE LAST DATA WORD, ECCW MUST BE RESET
0025	030254	022701	000001			CMP	#1,R1	:IN THE EXPECTED MARI TO INDICATE ECC
0026	030260	001003				BNE	28\$:BEING WRITTEN
0027	030262	042737	020000	003224		BIC	#ECCW,E.MARI	
0028	030270	005300			28\$:	DEC	R0	:CHECK IF THROUGH WITH WORD
0029	030272	001325				BNE	32\$:NO, CONTINUE
0030	030274	005301				DEC	R1	:CHECK IF AT END OF SECTOR
0031	030276	001405				BEQ	39\$:YES -SKIP
0032	030300	022703	056426			CMP	#ECCBUF+(103*2),R3	:CHECK IF 103 WORDS TRANSFERED
0033	030304	003315				BGT	30\$:NO - GO TO GET NEXT WORD
0034	030306	005004				CLR	R4	:ELSE CLEAR R4 FOR ZEROS
0035	030310	000714				BR	31\$:GO SIMULATE REST OF SECTOR
0036	030312	012737	051311	001310	39\$:	MOV	#EM333,EMW	:LOAD ERROR MESSAGE
0037	030320	005037	003262			CLR	BITCNT	:INITIALIZE BIT COUNT
0038	030324	012701	000002			MOV	#2,R1	:LOAD NUMBER OF ECC WORDS
0039	030330	012703	003266			MOV	#ECCHI,R3	:LOAD ADDRESS OF ECC
0040	030334	012304			40\$:	MOV	(R3)+,R4	:GET NEXT ECC WORD
0041	030336	012700	000020			MOV	#16,R0	:LOAD BITS PER WORD
0042	030342	013737	003256	003260	42\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
0043	030350	013737	003254	003256		MOV	PR.BIT,M1.BIT	
0044	030356	013737	003252	003254		MOV	P1.BIT,PR.BIT	
0045	030364	006004				ROR	R4	:DETERMINE NEXT BIT
0046	030366	103403				BCS	44\$	
0047	030370	005037	003252			CLR	P1.BIT	
0048	030374	000403				BR	45\$	
0049								
0050	030376	012737	000001	003252	44\$:	MOV	#1,P1.BIT	
0051	030404	004737	036302		45\$:	JSR	PC,WRTBIT	:WRITE NEXT BIT
0052	030410	104002				ERROR	2	
0053	030412	005237	003262			INC	BITCNT	:INCREMENT BIT COUNT
0054	030416	022700	000002			CMP	#2,R0	:IF THE LAST BIT OF THE LAST ECC WORD
0055	030422	001006				BNE	46\$:IS BEING WRITTEN, ECCW MUST BE SET

5556	030424	022701	000001		CMP	#1,R1	:IN EXPECTED MR1 TO INDICATE ECC
5557	030430	001903			BNE	46\$:WRITING IS DONE
5558	030432	052737	020000	003224	BIS	#ECCW,E.MR1	
5559	030440	005300			DEC	RO	:CHECK IF THROUGH WITH WORD
5560	030442	001337			9NE	42\$:NO CONTINUE
5561	030444	005301			DEC	R1	:CHECK IF FINISH WITH ECC
5562	030446	001332			BNE	40\$:NO CONTINUE
5563	030450	012737	051347	001310	MOV	#EM334,EMW	:LOAD ERROR MESSAGE
5564	030456	005037	003262		CLR	BITCNT	:CLEAR BIT COUNT
5565	030462	012700	000017		MOV	#15,RO	:LOAD POSTAMBLE BIT COUNT
5566	030466	013737	003256	003260	MOV	M1.BIT,M2.BIT	:SHIFT BIT
5567	030474	013737	003254	003256	MOV	PR.BIT,M1.BIT	
5568	030502	013737	003252	003254	MOV	P1.BIT,PR.BIT	
5569	030510	005037	003252		CLR	P1.BIT	
5570	030514	004737	036302		JSR	PC,WRTBIT	:WRITE NEXT BIT
5571	030520	104002			ERROR	2	
5572	030522	005237	003262		INC	BITCNT	:INCREMENT BIT COUNT
5573	030526	005300			DEC	RO	:CHECK IF THROUGH WITH POSTAMBLE
5574	030530	001356			BNE	47\$:NO CONTINUE
5575	030532	012700	000014		MOV	#3*4,RO	:ISSUE CLOCKS TO COMPLETE COMMAND
5576	030536	012762	000440	000026	MOV	#DMD!MCLK,RKMR1(R2)	:ISSUE CLOCK PULSES
5577	030544	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
5578	030552	005300			DEC	RO	
5579	030554	001370			BNE	50\$	
5580	030556	016237	000000	00314C	MOV	RKCS1(R2),T.CS1	:SAVE CS1
5581	030564	016237	000010	003150	MOV	RKCS2(R2),T.CS2	:SAVE CS2
5582	030572	016237	000014	003154	MOV	RKER(R2),T.ER	:SAVE ERROR REG
5583	030600	012737	000222	003200	MOV	#RDY!WRDATA&<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
5584	030606	012737	000100	003210	MOV	#IR,E.CS2	:LOAD EXPECTED CS2
5585	030614	005037	003214		CLR	E.ER	:LOAD EXPECTED ERROR REG.
5586	030620	016237	000320	003160	MOV	RKDCYL(R2),T.DCYL	:SAVE CYLINDER ADDR REG
5587	030626	016237	000006	003146	MOV	RKDA(R2),T.DA	:SAVE DISK ADDR REG
5588	030634	016237	000004	003144	MOV	RKBA(R2),T.BA	:SAVE BUS ADDR REG
5589	030642	016237	000002	003142	MOV	RKWC(R2),T.WC	:SAVE WORD COUNT
5590	030650	005037	003220		CLR	E.DCYL	:LOAD EXPECTED CYLINDER AND REG.
5591	030654	012737	000001	003206	MOV	#1,E.DA	:LOAD EXPECTED DISK
5592	030662	012737	056426	003204	MOV	#ECCBUF+<103*2>,E.BA	:LOAD EXPECTED
5593	030670	005037	003202		CLR	E.WC	:LOAD EXPECTED ECC WORD
5594	030674	023737	003200	003140	CMP	E.CS1,T.CS1	:CHECK CS1
5595	030702	001401			BEQ	55\$	
5596	030704	104154			ERROR	154	
5597	030706	023737	003210	003150	CMP	E.CS2,T.CS2	:CHECK CS2
5598	030714	001401			BEQ	56\$	
5599	030716	104155			ERROR	155	
5600	030720	023737	003214	003154	CMP	E.ER,T.ER	:CHECK ERROR REG
5601	030726	001401			BEQ	57\$	
5602	030730	104156			ERROR	156	
5603	030732	023737	003204	003144	CMP	E.BA,T.BA	:CHECK BUS ADDR
5604	030740	001401			BEQ	58\$	
5605	030742	104157			ERROR	157	
5606	030744	023737	003202	003142	CMP	E.WC,T.WC	:CHECK WORD COUNT
5607	030752	001401			BEQ	59\$	
5608	030754	104160			ERROR	160	
5609	030756	023737	003220	003160	CMP	E.DCYL,T.DCYL	:CHECK CYLINDER ADDR
5610	030764	001401			BEQ	60\$	
5611	030766	104161			ERROR	161	

```

5612 030770 033737 003206 003146 60$: CMP E,DA,T,DA ;CHECK DISK ADDR.
5613 030776 001401 BEQ TST46 ;:YES, GO ON TO NEXT TEST
5614 031000 104162 ERROR 162
5615 .SBTTL **18 BIT FORMAT WRITES
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629 031002 000004
5630 031004 012737 000012 001200 TST46: SCOPE
5631
5632 031012 013702 001270 MOV #10.,$TIMES ;:DO 10. ITERATIONS
5633 031016 012762 100000 000000 MOV $BASE,R2 ;LOAD RK611 BASE
5634 031024 012700 002000 MOV #CLR,RKCS1(R2) ;CLEAR CONTROLLER
5635 031030 005300 5$: MOV #2000,R0 ;SET A STALL COUNT
5636 031032 001376 BNE 5$ ;LOOP UNTIL COUNT 0
5637
5638 031034 012762 000040 000026 MOV #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5639 031042 012762 055204 000004 MOV #MOD2BF,RKBA(R2) ;LOAD BUFFER ADDRESS
5640 031050 012762 177400 000002 MOV #-400,RKWC(R2) ; --- WORD COUNT
5641 031056 012762 010023 000000 MOV #WRDATA!CFMT,RKCS1(R2) ; --- START COMMAND
5642 031064 012700 005136 MOV #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5643 ; READY FOR SECTOR PULSE
5644 031070 012762 000440 000026 7$: MOV #DMD!MCLK,RKMR1(R2)
5645 031076 012762 000040 000026 MOV #DMD,RKMR1(R2)
5646 031104 005300 DEC R0
5647 031106 001370 BNE 7$
5648
5649 031110 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5650 031116 012762 000040 000026 MOV #DMD,RKMR1(R2)
5651
5652 031124 005037 003254 CLR PR.BIT ;GENERATE SYNC
5653 031130 005037 003256 CLR M1.BIT
5654 031134 012700 000377 MOV #255.,R0
5655
5656 031140 004737 037030 9$: JSR PC,RDBIT ;SIMULATE SYNC 0 BITS
5657 031144 005300 DEC R0
5658 031146 001374 BNE 9$
5659
5660 031150 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNC 1 BIT
5661 031156 004737 037030 JSR PC,RDBIT
5662
5663 031162 012703 053504 MOV #HEAD11,R3 ;SIMULATE HEADER
5664 031166 012701 000003 MOV #3,R1 ; CYL 0 TRK 0, SEC 0
5665 031172 012304 11$: MOV (R3)+,R4 ;GET HEADER WORD
5666 031174 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
5667 031200 013737 003254 13$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT

```


E09

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 108
 DZR6DA.P11 28-SEP-76 15:50 T46 18 BIT WRITE DATA (PART 1)

SEG 0108

5668	031206	006004				ROR	R4	:GET NEXT BIT
5669	031210	103403				BCS	15\$	
5670	031212	005037	003254			CLR	PR.BIT	
5671	031216	000403				BR	16\$	
5672								
5673	031220	012737	000001	003254	15\$:	MOV	#1,PR.BIT	
5674	031226	004737	037030		16\$:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
5675	031232	005300				DEC	RO	:READY FOR NEXT HEADER WORD?
5676	031234	001361				BNE	13\$:NO - GET NEXT BIT THIS WORD
5677	031236	005301				DEC	R1	:HEADER DONE?
5678	031240	001354				BNE	11\$:NO - GET NEXT HEADER WORD
5679	031242	012700	000101			MOV	#65,RO	:SET COUNT FOR GAP.
5680	031246	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT	:SIMULATE GAP
5681	031254	005037	003254			CLR	PR.BIT	
5682	031260	004737	037030			JSR	PC,RDBIT	
5683	031264	005300				DEC	RO	
5684	031266	001367				BNE	20\$	
5685	031270	012700	000400			MOV	#256,RO	:SET COUNT FOR WRITE DATA SYNC
5686	031274	012737	050672	001310		MOV	#EM327,EMW	:LOAD ERROR MESSAGE
5687	031302	005037	003252			CLR	P1.BIT	:CLEAR BITS
5688	031306	005037	003254			CLR	PR.BIT	
5689	031312	005037	003256			CLR	M1.BIT	
5690	031316	005037	003260			CLR	M2.BIT	
5691	031322	005037	003262			CLR	BITCNT	:CLEAR BIT COUNTER
5692	031326	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WRTGAT,E.MR1	:SET EXPECTED MRI
5693	031334	004737	036302		22\$:	JSR	PC,WRTBIT	:SIMULATE SYNC 0
5694	031340	104002				ERROR	2	
5695	031342	005237	003262			INC	BITCNT	:BUMP BIT COUNT
5696	031346	005300				DEC	RO	:LOOP UNTIL SYNC 0 WRITTEN
5697	031350	001371				BNE	22\$	
5698	031352	012737	000001	003252		MOV	#1,P1.BIT	:SIMULATE SYNC 1
5699	031360	004737	036302			JSR	PC,WRTBIT	
5700	031364	104002				ERROR	2	
5701	031366	005037	003266			CLR	ECCHI	:INITIALIZE ECC WORDS.
5702	031372	005037	003270			CLR	ECCLO	
5703	031376	005037	003262			CLR	BITCNT	:CLEAR BIT COUNT
5704	031402	012703	055204			MOV	#MOD2BF,R3	:SET DATA POINTER
5705	031406	012701	000006			MOV	#6,R1	:SET DATA COUNT
5706	031412	012304		24\$:		MOV	(R3)+,R4	:GET DATA WORD
5707	031414	012737	051615	001310		MOV	#EM338,EMW	:LOAD MESSAGE (18 BIT DATA WRITE
5708	031422	012700	000022			MOV	#18,RO	:LOAD BITS PER WORD
5709	031426	013737	003256	003260	25\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
5710	031434	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5711	031442	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5712	031450	000241				CLC		:CLEAR CARRY & GET NEXT BIT
5713	031452	006004				ROR	R4	
5714	031454	103403				BCS	27\$	
5715	031456	005037	003252			CLR	P1.BIT	
5716	031462	000403				BR	28\$	
5717								
5718	031464	012737	000001	003252	27\$:	MOV	#1,P1.BIT	
5719	031472	004737	036302		28\$:	JSR	PC,WRTBIT	:SIMULATE NEXT BIT
5720	031476	104002				ERROR	2	
5721	031500	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	:GET ECC PATTERN
5722	031506	004737	036134			JSR	PC,ECCGEN	:COMPUTE EXPECTED ECC
5723	031512	023737	003174	003234		CMP	T.ECPT,E.ECPT	:CHECK IF CORRECT

```

5724 031520 001401 BEQ 29$ ;YES - SKIP
5725 031522 104164 ERROR 164
5726 031524 005237 003262 29$: INC BITCNT
5727 031530 005300 DEC RO
5728 031532 020027 000002 CMP RO, #2 ;:ST 16 BITS JUST DONE?
5729 031536 001403 BEQ 31$ ;YES - SKIP
5730 031540 005700 TST RO ;ELSE TEST IF WORD DONE
5731 031542 001331 BNE 25$ ;NO - DO NEXT BIT
5732 031544 000406 BR 32$ ;ELSE DO NEXT WORD
5733 031546 012737 051703 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5734 031554 012704 000000 MOV #0,R4 ;SET UPPER BITS OF WORD
5735 031560 000722 BR 25$ ;GO DO BITS
5736
5737 031562 005301 32$: DEC R1 ;ALL WORDS DONE?
5738 031564 001312 BNE 24$ ;NO - GET NEXT WORD

```

```

*****
:TEST 47 18 BIT WRITE DATA (PART 2)

```

```

:
: CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
: PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
: OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
: CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
: AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
: AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777
: WITH BAD PARITY SET. VERIFY WRITING OF TWO PARITY
: BITS ON SIMULATED DISK.

```

```

: NOTE: THIS TEST IS ONLY EXECUTED IF MEMORY
: PARITY ENABLE IS PRESENT FOR BUFFER
: LOCATION.

```

```

5756
5757 031566 000004 ST47: SCOPE
5758 031570 012737 000012 001200 MOV #10, $TIMES ;:DO 10. ITERATIONS
5759
5760 031576 005737 003274 TST MEMPAR ;CHECK IF MEMORY PARITY AVAIL.
5761 031602 001017 BNE 25$ ;YES - SKIP
5762 031604
5763 031604 012737 000001 001200 1$: MOV #1, $TIMES ;FORCE INTERATION COUNT TO 1
5764 031612 005227 177777 INC #-1 ;ONLY DO ONCE
5765 031616 001007 BNE 64$ ;NO, GO TO NEXT TEST
5766 031620 104401 044043 TYPE TSTBY1 ;TYPE TEST N BYPASSED
5767 031624 013746 001220 MOV $TESTN, -(SP) ;:SAVE $TESTN FOR TYPEOUT
5768 031630 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5769 031632 104401 044053 TYPE TSTBY2
5770 031636 000137 032440 64$: JMP TST50 ;GO TO NEXT TEST
5771
5772 031642 004237 034116 2$: JSR R2, WTRPAR
5773 031646 000006 6
5774 031650 057220 BADPAR
5775 031652 005700 TST RO
5776 031654 001353 BNE 1$
5777 031656 013737 031664 001110 MOV 3$, $LPERP
5778
5779 031664 3$:

```

```

5780
5781 031664 013702 001270      MOV    $BASE,R2      ;LOAD RK611 BASE
5782 031670 012762 100000 000000  MOV    #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5783 031676 012700 002000      MOV    #2000,R0      ;SET A STALL COUNT
5784 031702 005300      55:   DEC    R0            ;LOOP UNTIL COUNT 0
5785 031704 001376      BNE    55
5786
5787 031706 012762 000040 000026  MOV    #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5788 031714 012762 057220 000004  MOV    #BADPAR,RKBA(R2) ;LOAD BUFFER ADDRESS
5789 031722 012762 177400 000002  MOV    #-400,RKWC(R2)  ; --- WORD COUNT
5790 031730 012762 010023 000000  MOV    #WRDATA:CFMT,RKCS1(R2) ; --- START COMMAND
5791 031736 012700 005136      MOV    #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE
5792
5793 031742 012762 000440 000026  75:   MOV    #DMD!MCLK,RKMR1(R2)
5794 031750 012762 000040 000026  MOV    #DMD,RKMR1(R2)
5795 031756 005300      DEC    R0
5796 031760 001370      BNE    75
5797
5798 031762 012762 000140 000026  MOV    #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5799 031770 012762 000040 000026  MOV    #DMD,RKMR1(R2)
5800
5801 031776 005037 003254      CLR    PR.BIT        ;GENERATE SYNC
5802 032002 005037 003256      CLR    M1.BIT
5803 032006 012700 000377      MOV    #255.,R0
5804
5805 032012 004737 037030      95:   JSR    PC,RDBIT      ;SIMULATE SYNC 0 BITS
5806 032016 005300      DEC    R0
5807 032020 001374      BNE    95
5808
5809 032022 012737 000001 003254  MOV    #1,PR.BIT     ;SIMULATE SYNC 1 BIT
5810 032030 004737 037030      JSR    PC,RDBIT
5811
5812 032034 012703 053504      MOV    #HEAD11,R3    ;SIMULATE HEADER
5813 032040 012701 000003      MOV    #3,R1         ;CYL 0 TRK 0 SEC 0
5814 032044 012304      115:  MOV    (R3)+,R4      ;GET HEADER WORD
5815 032046 012700 000020      MOV    #16.,R0      ;LOAD BITS PER WORD
5816 032052 013737 003254 003256  135:  MOV    PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5817 032060 006004      ROR    R4            ;GET NEXT BIT
5818 032062 103403      BCS    155
5819 032064 005037 003254      CLR    PR.BIT
5820 032070 000403      BR     165
5821
5822 032072 012737 000001 003254  155:  MOV    #1,PR.BIT
5823 032100 004737 037030  165:  JSR    PC,RDBIT     ;SIMULATE NEXT BIT
5824 032104 005300      DEC    R0            ;READY FOR NEXT HEADER WORD?
5825 032106 001361      BNE    135          ;NO - GET NEXT BIT THIS WORD
5826 032110 005301      DEC    R1            ;HEADER DONE?
5827 032112 001354      BNE    115          ;NO - GET NEXT HEADER WORD
5828 032114 012700 000101      MOV    #65.,R0      ;SET COUNT FOR GAP.
5829 032120 013737 003254 003256  205:  MOV    PR.BIT,M1.BIT ;SIMULATE GAP
5830 032126 005037 003254      CLR    PR.BIT
5831 032132 004737 037030      JSR    PC,RDBIT
5832 032136 005300      DEC    R0
5833 032140 001367      BNE    205
5834 032142 012700 000400      MOV    #256.,R0     ;SET COUNT FOR WRITE DATA SYNC
5835 032146 012737 050672 001310  MOV    #EM327,EMW    ;LOAD ERROR MESSAGE
    
```

H09

```

5836 032154 005037 003252 CLR P1.BIT ;CLEAR BITS
5837 032160 005037 003254 CLR PR.BIT
5838 032164 005037 003256 CLR M1.BIT
5839 032170 005037 003260 CLR M2.BIT
5840 032174 005037 003262 CLR BITCNT ;CLEAR BIT COUNTER
5841 032200 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5842 032206 004737 036302 22$: JSR PC,WRTBIT ;SIMULATE SYNC 0
5843 032212 104002 ERROR 2
5844 032214 005237 003262 INC BITCNT ;BUMP BIT COUNT
5845 032220 005300 DEC R0 ;LOOP UNTIL SYNC 0 WRITTEN
5846 032222 001371 BNE 22$
5847 032224 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNC 1
5848 032232 004737 036302 JSR PC,WRTBIT
5849 032236 104002 ERROR 2
5850 032240 005037 003266 CLR ECCHI ;INITIALIZE ECC WORDS.
5851 032244 005037 003270 CLR ECCL0
5852 032250 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5853 032254 012703 055204 MOV #MOD2BF,R3 ;SET DATA POINTER
5854 032260 012701 000006 MOV #6,R1 ;SET DATA COUNT
5855 032264 012304 24$: MOV (R3)+,R4 ;GET DATA WORD
5856 032266 012737 051615 001310 MOV #EM338,EMW ;LOAD MESSAGE (18 BIT DATA WRITE)
5857 032274 012700 000022 MOV #18,R0 ;LOAD BITS PER WORD
5858 032300 013737 003256 003260 25$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5859 032306 013737 003254 003256 MOV PR.BIT,M1.BIT
5860 032314 013737 003252 003254 MOV P1.BIT,PR.BIT
5861 032322 000241 CLC ;CLEAR CARRY & GET NEXT BIT
5862 032324 006004 ROR R4
5863 032326 103403 BCS 27$
5864 032330 005037 003252 CLR P1.BIT
5865 032334 000403 BR 28$
5866
5867 032336 012737 000001 003252 27$: MOV #1,P1.BIT
5868 032344 004737 036302 28$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5869 032350 104002 ERROR 2
5870 032352 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET ECC PATTERN
5871 032360 004737 036134 JSR PC,ECCGEN ;COMPUTE EXPECTED ECC
5872 032364 023737 003174 003234 CMP T.ECPT,E.ECPT ;CHECK IF CORRECT
5873 032372 001401 BEQ 29$ ;YES - SKIP
5874 032374 104164 ERROR 164
5875 032376 005237 003262 29$: INC BITCNT
5876 032402 005300 DEC R0
5877 032404 020027 000002 CMP R0,#2 ;1ST 16 BITS JUST DONE?
5878 032410 001403 BEQ 31$ ;YES - SKIP
5879 032412 005700 TST R0 ;ELSE TEST IF WORD DONE
5880 032414 001331 BNE 25$ ;NO - DO NEXT BIT
5881 032416 000406 BR 32$ ;ELSE DO NEXT WORD
5882 032420 012737 051703 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5883 032426 012704 000002 MOV #2,R4 ;SET UPPER BITS OF WORD
5884 032432 000722 BR 25$ ;GO DO BITS
5885
5886 032434 005301 32$: DEC R1 ;ALL WORDS DONE?
5887 032436 001312 BNE 24$ ;NO - GET NEXT WORD
5888
5889
5890
5891

```

*TEST 50 CLEAR BAD PARITY BUFFER

5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947

032440 000004
032442 012737 000001 001200
032450 012700 000006
032454 012701 057220
032460 012721 177777
032464 005300
032466 001374

032470 000004
032472 012737 000012 001200

032500 013702 001270
032504 012762 100000 000000
032512 012700 002000
032516 005300
032520 001376

032522 012762 000040 000026
032530 012762 056220 000004
032536 012762 177400 000002
032544 012762 010023 000000
032552 012700 005136

032556 012762 000440 000026
032564 012762 000040 000026
032572 005300
032574 001370

032576 012762 000140 000026
032604 012762 000040 000026

032612 005037 003254
032616 005037 003256
032622 012700 000377

032626 004737 037030
032632 005300
032634 001374

```

;*
;* THE SOLE PURPOSE OF THIS TEST IS TO CLEAR BADPAR
;* BUFFER AND REMOVE THE BAD PARITY.
;*
*****
T50: SCOPE
MOV #1, $TIMES ;; DO 1 ITERATION
MOV #6, R0
MOV #BADPAR, R1
IS: MOV #-1, (R1)+
DEC R0
BNE IS
*****
TEST 51 18 BIT WRITE DATA (PART 3)
;*
;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
;* OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
;* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEE;
;* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
;* AND A GOOD HEADER. CLOCK THROUGH 400 18 BIT WORDS
;* AND THE 32 BIT ECC. VERIFY THAT THE ECC IS
;* WRITTEN CORRECTLY.
*****
T51: SCOPE
MOV #10, $TIMES ;; DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
MOV #CLR, RKCS1(R2) ;CLEAR CONTROLLER
MOV #2000, R0 ;SET A STALL COUNT
5$: DEC R0 ;LOOP UNTIL COUNT 0
BNE 5$
MOV #DMD, RKMRI(R2) ;SET DIAGNOSTIC MODE
MOV #ECCBUF, RKBA(R2) ;LOAD BUFFER ADDRESS
MOV #-400, RKWC(R2) ; --- WORD COUNT
MOV #WRDATA:CFMT, RKCS1(R2) ; --- START COMMAND
MOV #66.*40.+<4.*3.+2>, R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE
7$: MOV #DMD!MCLK, RKMRI(R2)
MOV #DMD, RKMRI(R2)
DEC R0
BNE 7$
MOV #DMD!MSP, RKMRI(R2) ;SIMULATE SECTOR PULSE
MOV #DMD, RKMRI(R2)
CLR PR.BIT ;GENERATE SYNC
CLR M1.BIT
MOV #255., R0
9$: JSR PC, R0BIT ;SIMULATE SYNC 0 BITS
DEC R0
BNE 9$

```

```

5948
5949 032636 012737 000001 003254      MOV    #1,PR.BIT           ;SIMULATE SYNC 1 BIT
5950 032644 004737 037030      JSR    PC,RDBIT
5951
5952 032650 012703 053504      MOV    #HEAD11,R3        ;SIMULATE HEADER
5953 032654 012701 000003      MOV    #3,R1              ;CYL 0 TRK 0 SEC 0
5954 032660 012304          11$:  MOV    (R3)+,R4           ;GET HEADER WORD
5955 032662 012700 000020      MOV    #16,R0             ;LOAD BITS PER WORD
5956 032666 013737 003254 003256 13$:  MOV    PR.BIT,M1.BIT     ;STORE PREVIOUS BIT
5957 032674 006004      ROR    R4                 ;GET NEXT BIT
5958 032676 103403      BCS   15$
5959 032700 005037 003254      CLR    PR.BIT
5960 032704 000403      BR    16$
5961
5962 032706 012737 000001 003254 15$:  MOV    #1,PR.BIT
5963 032714 004737 037030 16$:  JSR    PC,RDBIT          ;SIMULATE NEXT BIT
5964 032720 005300      DEC    R0                 ;READY FOR NEXT HEADER WORD?
5965 032722 001361      BNE   13$                 ;NO - GET NEXT BIT THIS WORD
5966 032724 005301      DEC    R1                 ;HEADER DONE?
5967 032726 001354      BNE   11$                 ;NO - GET NEXT HEADER WORD
5968 032730 012700 000101      MOV    #65,R0             ;SET COUNT FOR GAP.
5969 032734 013737 003254 003256 20$:  MOV    PR.BIT,M1.BIT     ;SIMULATE GAP
5970 032742 005037 003254      CLR    PR.BIT
5971 032746 004737 037030      JSR    PC,RDBIT
5972 032752 005300      DEC    R0
5973 032754 001367      BNE   20$
5974 032756 012700 000400      MOV    #256,R0           ;SET COUNT FOR WRITE DATA SYNC
5975 032762 012737 050672 001310      MOV    #EM327,EMW        ;LOAD ERROR MESSAGE
5976 032770 005037 003252      CLR    P1.BIT           ;CLEAR BITS
5977 032774 005037 003254      CLR    PR.BIT
5978 033000 005037 003256      CLR    M1.BIT
5979 033004 005037 003260      CLR    M2.BIT
5980 033010 005037 003262      CLR    BITCNT           ;CLEAR BIT COUNTER
5981 033014 012737 062040 003224      MOV    #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5982 033022 004737 036302 003224 22$:  JSR    PC,WRTBIT        ;SIMULATE SYNC 0
5983 033026 104002      ERROR 2
5984 033030 005237 003262      INC    BITCNT           ;BUMP BIT COUNT
5985 033034 005300      DEC    R0                 ;LOOP UNTIL SYNC 0 WRITTEN
5986 033036 001371      BNE   22$
5987 033040 012737 000001 003252      MOV    #1,P1.BIT        ;SIMULATE SYNC 1
5988 033046 004737 036302      JSR    PC,WRTBIT
5989 033052 104002      ERROR 2
5990 033054 005037 003266      CLR    ECCHI           ;INITIALIZE ECC WORDS.
5991 033060 005037 003270      CLR    ECCLO
5992 033064 005037 003262      CLR    BITCNT           ;CLEAR BIT COUNT
5993 033070 012703 056220      MOV    #ECCBUF,R3       ;SET DATA POINTER
5994 033074 012701 000377      MOV    #255,R1          ;SET DATA COUNT
5995 033100 012304          24$:  MOV    (R3)+,R4         ;GET DATA WORD
5996 033102 012737 051615 001310      MOV    #EM338,EMW       ;LOAD MESSAGE (18 BIT DATA WRITE)
5997 033110 012700 000022      MOV    #18,R0           ;LOAD BITS PER WORD
5998 033114 013737 003256 003260 25$:  MOV    M1.BIT,M2.BIT    ;SHIFT BITS
5999 033122 013737 003254 003256      MOV    PR.BIT,M1.BIT
6000 033130 013737 003252 003254      MOV    P1.BIT,PR.BIT
6001 033136 000241      CLC
6002 033140 006004      ROR    R4
6003 033142 103403      BCS   27$

```


6060	033424	012737	051703	001310	425:	MOV	#EM339,EMW	;GET MESSAGE FOR BITS 16 AND 17
6061	033432	012704	000000			MOV	#0,R4	;SET UP UPPER BITS
6062	033436	000714				BR	34\$;GO DO LAST TWO BITS
6063	033440	012737	051311	001310	455:	MOV	#EM333,EMW	;MESSAGE (ECC WRITE)
6064	033446	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
6065	033452	012703	003266			MOV	#ECCHI,R3	;LOAD POINTER TO ECC WORDS
6066	033456	012701	000002			MOV	#2,R1	;SET FOR 2 WORDS
6067	033462	012304			435:	MOV	(R3)+,R4	;GET ECC WORD
6068	033464	012700	000020			MOV	#16,R0	;SET BIT COUNT
6069	033470	013737	003256	003260	445:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6070	033476	013737	003254	003256		MOV	PR.BIT,M1.BIT	
6071	033504	013737	003252	003254		MOV	P1.BIT,PR.BIT	
6072	033512	006004				ROR	R4	;LOAD NEXT BIT
6073	033514	103403				BCS	46\$	
6074	033516	005037	003252			CLR	P1.BIT	
6075	033522	000403				BR	47\$	
6076								
6077	033524	012737	000001	003252	465:	MOV	#1,P1.BIT	
6078	033532	004737	036302		475:	JSR	PC,WRTBIT	;SIMULATE NEXT BIT
6079	033536	104002				ERROR	2	
6080	033540	022700	000002			CMP	#2,R0	;IF THIS BIT IS THE LAST ECC BIT
6081	033544	001006				BNE	49\$;ECCW MUST BE SET TO INDICATE
6082	033546	022701	000001			CMP	#1,R1	;ECC IS DONE
6083	033552	001003				BNE	49\$	
6084	033554	052737	020000	003224		BIS	#ECCW,E.MR1	
6085	033562	005237	003262		495:	INC	BITCNT	;BUMP BIT COUNT
6086	033566	005300				DEC	R0	;TEST IF LAST BIT THIS WORD IS DONE
6087	033570	001337				BNE	44\$;NO - LOOP
6088	033572	005301				DEC	R1	;TEST IF BOTH WORDS WRITTEN
6089	033574	001332				BNE	43\$;NO - LOOP
6090	033576	012737	051347	001310		MOV	#EM334,EMW	;LOAD MESSAGE (POSTAMBLE)
6091	033604	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
6092	033610	012700	000017			MOV	#15,R0	;SET GAP COUNT
6093	033614	013737	003256	003260	515:	MOV	M1.BIT,M2.BIT	;SHIFT REST OF BITS
6094	033622	013737	003254	003256		MOV	PR.BIT,M1.BIT	
6095	033630	013737	003252	003254		MOV	P1.BIT,PR.BIT	
6096	033636	005037	003252			CLR	P1.BIT	;CLEAR NEXT BIT
6097	033642	004737	036302			JSR	PC,WRTBIT	;SIMULATE BIT
6098	033646	104002				ERROR	2	
6099	033650	005237	003262			INC	BITCNT	;BUMP BIT COUNT
6100	033654	005300				DEC	R0	;GAP WRITTEN?
6101	033656	001356				BNE	51\$;NO - LOOP
6102	033660	012737	000001	003234		MOV	#1,E.ECPT	;SET EXPECTED PATTERN
6103	033666	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	;GET '611 PATTERN
6104	033674	023737	003174	003234		CMP	T.ECPT,E.ECPT	;TEST IF EQUAL
6105	033702	001401				BEQ	56\$;YES - SKIP
6106	033704	104163				ERROR	163	;ELSE REPORT
6107	033706				565:			

```

6108          .SBTTL  END OF PASS ROUTINE
6109
6110          ::*****
6111          ::*INCREMENT THE PASS NUMBER ($PASS)
6112          ::*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYY"
6113          ::*WHERE XXXXX AND YYYY ARE DECIMAL NUMBERS
6114          ::*IF THERES A MONITOR GO TO IT
6115          ::*IF THERE ISN'T JUMP TO TST1
6116
6117          $EOP:
6118          033706          000004          SCOPE
6119          033710          005037          001102          CLR          $TSTNM          ;;ZERO THE TEST NUMBER
6120          033714          005037          001200          CLR          $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
6121          033720          005237          001222          INC          $PASS          ;;INCREMENT THE PASS NUMBER
6122          033724          042737          100000          001222          BIC          #100000,$PASS          ;;DON'T ALLOW A NEG. NUMBER
6123          033732          005327          DEC          (PC)+          ;;LOOP?
6124          033734          000001          $EOPCT: .WORD          1
6125          033736          003063          BGT          $DOAGN          ;;YES
6126          033740          012737          MOV          (PC)+,2(PC)+          ;;RESTORE COUNTER
6127          033742          000001          $ENDCT: .WORD          1
6128          033744          033734          $EOPCT
6129          033746          104401          033754          TYPE          ,65$          ;;TYPE ASCIZ STRING
6130          033752          000407          BR          64$          ;;GET OVER THE ASCIZ
6131          65$: .ASCIZ          <12><15>/END PASS #/
6132          64$:
6133          033772          013746          001222          MOV          $PASS,-(SP)          ;;SAVE $PASS FOR TYPEOUT
6134          033776          104405          TYPDS          ;;TYPE PASS NUMBER
6135          034000          104401          034006          TYPE          ,67$          ;;GO TYPE--DECIMAL ASCII WITH SIGN
6136          034004          000421          BR          66$          ;;TYPE ASCIZ STRING
6137          67$: .ASCIZ          / TOTAL ERRORS SINCE LAST REPORT /
6138          66$:
6139          034050          013746          001112          MOV          $ERTTL,-(SP)          ;;SAVE $ERTTL FOR TYPEOUT
6140          034054          104405          TYPDS          ;;TOTAL NUMBER OF ERRORS
6141          034056          104401          001211          TYPE          , $CRLF          ;;GO TYPE--DECIMAL ASCII WITH SIGN
6142          034062          005037          001112          CLR          $ERTTL          ;;TYPE CARRIAGE RETURN, LINE FEED
6143          034066          013700          000042          $GET42: MOV          2*42,RO          ;;CLEAR ERROR TOTAL
6144          034072          001405          BEQ          $DOAGN          ;;GET MONITOR ADDRESS
6145          034074          000005          RESET          ;;BRANCH IF NO MONITOR
6146          034076          004710          $ENDAD: JSR          PC,(RO)          ;;CLEAR THE WORLD
6147          034100          000240          NOP          ;;GO TO MONITOR
6148          034102          000240          NOP          ;;SAVE ROOM
6149          034104          000240          NOP          ;;FOR
6150          034106          000137          $DOAGN: JMP          2(PC)+          ;;ACT11
6151          034110          004350          $RTNAD: .WORD          TST1          ;;RETURN
6152          034112          377          377          000          $ENULL: .BYTE          -1,-1,0          ;;NULL CHARACTER STRING
6153          034116          .EVEN
6154
6155          .SBTTL  GENERATE BAD PARITY IN MEMORY
6156          ::*
6157          ::* THIS ROUTINE ATTEMPTS TO GENERATE BAD PARITY WORDS IN
6158          ::* MEMORY. THE NUMBER OF WORDS TO BE GENERATED AND THE AREA
6159          ::* WHERE THEY ARE TO BE PLACED IS SPECIFIED IN THE WORDS
6160          ::* FOLLOWING THE CALL. IF ALL THE SPECIFIED WORDS CANNOT
6161          ::* BE GENERATED, THE BUFFER IS CLEARED OF BAD PARITY WORDS.
6162
6163

```

6164			* :				
6165			* :	RO IS USED AS A FLAG TO INDICATE IF BAD PARITY WAS			
6166			* :	GENERATED. IF RO IS C WHEN THE ROUTINE RETURNS TO CALLER			
6167			* :	PARITY WAS MADE BAD IN ALL THE DESIRED WORDS. IF RO IS ALL			
6168			* :	ONES IT WAS NOT SUCCESSFUL.			
6169			* :				
6170			* :	CALL: JSR R2,WRTPAR			
6171			* :	POINTER TO AREA TO BE ALTERED			
6172			* :	NUMBER OF WORDS			
6173			* :	RETURN: RTS R2			
6174			* :	WITH RO AS SUCCESS FLAG			
6175			* :				
6176	034116	011200		WRTPAR: MOV (R2),RO			:GET BUFFER POINTER
6177	034120	016201	000002	MOV 2(R2),R1			:GET NUMBER OF WORDS
6178	034124	012703	172100	MOV #MEMBAS,R3			:GET BASE OF VECTOR AREA
6179	034130	013704	003274	MOV MEMPAR,R4			:STORE FLAGS
6180	034134	012705	000020	MOV #16.,R5			:GET NUMBER OF REGISTERS
6181	034140	006004		5\$: ROR R4			:CHECK IF PARITY ENABLE ON THIS BANK
6182	034142	103002		BCC 7\$:NO, TRY NEXT BANK
6183	034144	012713	000005	MOV #WR.PAR,(R3)			:ALLOW SETTING OF BAD PARITY
6184	034150	062703	000002	7\$: ADD #2,R3			:CALCULATE NEXT ADDRESS
6185	034154	005305		DEC R5			:CHECK IF FINISHED
6186	034156	001370		BNE 5\$:NO, GET NEXT PARITY MEMORY ADDRESS
6187	034160	012721	177777	8\$: MOV #-1,(R1)+			:WRITE BAD PARITY
6188	034164	005300		DEC RO			:LOOP UNTIL ALL LOCATIONS WRITTEN
6189	034166	001374		BNE 8\$			
6190	034170	012703	172100	MOV #MEMBAS,R3			:LOAD BASE OF CONTROL REGS
6191	034174	013704	003274	MOV MEMPAR,R4			:LOAD FLAGS
6192	034200	012705	000020	MOV #16.,R5			:GET NUMBER OF REGISTERS
6193	034204	006004		15\$: ROR R4			:CHECK IF PARITY ENABLE ON THIS BANK
6194	034206	103002		BCC 17\$			
6195	034210	012713	000001	MOV #PAR.EN,(R3)			:ALLOW PARITY DETECTION
6196	034214	052703	000002	17\$: ADD #2,R3			:CHECK IF FINISHED
6197	034220	005305		DEC R5			:CHECK IF FINISHED
6198	034222	001370		BNE 15\$:NO, GET NEXT PARITY MEMORY ADDRESS
6199	034224	012737	034374 000114	MOV #30\$,MEMVEC			:SET PARITY TRAP VECTOR
6200	034232	011200		MOV (R2),RO			:NUMBER OF WORDS
6201	034234	016201	000002	MOV 2(R2),R1			:BUFFER POINTER
6202	034240	010004		MOV RO,R4			:STORE NUMBER
6203	034242	005721		10\$: TST (R1)+			:TEST FIRST BUFFER WORD - SHOULD GET
6204	034244	000240		NOP			:TRAP WHICH DECREMENTS R4
6205	034246	000240		NOP			
6206	034250	005300		11\$: DEC RO			:DEC NUMBER OF WORDS COUNTER
6207	034252	001373		BNE 10\$:ALL CHECKED? NO - SKIP
6208	034254	005704		TST R4			:TEST IF ALL LOCATIONS BAD
6209	034256	031444		BEQ 20\$:YES - SKIP
6210	034260	011200		MOV (R2),RO			:GET NUMBER OF WORDS
6211	034262	016201	000002	MOV 2(R2),R1			:BUFFER POINTER
6212	034266	012721	000000	12\$: MOV #0,(R1)+			:CLEAR ALL OF AREA THAT WAS MADE BAD
6213	034272	005300		DEC RO			
6214	034274	001374		BNE 12\$			
6215	034276	012700	177777	MOV #-1,RO			:SET FLAG THAT COULDN'T SET BAD PARITY
6216	034302	012703	172100	13\$: MOV #MEMBAS,R3			:LOAD BASE OF CONTROL REGISTERS
6217	034306	013704	003274	MOV MEMPAR,R4			:NOW SET UP TO CLEAR PARITY
6218	034312	012705	000020	MOV #16.,R5			:CONTROL REGISTERS AND SET
6219	034316	006004		14\$: ROR R4			:ENABLE WHERE APPLICABLE

```

0220 034320 103002          BCC      18$
0221 034322 012713 000001    MOV     @PAR.EN,(R3)
0222 034326 062703 000002    18$:   ADD     @2,R3
0223 034332 005305          DEC     R5
0224 034334 001370          BNE     14$
0225 034336 022222          CMP     (R2)+,(R2)+ ;SET POINTER FOR RETURN
0226 034340 012737 034530 000114    MOV     @MEMERR, MEMVEC ;RESET TRAP VECTOR
0227 034346 012737 000340 000116    MOV     @PR7, MEMVEC+2
0228 034354 012746 000000          MOV     @PRO,-(SP) ;RESET PRIORITY TO 0
0229 034360 012746 034366          MOV     @19$,-(SP)
0230 034364 000002          RTI

0231 034366 000202          19$:   RTS     R2

0232 034370 005000          20$:   CLR     R0 ;CLEAR FLAG - BAD PARITY SET UP
0233 034372 000743          BR     13$

0234 034374 005304          30$:   DEC     R4 ;DEC THE COUNTER
0235 034376 022626          CMP     (SP)+,(SP)+ ;CLEAN STACK
0236 034400 000723          BR     11$ ;GO CHECK NEXT LOCATION

.SBTTL CHECK FOR MEMORY CHECK ENABLE
;* THIS ROUTINE CHECKS ALL AVAILABLE MEMORY FOR PARITY OPTION.
;* IF OPTION IS PRESENT, PARITY DETECT IS ENABLED ON THAT BANK.
;*
;* CALL: JSR PC,PARCHK
;* RETURN: RTS PC

0237 034402 012737 034460 000004    PARCHK: MOV     @20$,ERRVEC ;SET VECTOR FOR MEMORY PARITY CHECK
0238 034410 012737 000340 000006    MOV     @PR7,ERRVEC+2
0239 034416 005037 003274          CLR     MEMPAR ;CLEAR FLAG
0240 034422 012703 172100          MOV     @MEMBAS,R3 ;LOAD REGISTER TO DETERMINE IF
; MEMORY CHECK ENABLE AVAILBLE
0241 034426 012704 000001          MOV     @1,R4 ;INITIALIZE MASK
0242 034432 012713 000001          16$:   MOV     @PAR.EN,(R3) ;ENABLE MEMORY CHECK
0243 034436 005713          TST     (R3)
0244 034440 050437 003274          BIS     R4, MEMPAR ;SET FLAG
0245 034444 062703 000002          ADD     @2,R3 ;BUMP TO NEXT CSR
0246 034450 000241          CLC
0247 034452 006104          ROL     R4 ;CHECK IF FINISHED
0248 034454 001366          BNE     16$ ;NO, SET UP NEXT MEMORY PARITY MODLLE
0249 034456 000406          BR     22$ ;RESTORE TRAP VECTOR

0250 034460 022626          20$:   CMP     (SP)+,(SP)+ ;ADJUST STACK
0251 034462 062703 000002          ADD     @2,R3
0252 034466 000241          CLC
0253 034470 006104          ROL     R4 ;CHECK IF FINISHED
0254 034472 001357          BNE     16$ ;NO, GET NEXT LOCATION
0255 034474 012737 000006 000004    22$:   MOV     @ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
0256 034502 005037 000006          CLR     ERRVEC+2
0257 034506 005737 003274          TST     MEMPAR ;CHECK IF MEMORY CHECK ENABLE AVAILBLE
0258 034512 001005          BNE     25$ ;YES, RETURN
0259 034514 012737 000116 000114    MOV     @MEMVEC+2, MEMVEC ;RESTORE TRAP CATCHER
0260 034522 005037 000116          CLR     MEMVEC+2
0261 034526 000207          25$:   RTS     PC ;RETURN
.SBTTL MEMORY CHECK ENABLE TRAP

```

C10

```

6276      ;*      THIS ROUTINE HANDLES ANY PARITY TRAP. THE LOCATION WHERE
6277      ;*      PARITY WAS BAD IS REPORTED.
6278
6279 034530 012737 034544 001202 MEMERR: MOV      #IOS,$ESCAPE      ;LOAD ESCAPE
6280 034536 011637 003244          MOV      (SP),TRAPPC      ;STORE PC
6281 034542 104001          ERROR 1          ;REPORT MEM PARITY ERROR
6282 034544 005037 001202 10S:   CLR      $ESCAPE      ;CLEAR ESCAPE
6283 034550 032777 001000 144362 BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
6284 034556 001001          BNE      15$          ;YES, FORCE STACK AND TRY AGAIN
6285 034560 000002          RTI          ;NO, RETURN
6286
6287 034562 012706 001100 15$:   MOV      #STACK,SP      ;INITIALIZE STACK
6288 034566 000177 144316          JMP      @SLPERA      ;LOOP ON ERROR
6289
6290      .SBTTL CONTROLLED HALT ROUTINE
6291      ;*      THIS ROUTINE IS ENTERED WHEN A ^C IS ENTERED IN THE KEYBOARD.
6292      ;*      IF NO MONITOR IS PRESENT THE PROGRAM HALTS ELSE THE PROGRAM
6293      ;*      IS FORCED TO LAST PASS AND JUMPS TO END OF PASS.
6294
6295 034572 013702 001270 000000 CTRHLT: MOV      $BASE,R2      ;SET ^611 BASE
6296 034576 012762 100000          MOV      @CLR,RKCS1(R2) ;CLEAR CONTROLLER
6297 034604 104401 043633          TYPE      ,OPRODS      ;TYPE HALT MESSAGE
6298 034610 012706 001100          MOV      #STACK,SP      ;CLEAR STACK
6299 034614 005037 001202          CLR      $ESCAPE      ;CLEAR ESCAPE
6300 034620 105037 001103          CLRB     $ERFLG      ;CLEAR ERROR FLAG
6301 034624 005737 000042          TST      42          ;TEST IF STAND ALONE
6302 034630 001404          BEQ      15$          ;YES - SKIP
6303 034632 005037 033734          CLR      $EOPCT      ;ZERO PASS COUNT
6304 034636 000137 033706          JMP      $EOP        ;GO TO END OF PASS
6305
6306 034642 000000 15:   HALT          ;HALT PROGRAM
6307 034644 000137 003340          JMP      START1      ;GO TO RESTART IF CONTINUE
6308
6309      .SBTTL SCOPE HANDLER ROUTINE
6310
6311      ;*****
6312      ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
6313      ;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
6314      ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
6315      ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6316      ;SW14=1      LOOP ON TEST
6317      ;SW11=1      INHIBIT ITERATIONS
6318      ;SW09=1      LOOP ON ERROR
6319      ;SW08=1      LOOP ON TEST IN SWR<7:0>
6320      ;CALL
6321      ;*      SCOPE      ;;SCOPE=IOT
6322
6323      $SCOPE:
6324 034650 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
6325 034652 032777 040000 144260 15:   BIT      #BIT14,$SWR      ;LOOP ON PRESENT TEST?
6326 034660 001131          BNE      $OVER          ;YES IF SW14=1
6327      ;*****START OF CODE FOR THE XOR TESTER*****
6328 034662 000416          $XTSTR: BR      65      ;IF RUNNING ON THE "XOR" TESTER CHANGE
6329          ;THIS INSTRUCTION TO A "NOP" (NOP=240)
6330 034664 013746 000004          MOV      @ERRVEC,-(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
6331 034670 012737 034710 000004          MOV      #5,$ERRVEC    ;SET FOR TIMEOUT

```

```

6332 034676 005737 177060          TST      20177060          ;; TIME OUT ON XCR?
6333 034702 012637 000004          MOV      (SP)+, 201ERRVEC ;; RESTORE THE ERROR VECTOR
6334 034706 000500          BR       $$VLAD          ;; GO TO THE NEXT TEST
6335 034710 022626          SS:     CMP      (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
6336 034712 012637 000004          MOV      (SP)+, 201ERRVEC ;; RESTORE THE ERROR VECTOR
6337 034716 000440          BR       75             ;; LOOP ON THE PRESENT TEST
6338 034720          ;; *****END OF CODE FOR THE XOR *****
6339 034720 032777 000400 144212 65:; *****
6340 034726 001421          BIT      #BIT08, 20SWR   ;; LOOP ON SPEC. TEST?
6341 034730 005046          BEQ     25             ;; BR IF NO
6342 034732 117716 144202          CLR     -(SP)          ;; CLEAR A TEMP. LOCATION
6343 034736 001414          MOVB    20SWR, (SP)     ;; PICKUP THE DESIRED TEST NUMBER
6344 034740 022716 000051          BEQ     85             ;; PASS - IF BAD TEST NUMBER IN SWR
6345 034744 002411          CMP     #51, (SP)      ;; CHECK THE NUMBER IN THE SWR
6346 034746 011637 001102          BLT     85             ;; BRANCH IF TEST NUMBER IS OUT OF RANGE
6347 034752 005316          MOV     (SP), $TSTNM    ;; UPDATE THE TEST NUMBER
6348 034754 006316          DEC     (SP)          ;; BACKUP BY ONE
6349 034756 062716 035162          ASL     (SP)          ;; SCALE THE TEST NUMBER AS AN INDEX
6350 034762 013637 001106          ADD     $$SWOBTBL, (SP) ;; FORM THE ADDRESS OF TEST POINTER
6351 034766 000466          MOV     2(SP)+, $LPADR  ;; SET LOOP ADDRESS TO DESIRED TEST
6352 034770 005726          BR      $OVER          ;; GO LOOP ON THE TEST
6353 034772 105737 001103 25:     TST     (SP)+          ;; CLEAN THE BAD TEST NUMBER OFF OF THE STACK
6354 034776 001421          TSTB    $ERFLG        ;; HAS AN ERROR OCCURRED?
6355 035000 123737 001115 201103 35:     BEQ     35             ;; BR IF NO
6356 035006 101015          CMPB    $ERMAX, $ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
6357 035010 032777 001000 144122 35:     BHI     35             ;; BR IF NO
6358 035016 001404          BIT     #BIT09, 20SWR  ;; LOOP ON ERROR?
6359 035020 013737 001110 001106 45:     BEQ     45             ;; BR IF NO
6360 035026 000446          MOV     $LPERR, $LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
6361 035030 105037 001103 45:     BR      $OVER
6362 035034 005037 001200          CLRB    $ERFLG        ;; ZERO THE ERROR FLAG
6363 035040 000415          CLR     $TIMES        ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
6364 035042 032777 004000 144070 35:     BR      15             ;; ESCAPE TO THE NEXT TEST
6365 035050 001011          BIT     #BIT11, 20SWR  ;; INHIBIT ITERATIONS?
6366 035052 005737 001222          BNE     15             ;; BR IF YES
6367 035056 001406          TST     $PASS         ;; IF FIRST PASS OF PROGRAM
6368 035060 005237 001104          BEQ     15             ;; INHIBIT ITERATIONS
6369 035064 023737 001200 001104          INC     $SICNT        ;; INCREMENT ITERATION COUNT
6370 035072 002024          CMP     $TIMES, $SICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
6371 035074 012737 000001 001104 15:     BGE     $OVER        ;; BR IF MORE ITERATION REQUIRED
6372 035102 013737 035160 001200          MOV     #1, $SICNT    ;; REINITIALIZE THE ITERATION COUNTER
6373 035110 105237 001102          MOV     $SMXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
6374 035114 113737 001102 001220 55VLAD: INCB    $TSTNM        ;; COUNT TEST NUMBERS
6375 035122 011637 001106          MOVB    $TSTNM, $TSTN  ;; SET TEST NUMBER IN APT MAILBOX
6376 035126 011637 001110          MOV     (SP), $LPADR   ;; SAVE SCOPE LOOP ADDRESS
6377 035132 005037 001202          MOV     (SP), $LPERR   ;; SAVE ERROR LOOP ADDRESS
6378 035136 112737 000001 001115          CLR     $ESCAPE       ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
6379 035144 013777 001102 143770 $OVER:  MOVB    #1, $ERMAX    ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6380 035152 013716 001106          MOV     $TSTNM, 20DISPLAY ;; DISPLAY TEST NUMBER
6381 035156 000002          MOV     $LPADR, (SP)  ;; FUDGE RETURN ADDRESS
6382 035160 003720          RTI                    ;; FIXES PS
6383 035162          $SMXCNT: 2000.        ;; MAX. NUMBER OF ITERATIONS
6384 035162 004352          $SWOBTBL:             ;; STARTING ADDRESS OF TEST 1
6385 035164 004604          .WORD   TST1+2        ;; STARTING ADDRESS OF TEST 2
6386 035166 005036          .WORD   TST2+2        ;; STARTING ADDRESS OF TEST 3
6387 035170 005270          .WORD   TST3+2        ;; STARTING ADDRESS OF TEST 4
6388 035172          .WORD   TST4+2

```

6388	035172	005520	.WORD	TST5+2	::: STARTING ADDRESS OF TEST 5
6389	035174	005750	.WORD	TST6+2	::: STARTING ADDRESS OF TEST 6
6390	035176	006200	.WORD	TST7+2	::: STARTING ADDRESS OF TEST 7
6391	035200	006464	.WORD	TST10+2	::: STARTING ADDRESS OF TEST 10
6392	035202	006734	.WORD	TST11+2	::: STARTING ADDRESS OF TEST 11
6393	035204	007204	.WORD	TST12+2	::: STARTING ADDRESS OF TEST 12
6394	035206	007460	.WORD	TST13+2	::: STARTING ADDRESS OF TEST 13
6395	035210	010130	.WORD	TST14+2	::: STARTING ADDRESS OF TEST 14
6396	035212	010554	.WORD	TST15+2	::: STARTING ADDRESS OF TEST 15
6397	035214	011114	.WORD	TST16+2	::: STARTING ADDRESS OF TEST 16
6398	035216	011506	.WORD	TST17+2	::: STARTING ADDRESS OF TEST 17
6399	035220	012102	.WORD	TST20+2	::: STARTING ADDRESS OF TEST 20
6400	035222	012476	.WORD	TST21+2	::: STARTING ADDRESS OF TEST 21
6401	035224	013072	.WORD	TST22+2	::: STARTING ADDRESS OF TEST 22
6402	035226	013466	.WORD	TST23+2	::: STARTING ADDRESS OF TEST 23
6403	035230	014254	.WORD	TST24+2	::: STARTING ADDRESS OF TEST 24
6404	035232	014744	.WORD	TST25+2	::: STARTING ADDRESS OF TEST 25
6405	035234	015340	.WORD	TST26+2	::: STARTING ADDRESS OF TEST 26
6406	035236	016012	.WORD	TST27+2	::: STARTING ADDRESS OF TEST 27
6407	035240	016464	.WORD	TST30+2	::: STARTING ADDRESS OF TEST 30
6408	035242	017136	.WORD	TST31+2	::: STARTING ADDRESS OF TEST 31
6409	035244	017610	.WORD	TST32+2	::: STARTING ADDRESS OF TEST 32
6410	035246	020376	.WORD	TST33+2	::: STARTING ADDRESS OF TEST 33
6411	035250	021164	.WORD	TST34+2	::: STARTING ADDRESS OF TEST 34
6412	035252	021752	.WORD	TST35+2	::: STARTING ADDRESS OF TEST 35
6413	035254	022442	.WORD	TST36+2	::: STARTING ADDRESS OF TEST 36
6414	035256	023132	.WORD	TST37+2	::: STARTING ADDRESS OF TEST 37
6415	035260	023622	.WORD	TST40+2	::: STARTING ADDRESS OF TEST 40
6416	035262	024312	.WORD	TST41+2	::: STARTING ADDRESS OF TEST 41
6417	035264	025026	.WORD	TST42+2	::: STARTING ADDRESS OF TEST 42
6418	035266	025552	.WORD	TST43+2	::: STARTING ADDRESS OF TEST 43
6419	035270	026276	.WORD	TST44+2	::: STARTING ADDRESS OF TEST 44
6420	035272	027534	.WORD	TST45+2	::: STARTING ADDRESS OF TEST 45
6421	035274	031004	.WORD	TST46+2	::: STARTING ADDRESS OF TEST 46
6422	035276	031570	.WORD	TST47+2	::: STARTING ADDRESS OF TEST 47
6423	035300	032442	.WORD	TST50+2	::: STARTING ADDRESS OF TEST 50
6424	035302	032472	.WORD	TST51+2	::: STARTING ADDRESS OF TEST 51

:::*****
:SBTTL LOOP ON INTERNAL ERROR

6428	035304	032777	001000	143626	SCOPI\$: BIT	#SW9,JSWR	:CHECK IF LOOP ON ERROR
6429	035312	001405			BEQ	SS	:NO RETURN
6430	035314	105737	001103		TSTB	SERFLG	:CHECK IF ERROR OCCURED
6431	035320	001402			BEQ	SS	:NO, RETURN
6432	035322	013716	001110		MOV	\$LPERR,(SP)	:GO BACK TO BEGINNING OF LOOP
6433	035326	000002			SS:	RTI	:RETURN

:::*****
:SBTTL TYPE ERROR ROUTINE
:*ENTRY JSR PC,TYPERR
:*RETURN RTS PC
:*
:*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH
:*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" (\$ERRTB)
:*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
:*THE ERROR.

F10

```

*****
TYPEERR: SAVREG
6444
6445 035330 104413
6446 035332 113737 001102 001220      MOVB  $STSTNM,$STSTN ;GET TEST NUMBER
6447 035340 042737 177400 001220      BIC  _#177400,$STSTN ;CLEAR UNUSED BITS
6448 035346 113700 001114      MOVB  $SITEMB,R0 ;ENTER ERROR NUMBER
6449 035352 042700 177400      BIC  _#177400,R0 ;CLEAR UNUSED BITS
6450 035356 005300      DEC  R0 ;FORM INDEX FOR ERROR TABLE
6451 035360 006300      ASL  R0
6452 035362 006300      ASL  R0
6453 035364 006300      ASL  R0
6454 035366 062700 001300 1$: ADD  _$ERRTB,R0 ;FORM ADDRESS OF ERROR ENTRY
6455 035372 012037 035405      MOV  (R0)+,2$ ;GET EM POINTER
6456 035376 001404      BEQ  3$ ;BRANCH IF THERE ISN'T ONE
6457 035400 104401 001211      TYPE .$CRLF ;TYPE CARRIAGE RETURN LINE FEED
6458 035404 104401      TYPE ;TYPE ERROR MESSAGE (EM)
6459 035406 000000 2$: .WORD 0 ;EM POINTER GOES HERE
6460 035410 012037 035424 3$: MOV  (R0)+,4$ ;GET DH POINTER
6461 035414 001404      BEQ  5$ ;BRANCH IF THERE ISN'T ONE
6462 035416 104401 001211      TYPE .$CRLF ;TYPE CR-LF
6463 035422 104401      TYPE ;TYPE DATA HEADER
6464 035424 000000 4$: .WORD 0 ;DH POINTER GOES HERE
6465 035426 012001 5$: MOV  (R0)+,R1 ;GET DT POINTER
6466 035430 001445      BEQ  20$ ;BRANCH IF THERE ARE NONE
6467 035432 005004      CLR  R4 ;RESET INDENT SWITCH
6468 035434 012000      MOV  (R0)+,R0 ;GET DF POINTER
6469 035436 012002      MOV  (R0)+,R2 ;STORE NUMBER OF DH'S
6470 035440 104401 001211      TYPE .$CRLF
6471 035444 112003 10$: MOVB  (R0)+,R3 ;GET & STORE NUMBER OF DATA WORDS
6472 035446 105720      TSTB (R0)+ ;BUMP PAST FORMAT WORD
6473 035450 005703      IST  R3 ;TEST IF ANY DATA FOR THIS HEADER
6474 035452 001416      BEQ  14$ ;NO - SKIP DATA PRINT
6475 035454 005704      TST  R4 ;CHECK IF INDENT WORDS
6476 035456 001004      BNE  12$ ;YES, GO INDENT
6477 035460 013146 11$: MOV  2(R1)+,-(SP) ;PUT FIRST DATA WORD ON STACK
6478 035462 104402      TYPOC ;TYPE IT
6479 035464 005303      DEC  R3 ;MORE DATA WORDS
6480 035466 001403      BEQ  13$ ;NO-BRANCH
6481 035470 104401 043753 12$: TYPE .SPACE2 ;TYPE SEPARATORS
6482 035474 000771      BR   11$ ;LOOP
6483 035476 104401 001211 13$: TYPE .$CRLF ;TYPE <CR><LF>
6484 035502 005710      TST  (R0) ;CHECK IF NEXT HEADER AVAILABLE
6485 035504 001401      BEQ  14$ ;NO, DO NOT CHANGE INDENT
6486 035506 005104      COM  R4 ;CHANGE INDENT
6487 035510 005302 14$: DEC  R2 ;MORE DH'S?
6488 035512 003414      BLE  20$ ;NO-BRANCH
6489 035514 012037 035534 15$: MOV  (R0)+,18$ ;GET NEXT DH POINTER
6490 035520 001751      BEQ  10$ ;IF NO HEADER GET DATA
6491 035522 005704      TST  R4 ;INDENT?
6492 035524 001402      BEQ  17$ ;NO-BRANCH
6493 035526 104401 043753      TYPE .SPACE2 ;INDENT
6494 035532 104401 17$: TYPE ;TYPE DH
6495 035534 000000 18$: .WORD 0 ;DH POINTER GOES HERE
6496 035536 104401 001211      TYPE .$CRLF
6497 035542 000740      BR   10$ ;LOOP
6498 035544 104414 20$: RESREG
6499 035546 005237 003250      INC  ERRCNT ;INCREMENT ERROR COUNT
    
```

G10

```

6500 035552 032777 010000 143360 BIT #SW12,DSWR ;CHECK IF ABORT AFTER 20 ERRORS
6501 035560 001421 BEQ 25$ ;NO RETURN
6502 035562 022737 000024 003250 CMP #20,ERRCNT ;CHECK IF EROR THRESHOLD EXCEEDED
6503 035570 103015 BHIS 25$ ;NO RETURN
6504 035572 104401 043756 TYPE .ABORT ;TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
6505 ; ERROR THRESHOLD EXCEEDED"
6506 035576 005737 000042 TST 42 ;CHECK IF IN CHAIN MODE
6507 035602 001407 BEQ 30$ ;NO HALT
6508 035604 012737 000001 033734 MOV #1,SEOPCT ;FORCE END OF PASS COUNT TO ONE FOR ABORT
6509 035612 012706 001100 MOV #STACK,SP ;INITIALIZE STACK
6510 035616 005137 033736 JMP SEOP ;BRING IN NEXT PROGRAM IN CHAIN
6511 035622 000000 30$: HALT
6512 035624 000207 25$: RTS PC
6513 ;
6514 ;
6515 ;
6516 ;
6517 ;
6518 ;
6519 ;
6520 ;
6521 ;
6522 ;
6523 ;
6524 ;
6525 ;
6526 ;
6527 ;
6528 ;
6529 ;
6530 ;
6531 ;
6532 ;
6533 ;
6534 ;
6535 ;
6536 ;
6537 ;
6538 ;
6539 ;
6540 ;
6541 ;
6542 ;
6543 ;
6544 ;
6545 ;
6546 ;
6547 ;
6548 ;
6549 ;
6550 ;
6551 ;
6552 ;
6553 ;
6554 ;
6555 ;

.SBTTL GENERATE HEADERS FOR SECTOR, TRACK, AND CYLINDER INCREMENTS
INCHDR: MOV CYLN,HEADER ;LOAD HEADER WORD 1
        CLR HEADER+2 ;CALCULATE HEADER WORD 2
        MOV# TRACK,HEADER+3
        ASR HEADER+2
        ASR HEADER+2
        ASR HEADER+2
        BISB SECTOR,HEADER+2
        BIS #140000,HEADER+2
        MOV HEADER,-(SP) ;GENERATE XOF
        MOV HEADER+2,HEADER+4
        BIC HEADER,HEADER+4
        BIC HEADER+2,(SP)
        BIS (SP)+,HEADER+4
        MOV CYLN,E.DCYL ;INCREMENT DISK ADDRESS
        MOV SECTOR,E.DA
        INCB E.DA
        CMPB #26,E.DA
        BNE 10$
        CLRB E.DA
        INCB E.DA+1
        CMPB #3,E.DA+1
        BNE 10$
        CLR E.DA
        INC E.DCYL
10$: RTS PC ;RETURN

.SBTTL CALCULATE EXPECTED HEADER
CALHDR: CLR R3 ;CLEAR EXPECTED FIRST WORD
        MOV E.DCYL,R1 ;STORE CYLINDER
        BEQ 5$ ;CHECK IF ZERO
        MOV #16,,R0 ;LOAD SHIFT COUNT
1$: ROL R1 ;CALCULATE FIRST WORD
        ROR R3
        DEC R0
        BNE 1$
5$: MOV R3,E.MR2 ;LOAD FIRST WORD
        MOV E.DA,-(SP) ;GET TRACK AND SECTOR
        BEQ 7$ ;CHECK IF TRACK = 0 AND SECTOR = 0
        BIC #377,(SP) ;CLEAR SECTOR BITS
    
```

H10

```
6556 036052 001403          BEQ      6$          :CHECK TRACK = 0
6557 036054 006216          ASR      (SP)       :SHIFT HEAD 3 BITS RIGHT
6558 036056 006216          ASR      (SP)
6559 036060 006216          ASR      (SP)
6560 036062 153716 003206    6$:  B1SB      E DA, (SP) ;OR IN SECTOR BITS
6561 036066 012601          7$:  MOV      (SP)+,R1
6562 036070 032737 010000 003200    BIT      #CFMT,E.CS1 ;CHECK FORMAT = 24 SECTOR
6563 036076 001402          BEQ      8$          ;NO, CALCULATE SECOND WORD
6564 036100 052701 001000    8$:  BIS      #BIT9,R1 ;SET FORMAT BIT
6565 036104 005003          CLR      R3         ;CLEAR EXPECTED SECOND WORD
6566 036106 005701          TST      R1         ;CHECK IF WORD = 0
6567 036110 001406          BEQ      15$        ;YES, RETURN
6568 036112 012700 000020    MOV      #26.,RO   ;LOAD SHIFT COUNT
6569 036116 006101          10$: ROL      R1
6570 036120 006003          ROR      R3
6571 036122 005300          DEC      RO
6572 036124 001374          BNE      10$
6573 036126 010337 003230    15$: MOV      R3,E.MR3 ;LOAD SECOND WORD
6574 036132 000207          RTS      PC         ;RETURN
6575
6576          ;:*****
6577          ;SBTTL GENERATE ECC WORD
6578
6579 036134 005737 003252    ECCGEN: TST      P1.BIT ;CHECK IF 1
6580 036140 001410          BEQ      3$          ;NO, CHECK IF BIT 31 = 1
6581 036142 032737 000001 003266    BIT      #1,ECCHI ;NO, CHECK IF BIT 31 = 1
6582 036150 001410          BEQ      5$          ;NO, SET ECC XORED BIT
6583 036152 005037 003272    1$:  CLR      ECCXOR ;CLEAR ECC XORED BIT
6584 036156 000241          CLC      ;CLEAR CARRY FLOP
6585 036160 000410          BR      7$          ;SHIFT IN ECC BIT
6586
6587 036162 032737 000001 003266    3$:  BIT      #1,ECCHI ;CHECK IF BIT 31 = 1
6588 036170 001770          BEQ      1$          ;NO, CLEAR ECC XORED BIT
6589 036172 012737 000001 003272    5$:  MOV      #1,ECCXOR ;SET ECC XOR
6590 036200 000261          SEC      ;SET CARRY FLOP
6591 036202 006037 003270    7$:  ROR      ECCLO
6592 036206 006037 003266    ROR      ECCHI
6593 036212 005737 003272    TST      ECCXOR ;CHECK IF XOR NEEDED
6594 036216 001422          BEQ      10$        ;NO, RETURN
6595 036220 012746 020020    MOV      #BIT13!BIT4,-(SP) ;DO XOR OF ECC BITS
6596 036224 043716 003270          BIC      ECCLO,(SP) ; 0, 2, 11, 21, 23
6597 036230 042737 020020 003270    BIC      #BIT13!BIT4,ECCLO
6598 036236 052637 003270          BIS      (SP)+,ECCLO
6599 036242 012746 002400          MOV      #BIT10!BIT8,-(SP)
6600 036246 043716 003266          BIC      ECCHI,(SP)
6601 036252 042737 002400 003266    BIC      #BIT10!BIT8,ECCHI
6602 036260 052637 003266          BIS      (SP)+,ECCHI
6603 036264 013737 003266 003234    10$: MOV      ECCHI,E.ECPT ;STORE ECC PATTERN
6604 036272 042737 174000 003234    BIC      #174000,E.ECPT ;MASK UNSEEN BITS
6605 036300 000207          RTS      PC         ;RETURN
6606          ;:*****
6607          ;SBTTL SIMULATE ONE BIT WRITE IN MAINTENCE MODE
6608
6609 036302 052737 042040 003224    WRTBIT: BIS      #DMD!MEWD!WRTGAT,E.MR1 ;INITIALIZE
6610 036310 042737 114000 003224    BIC      #RDGATE!PCA!PCD,E.MR1 ; EXPECTED MR1
6611 036316 005737 003254          TST      PR.BIT    ;CHECK IF ONE
```

6612	036322	001122				BNE	20\$;YES, SIMULATE A ONE
6613	036324	005737	003256			TST	M1.BIT		;CHECK IF PREVIOUS ONE
6614	036330	001023				BNE	10\$;YES, NO TRANSITION
6615	036332	042737	002000	003224		BIC	#MEWD,E.		;INDICATE TRANSITION
6616	036340	005737	003252			TST	P1.BIT		;CHECK IF NEXT BIT = 1
6617	036344	001007				BNE	5\$;YES, CHECK FOR PRECOMP ADVANCE
6618	036346	005737	003260			TST	M2.BIT		;CHECK FOR PRECOMP. DELAY
6619	036352	001412				BEQ	10\$;NO, CLOCK IN ZERO
6620	036354	052737	010000	003224		BIS	#PCD,E.MR1		;SET PRECOMP. DELAY
6621	036362	000406				BR	10\$;CLOCK IN ZERO
6622									
6623	036364	005737	003260		5\$:	TST	M2.BIT		;CHECK FOR PRECOMP. ADVANCE
6624	036370	001003				BNE	10\$;CLOCK IN
6625	036372	052737	004000	003224		BIS	#PCA,E.MR1		;SET PRECOMP. DELAY
6626	036400	012762	000440	000026	10\$:	MOV	#DMD!MCLK,RKMR1(R2)		;CLOCK IN DATA BIT
6627	036406	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
6628	036414	016237	000026	003164		MOV	RKMR1(R2),T.MR1		;STORE MR1
6629	036422	023737	003164	003224		CMP	T.MR1,E.MR1		;CHECK IF MR1 CORRECT
6630	036430	001416				BEQ	15\$;YES, CLOCK IN REST OF BIT
6631	036432	012737	036452	001202		MOV	#13\$, \$ESCAPE		;LOAD ESCAPE
6632	036440	012737	052775	001312		MOV	#EMW2,EMW+2		;LOAD ERROR MESSAGE
6633	036446	011646				MOV	(SP),-(SP)		;STORE RETURN
6634	036450	000207				RTS	PC		;REPORT ERROR
6635	036452	032777	001000	142460	13\$:	BIT	#SW9, \$SWR		;CHECK IF LOOP ON ERROR
6636	036460	001402				BEQ	15\$;NO, CONTINUE
6637	036462	000137	037014			JMP	63\$;GO LOOP ON ERROR
6638									
6639	036466	052737	002000	003224	15\$:	BIS	#MEWD,E.MR1		;RESET TRANSITION INDICATION
6640	036474	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)		;CLOCK IN DATA BIT
6641	036502	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
6642	036510	016237	000026	003164		MOV	RKMR1(R2),T.MR1		;STORE MR1
6643	036516	023737	003164	003224		CMP	T.MR1,E.MR1		;CHECK IF MR1 CORRECT
6644	036524	001414				BEQ	18\$;YES, RETURN
6645	036526	012737	036546	001202		MOV	#17\$, \$ESCAPE		;LOAD ESCAPE
6646	036534	012737	053065	001312		MOV	#EMW4,EMW+2		;LOAD ERROR MESSAGE
6647	036542	011646				MOV	(SP),-(SP)		;SAVE RETURN
6648	036544	000207				RTS	PC		;REPORT ERROR
6649									
6650	036546	032777	001000	142364	17\$:	BIT	#SW9, \$SWR		;CHECK IF LOOP ON ERROR
6651	036554	001117				BNE	63\$		
6652	036556	005037	001202		18\$:	CLR	\$ESCAPE		;CLEAR ESCAPE
6653	036562	062716	000002			ADD	#2,(SP)		;ADJUST ESCAPE
6654	036566	000207				RTS	PC		;RETURN
6655									
6656									
6657	036570	005737	003252		20\$:	TST	P1.BIT		;CHECK IF NEXT BIT A ONE
6658	036574	001007				BNE	30\$;YES, CHECK FOR PRECOMP. DELAY
6659	036576	005737	003256			TST	M1.BIT		;CHECK FOR PRECOMP. ADVANCE
6660	036602	001412				BEQ	40\$;NO, CHECK MR1
6661	036604	052737	004000	003224		BIS	#PCA,E.MR1		;SET PRECOMP ADVANCE
6662	036612	000406				BR	40\$;CHECK MR1
6663									
6664	036614	005737	003256		30\$:	TST	M1.BIT		;CHECK FOR PRECOMP. DELAY
6665	036620	001003				BNE	40\$;NO, CHECK MR1
6666	036622	052737	010000	003224		BIS	#PCD,E.MR1		;SET PRECOMP. DELAY
6667	036630	012762	000440	000026	40\$:	MOV	#DMD!MCLK,RKMR1(R2)		;CLOCK IN DATA BIT

J10

```

6668 036636 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6669 036644 016237 000026 003164      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6670 036652 023737 003164 003224      CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6671 036660 001414          BEQ      45$ ;YES, CLOCK IN REST OF BIT
6672 036662 012737 036702 001202      MOV      #42$, $ESCAPE ;LOAD ESCAPE
6673 036670 012737 052775 001312      MOV      #EMW2,EMW+2 ;LOAD ERROR MESSAGE
6674 036676 011646          MOV      (SP),-(SP) ;STORE RETURN
6675 036700 000207          RTS      PC ;REPORT ERROR
6676
6677 036702 032777 001000 142230 42$: BIT      #SW9,$SWR ;CHECK IF LOOP ON ERROR
6678 036710 001041          BNE      63$ ;YES, LOOP ERROR
6679 036712 042737 002000 003224 45$: BIC      #MEWD,E.MR1 ;INDICATE A TRANSITION
6680 036720 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK TRANSITION
6681 036726 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6682 036734 016237 000026 003164      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6683 036742 023737 003164 003224      CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6684 036750 001414          BEQ      50$ ;YES, RETURN
6685 036752 012737 036772 001202      MOV      #47$, $ESCAPE ;LOAD ESCAPE
6686 036760 012737 053065 001312      MOV      #EMW4,EMW+2 ;LOAD ERROR MESSAGE
6687 036766 011646          MOV      (SP),-(SP) ;SAVE RETURN
6688 036770 000207          RTS      PC ;REPORT ERROR
6689
6690 036772 032777 001000 142140 47$: BIT      #SW9,$SWR ;CHECK IF LOOP ON ERROR
6691 037000 001005          BNE      63$ ;YES, REPORT ERROR
6692 037002 005037 001202 50$: CLR      $ESCAPE ;LOAD ESCAPE
6693 037006 062716 000002      ADD      #2,(SP) ;ADJUST RETURN
6694 037012 000207          RTS      PC ;RETURN
6695
6696 037014 005037 001202 63$: CLR      $ESCAPE ;CLEAR ESCAPE
6697 037020 012706 001100      MOV      #STACK,SP ;FORCE STACK
6698 037024 000177 142060      JMP      @SLPERR ;JUMP TO LOOP ADDRESS
6699
6700 ;*****
6701 .SBTTL SIMULATE ONE BIT OF READ DATA IN MAINTENENCE MODE
6702
6703 03703C 105737 003254 ROBIT: TSTB   PR.BIT ;CHECK IF ONE
6704 037034 001024          BNE      10$ ;YES, SIMULATE ONE
6705 037036 105737 003256      TSTB   M1.BIT ;CHECK IF PREVIOUS ONE
6706 037042 001404          BEQ      4$ ;INSERT TRANSITION
6707 037044 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2) ;DO NOT INSERT TRANSITION
6708 037052 000403          BR      5$ ;CLOCK IN ZERO
6709
6710 037054 012762 001440 000026 4$: MOV      #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
6711 037062 012762 000040 000026 5$: MOV      #DMD,RKMR1(R2) ;CLOCK IN ZERO
6712 037070 012762 000440 000025      MOV      #DMD!MCLK,RKMR1(R2)
6713 037076 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6714 037104 000207          RTS      PC ;RETURN
6715
6716 037106 012762 000440 000026 10$: MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
6717 037114 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6718 037122 012762 001440 000026      MOV      #DMD!MCLK!MERD,RKMR1(R2)
6719 037130 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6720 037136 000207          RTS      PC ;RETURN
6721 .SBTTL APT COMMUNICATIONS ROUTINE
6722
6723 ;*****

```

```

6724 037140 112737 000001 037404 SATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
6725 037146 112737 000001 037402 SATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
6726 037154 000403 BR SATYC
6727 037156 112737 000001 037404 SATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
6728 037164 SATYC:
6729 037164 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
6730 037166 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
6731 037170 105737 037402 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
6732 037174 001450 BEQ 5$ ;;IF NOT: BR
6733 037176 122737 000001 001234 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
6734 037204 001031 BNE 3$ ;;IF NOT: BR
6735 037206 132737 000100 001235 BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
6736 037214 001425 BEQ 3$ ;;IF NOT: BR
6737 037216 017600 MOV #4(SP),R0 ;;GET MESSAGE ADDR.
6738 037222 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
6739 037230 005737 001214 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
6740 037234 001375 BNE 1$ ;;IF NOT: WAIT
6741 037236 010037 001230 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
6742 037242 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
6743 037244 001376 BNE 2$
6744 037246 163700 001230 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
6745 037252 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
6746 037254 010037 001232 MOV R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
6747 037260 012737 000004 001214 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
6748 037266 000413 BR 5$
6749 037270 017637 000004 037314 3$: MOV #4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
6750 037276 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
6751 037304 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
6752 037310 004737 037606 JSR PC,$TYPE ;;CALL TYPE MACRO
6753 037314 000000 4$: .WORD 0
6754 037316 5$:
6755 037316 105737 037404 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
6756 037322 001416 BEQ 12$ ;;IF NOT: BR
6757 037324 005737 001234 TST $ENV ;;RUNNING UNDER APT?
6758 037330 001413 BEQ 12$ ;;IF NOT: BR
6759 037332 005737 001214 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
6760 037336 001375 BNE 11$ ;;IF NOT: WAIT
6761 037340 017637 000004 001216 MOV #4(SP),$FATAL ;;GET ERROR #
6762 037346 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
6763 037354 005237 001214 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
6764 037360 105037 037404 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
6765 037364 105037 037403 CLRB $LFLG ;;CLEAR LOG FLAG
6766 037370 105037 037402 CLRB $MFLG ;;CLEAR MESSAGE FLAG
6767 037374 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
6768 037376 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
6769 037400 000207 RTS PC ;;RETURN
6770 037402 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
6771 037403 000 $LFLG: .BYTE 0 ;;LOG FLAG
6772 037404 000 $FFLG: .BYTE 0 ;;FATAL FLAG
6773 037406 .EVEN
6774 000200 APTSIZE=200
6775 000001 APTENV=001
6776 000100 APTPOOL=100
6777 000040 APTCSUP=040
6778 .SBTTL ERROR HANDLER ROUTINE
6779

```

```

6780
6781
6782
6783
6784
6785
6786
6787
6788
6789
6790
6791
6792 037406
6793 037406 104407
6794 037410 105237 001103
6795 037414 001775
6796 037416 013777 001102 141516
6797 037424 032777 002000 141506
6798 037432 001402
6799 037434 104401 001204
6800 037440 005237 001112
6801 037444 011637 001116
6802 037450 162737 000002 001116
6803 037456 117737 141434 001114
6804 037464 032777 020000 141446
6805 037472 001004
6806 037474 004737 035330
6807 037500 104401 001211
6808 037504
6809 037504 122737 000001 001234
6810 037512 001007
6811 037514 113737 001114 037526
6812 037522 004737 037156
6813 037526 000
6814 037527 000
6815 037530 000777
6816 037532 005777 141402
6817 037536 100002
6818 037540 000000
6819 037542 104407
6820 037544 032777 001000 141366
6821 037552 001402
6822 037554 013716 001110
6823 037560 005737 001202
6824 037564 001402
6825 037566 013716 001202
6826 037572
6827 037572 022737 034076 000042
6828 037600 001001
6829 037602 000000
6830 037604
6831 037504 000002
6832
6833
6834
6835

```

```

*****
*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

SERFOR:
7$: CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
INCB SERFLG ;; SET THE ERROR FLAG
BEG 7$ ;; DON'T LET THE FLAG GO TO ZERO
MOV $STNM,DISP ;; DISPLAY TEST NUMBER AND ERROR FLAG
BIT $BIT10,SWR ;; BELL ON ERROR?
BEQ 1$ ;; NO - SKIP
TYPE $BELL ;; RING BELL
1$: INC $ERTTL ;; COUNT THE NUMBER OF ERRORS
MOV (SP),SERPC ;; GET ADDRESS OF ERROR INSTRUCTION
SUB $2,SERPC
MOV $SERPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
BIT $BIT13,SWR ;; SKIP TYPEOUT IF SET
BNE 20$ ;; SKIP TYPEOUTS
JSR PC,TYPERR ;; GO TO USER ERROR ROUTINE
TYPE $SCRLF

20$: CMPB $APTENV,$ENV ;; RUNNING IN APT MODE
BNE 2$ ;; NO SKIP APT ERROR REPORT
MOVB $ITEMB,21$ ;; SET ITEM NUMBER AS ERROR NUMBER
JSR PC,$ATY4 ;; REPORT FATAL ERROR TO APT

21$: .BYTE 0
.BYTE 0

22$: BR 22$ ;; APT ERROR LOOP
2$: TST $SWR ;; HALT ON ERROR
BPL 3$ ;; SKIP IF CONTINUE
HALT ;; HALT ON ERROR!
CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
BIT $BIT09,SWR ;; LOOP ON ERROR SWITCH SET?
BEQ 4$ ;; BR IF NO
MOV $LPERR,(SP) ;; FUDGE RETURN FOR LOOPING
4$: TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
BEQ 5$ ;; BR IF NONE
MOV $ESCAPE,(SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE

5$: CMP $SENDAD,$42 ;; ACT-11 AUTO-ACCEPT?
BNE 6$ ;; BRANCH IF NO
HALT ;; YES

6$: RTI ;; RETURN

```

.SBTTL TYPE ROUTINE

M10

TYPE ROUTINE

```

6836      ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
6837      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
6838      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
6839      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
6840      ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
6841      ;*
6842      ;*CALL:
6843      ;*1) USING A TRAP INSTRUCTION
6844      ;*      TYPE      .MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
6845      ;*OR
6846      ;*      TYPE
6847      ;*      MESADR
6848      ;*
6849
6850 037606 105737 001157      $TYPE:  TSTB      $TPFLG      ;; IS THERE A TERMINAL?
6851 037607 100002          BPL      1$      ;; BR IF YES
6852 037614 000000          HALT          ;; HALT HERE IF NO TERMINAL
6853 037616 000430          BR      3$      ;; LEAVE
6854 037620 010046      1$:  MOV      RO, -(SP)      ;; SAVE RO
6855 037622 017600 000002      MOV      @2(SP), RO      ;; GET ADDRESS OF ASCIZ STRING
6856 037626 122737 000001 001234      CMPB     $APTENV, $ENV      ;; RUNNING IN APT MODE
6857 037634 001011          BNE      62$      ;; NO, GO CHECK FOR APT CONSOLE
6858 037636 132737 000100 001235      BITB     $APTPOOL, $ENVM      ;; SPOOL MESSAGE TO APT
6859 037644 001405          BEQ      62$      ;; NO, GO CHECK FOR CONSOLE
6860 037646 010037 037656      MOV      RO, 61$      ;; SETUP MESSAGE ADDRESS FOR APT
6861 037652 004737 037146      JSR      PC, $ATY3      ;; SPOOL MESSAGE TO APT
6862 037656 000000          .WORD     0      ;; MESSAGE ADDRESS
6863 037660 132737 000040 001235      62$:  BITB     $APTCSUP, $ENVM      ;; APT CONSOLE SUPPRESSED
6864 037666 001003          BNE      60$      ;; YES, SKIP TYPE OUT
6865 037670 112046      2$:  MOVB     (RO)+, -(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
6866 037672 001005          BNE      4$      ;; BR IF IT ISN'T THE TERMINATOR
6867 037674 005726          TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
6868 037676 012600      60$:  MOV      (SP)+, RO      ;; RESTORE RO
6869 037700 062716 000002      3$:  ADD      #2, (SP)      ;; ADJUST RETURN PC
6870 037704 000002          RTI          ;; RETURN
6871 037706 122716 000011      4$:  CMPB     $HT, (SP)      ;; BRANCH IF <HT>
6872 037712 001430          BEQ      8$      ;;
6873 037714 122716 000200      CMPB     $CRLF, (SP)      ;; BRANCH IF NOT <CRLF>
6874 037720 001006          BNE      5$      ;;
6875 037722 005726          TST      (SP)+      ;; POP <CR><LF> EQUIV
6876 037724 104401          TYPE          ;; TYPE A CR AND LF
6877 037726 001211          $CRLF
6878 037730 105037 040064      CLRAB    $CHARCNT      ;; CLEAR CHARACTER COUNT
6879 037734 000755          BR      2$      ;; GET NEXT CHARACTER
6880 037736 004737 040020      5$:  JSR      PC, $TYPEC      ;; GO TYPE THIS CHARACTER
6881 037742 123726 001156      6$:  CMPB     $FILLC, (SP)+      ;; IS IT TIME FOR FILLER CHARS.?
6882 037746 001350          BNE      2$      ;; IF NO GO GET NEXT CHAR.
6883 037750 013746 001154      MOV      $NULL, -(SP)      ;; GET # OF FILLER CHARS. NEEDED
6884      AND     THE NULL CHAR.
6885 037754 105366 000001      7$:  DECB     1(SP)      ;; DOES A NULL NEED TO BE TYPED?
6886 037760 002770          BLT      6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
6887 037762 004737 040020      JSR      PC, $TYPEC      ;; GO TYPE A NULL
6888 037766 105337 040064      DECB     $CHARCNT      ;; DO NOT COUNT AS A COUNT
6889 037772 000770          BR      7$      ;; LOOP
6890
6891      ;HORIZONTAL TAB PROCESSOR

```

```

6892
6893 037774 112716 000040 8$: MOVB #' (SP) ;; REPLACE TAB WITH SPACE
6894 040000 004737 040023 9$: JSR PC,$TYPEC ;; TYPE A SPACE
6895 040004 132737 000007 040064 BITB #7,$CHARCNT ;; BRANCH IF NOT AT
6896 040012 001372 BNE 9$ ;; TAB STOP
6897 040014 005726 TST (SP)+ ;; POP SPACE OFF STACK
6898 040016 000724 BR 2$ ;; GET NEXT CHARACTER
6899 040020 105777 141124 $TYPEC: STB 2$STPS ;; WAIT UNTIL PRINTER IS READY
6900 040024 100375 BPL $TYPEC
6901 040026 116677 000002 141116 MOVB 2(SP),2$STPB ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6902 040034 122766 000015 000002 CMPB #CR,2(SP) ;; IS CHARACTER A CARRIAGE RETURN?
6903 040042 001003 BNE 1$ ;; BRANCH IF NO
6904 040044 105037 040064 CLRB $CHARCNT ;; YES--CLEAR CHARACTER COUNT
6905 040050 000406 BR $TYPEX ;; EXIT
6906 040052 122766 000012 000002 1$: CMPB #LF,2(SP) ;; IS CHARACTER A LINE FEED?
6907 040060 001402 BEQ $TYPEX ;; BRANCH IF YES
6908 040062 105227 INCB (PC)+ ;; COUNT THE CHARACTER
6909 040064 00000C $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
6910 040066 000207 $TYPEX: RTS PC
6911
6912 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
6913
6914 *****
6915 ; THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6916 ; OCTAL (ASCII) NUMBER AND TYPE IT.
6917 ; $TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6918 ; *CALL:
6919 ; * MOV NUM,-(SP) ;; NUMBER TO BE TYPED
6920 ; * TYPOS ;; CALL FOR TYPEOUT
6921 ; * .BYTE N ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6922 ; * .BYTE M ;; M=1 OR 0
6923 ; * ;; 1=TYPE LEADING ZEROS
6924 ; * ;; 0=SUPPRESS LEADING ZEROS
6925
6926 ; $STYPOS---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
6927 ; $TYPOS OR $TYPOC
6928 ; *CALL:
6929 ; * MOV NUM,-(SP) ;; NUMBER TO BE TYPED
6930 ; * TYPON ;; CALL FOR TYPEOUT
6931
6932 ; $TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6933 ; *CALL:
6934 ; * MOV NUM,-(SP) ;; NUMBER TO BE TYPED
6935 ; * TYPOC ;; CALL FOR TYPEOUT
6936
6937 040070 017646 000000 $TYPOS: MOV 2(SP),-(SP) ;; PICKUP THE MODE
6938 040074 116637 000001 040313 MOVB 1(SP),$OFILL ;; LOAD ZERO FILL SWITCH
6939 040102 112637 040315 MOVB (SP)+,$SOMODE+1 ;; NUMBER OF DIGITS TO TYPE
6940 040106 062716 000002 ADD #2,(SP) ;; ADJUST RETURN ADDRESS
6941 040112 000406 BR $TYPON
6942 040114 112737 000001 040313 $TYPOC: MOVB #1,$OFILL ;; SET THE ZERO FILL SWITCH
6943 040122 112737 000006 040315 MOVB #6,$SOMODE+1 ;; SET FOR SIX(6) DIGITS
6944 040130 112737 000005 040312 $TYPON: MOVB #5,$OCNT ;; SET THE ITERATION COUNT
6945 040136 010346 MOV R3,-(SP) ;; SAVE R3
6946 040140 010446 MOV R4,-(SP) ;; SAVE R4
6947 040142 010546 MOV R5,-(SP) ;; SAVE R5

```

B11

6998	040144	113704	040315	MOVB	\$OMODE+1,R4	::GET THE NUMBER OF DIGITS TO TYPE
6999	040150	005404		NEG	R4	
6990	040150	062704	000006	ADD	#6,R4	::SUBTRACT IT FOR MAX. ALLOWED
6991	040156	110437	040314	MOVB	R4,\$OMODE	::SAVE IT FOR USE
6992	040160	113704	040313	MOVB	\$OFILL,R4	::GET THE ZERO FILL SWITCH
6993	040166	016605	000012	MOV	12,\$P1,R5	::PICKUP THE INPUT NUMBER
6994	040172	005003		CLR	R3	::CLEAR THE OUTPUT WORD
6995	040174	006105		15: ROL	R5	::ROTATE MSB INTO "C"
6996	040176	000404		BR	25	::GO DO MSB
6997	040200	006105		25: ROL	R5	::FORM THIS DIGIT
6998	040202	006105		ROL	R5	
6999	040204	036105		ROL	R5	
6990	040206	010503		MOV	R5,R3	
6991	040210	006103		35: ROL	R3	::GET LSB OF THIS DIGIT
6992	040212	105337	040314	DECB	\$OMODE	::TYPE THIS DIGIT?
6993	040216	100016		BPL	75	::BR IF NO
6994	040220	042703	177770	BTC	#177770,R3	::GET RID OF JUNK
6995	040224	001002		BNE	45	::TEST FOR 0
6996	040226	005704		TST	R4	::SUPPRESS THIS 0?
6997	040230	001403		BEQ	55	::BR IF YES
6998	040232	005204		45: INC	R4	::DON'T SUPPRESS ANYMORE 0'S
6999	040234	052703	000060	BIS	#'0,R3	::MAKE THIS DIGIT ASCII
6990	040240	052703	000040	55: BIS	#' ,R3	::MAKE ASCII IF NOT ALREADY
6991	040244	110337	040310	MOVB	R3,#5	::SAVE FOR TYPING
6992	040250	104401	040310	TYPE	#5	::GO TYPE THIS DIGIT
6993	040254	105337	040312	75: DECB	\$OCNT	::COUNT BY 1
6994	040260	003347		BGT	25	::BR IF MORE TO DO
6995	040262	002402		BLT	65	::BR IF DONE
6996	040264	005204		INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
6997	040266	000744		BR	25	::GO DO THE LAST DIGIT
6998	040270	012605		65: MOV	(SP)+,R5	::RESTORE R5
6999	040272	012604		MOV	(SP)+,R4	::RESTORE R4
6990	040274	012603		MOV	(SP)+,R3	::RESTORE R3
6991	040276	016666	000002 000004	MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
6992	040304	012616		MOV	(SP)+,(SP)	
6993	040306	000002		RTI		::RETURN
6994	040310	000		85: .BYTE	0	::STORAGE FOR ASCII DIGIT
6995	040311	000		.BYTE	00	::TERMINATOR FOR TYPE ROUTINE
6996	040312	000		\$OCNT:	.BYTE 00	::OCTAL DIGIT COUNTER
6997	040313	000		\$OFILL:	.BYTE 00	::ZERO FILL SWITCH
6998	040314	000000		\$OMODE:	.WORD 0	::NUMBER OF DIGITS TO TYPE
6999				.SETTL	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE	
7000				*****		
7001				*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT		
7002				*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE		
7003				*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED		
7004				*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE		
7005				*REPLACED WITH SPACES.		
7006				*CALL:		
7007				* MOV	NUM,-(SP)	::PUT THE BINARY NUMBER ON THE STACK
7008				* TYPDS		::GO TO THE ROUTINE
7009				*****		
7010	040316	010046		\$TYPDS:	MOV R0,-(SP)	::PUSH R0 ON STACK
7011	040316	010146		MOV	R1,-(SP)	::PUSH R1 ON STACK

C11

```

7004 040322 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7005 040324 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7006 040326 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7007 040330 012746 020200    MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
7008 040334 016605 000020    MOV      20(SP),R5     ;;GET THE INPUT NUMBER
7009 040340 100004      BPL      1$           ;;BR IF INPUT IS POS.
7010 040342 005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
7011 040344 112766 000055 000001    MOVVB   #'-,(SP)      ;;MAKE THE ASCII NUMBER NEG.
7012 040352 005000 1$:      CLR      R0           ;;ZERO THE CONSTANTS INDEX
7013 040354 012703 040532    MOV      #SDBLK,R3     ;;SETUP THE OUTPUT POINTER
7014 040360 112723 000040    MOVVB   #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
7015 040364 035002 2$:      CLR      R2           ;;CLEAR THE BCD NUMBER
7016 040366 016001 040522    MOV      $DTBL(R0),R1  ;;GET THE CONSTANT
7017 040372 160105 3$:      SUB      R1,R5         ;;FORM THIS BCD DIGIT
7018 040374 002402      BLT      4$           ;;BR IF DONE
7019 040376 005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
7020 040400 000774      BR       3$
7021 040402 060105 4$:      ADD      R1,R5         ;;ADD BACK THE CONSTANT
7022 040404 005702      TST      R2           ;;CHECK IF BCD DIGIT=0
7023 040406 001002      BNE      5$           ;;FALL THROUGH IF 0
7024 040410 105716      TSTB    (SP)         ;;STILL DOING LEADING 0'S?
7025 040412 100407      BMI     7$           ;;BR IF YES
7026 040414 106316 5$:      ASLB    (SP)         ;;MSD?
7027 040416 103003      BCC     6$           ;;BR IF NO
7028 040420 116663 000001 177777    MOVVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
7029 040426 052702 6$:      BIS     #'0,R2        ;;MAKE THE BCD DIGIT ASCII
7030 040432 052702 7$:      BIS     #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7031 040436 110223      MOVVB   R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7032 040440 005720      TST     (R0)+        ;;JUST INCREMENTING
7033 040442 020027 000010    CMP     R0,#10       ;;CHECK THE TABLE INDEX
7034 040446 002746      BLT     2$           ;;GO DO THE NEXT DIGIT
7035 040450 003002      BGT     8$           ;;GO TO EXIT
7036 040452 010502      MOV     R5,R2        ;;GET THE LSD
7037 040454 000764      BR      6$           ;;GO CHANGE TO ASCII
7038 040456 105726 8$:      TSTB   (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
7039 040460 100003      BPL     9$           ;;BR IF NO
7040 040462 116663 177777 177776    MOVVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
7041 040470 105013 9$:      CLRB   (R3)         ;;SET THE TERMINATOR
7042 040472 012605      MOV     (SP)+,R5     ;;POP STACK INTO R5
7043 040474 012603      MOV     (SP)+,R3     ;;POP STACK INTO R3
7044 040476 012602      MOV     (SP)+,R2     ;;POP STACK INTO R2
7045 040500 012601      MOV     (SP)+,R1     ;;POP STACK INTO R1
7046 040502 012600      MOV     (SP)+,R0     ;;POP STACK INTO R0
7047 040504 104401 040532    TYPE   $SDBLK        ;;NOW TYPE THE NUMBER
7048 040510 016666 000002 000004    MOV     2(SP),4(SP)  ;;ADJUST THE STACK
7049 040516 012616      MOV     (SP)+,(SP)
7050 040520 000002      RTI
7051 040522 023420  $DTBL: 10000.        ;;RETURN TO USER
7052 040524 001750      1000.
7053 040526 000144      100.
7054 040530 000012      10.
7055 040532 000004      $SDBLK: .BLKW 4
7056 .SBTTL TTY INPUT ROUTINE
7057
7058 ;;*****
7059 .ENABL LSB

```

```

7060 040542 000000      $TKCNT: .WORD 0          ;; NUMBER OF ITEMS IN QUEUE
7061 040544 000000      $TKQIN: .WORD 0          ;; INPUT POINTER
7062 040546 000000      $TKQOUT: .WORD 0         ;; OUTPUT POINTER
7063 040550 000001      $TKQSRT: .BLKB 1         ;; TTY KEYBOARD QUEUE
7064 040551
7065 040552
7066
7067
7068
7069
7070
7071
7072
7073
7074
7075 040552 005037 040542      $TKINT: CLR      $TKCNT          ;; CLEAR COUNT OF ITEMS IN QUEUE
7076 040556 012737 040550 040544      MOV      $TKQSRT,$TKQIN        ;; MOVE THE STARTING ADDRESS OF THE
7077 040564 013737 040544 040546      MOV      $TKQIN,$TKQOUT        ;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
7078 040572 012737 040622 000060      MOV      $TKSRV,$TKVEC        ;; INITIALIZE THE KEYBOARD VECTOR
7079 040600 012737 000200 000062      MOV      #200,$TKVEC+2        ;; "BR" LEVEL 4
7080 040606 005777 140334          TST      $TKB                  ;; CLEAR DONE FLAG
7081 040612 012777 000100 140324      MOV      #100,$TKS            ;; ENABLE TTY KEYBOARD INTERRUPT
7082 040620 000207          RTS      PC                    ;; RETURN TO CALLER
7083
7084
7085
7086
7087
7088
7089
7090
7091 040622 117746 140320      $TKSRV: MOVB     $TKB, -(SP)      ;; PICKUP THE CHARACTER
7092 040626 042716 177600      BIC      #10177, (SP)           ;; STRIP THE JUNK
7093 040632 021627 000003      CMP      (SP), #3              ;; IS IT A CONTROL C?
7094 040636 001007          BNE      1$                    ;; BRANCH IF NO
7095 040640 104401 041736      TYPE     $CNTLC                ;; TYPE A CONTROL-C (10)
7096 040644 004737 040552      JSP     PC,$TKINT              ;; INIT THE KEYBOARD
7097 040650 005726          TST     (SP)+                  ;; CLEAN UP STACK
7098 040652 000137 034572      JMP     CTRHLT                 ;; CONTROL C RESTART
7099 040656 021627 000007      1$:    CMP      (SP), #7         ;; IS IT A CONTROL G?
7100 040662 001004          BNE     2$                    ;; BRANCH IF NO
7101 040664 022737 000176 001140      CMP      $SWREG,$SWR           ;; IS SOFT-SWR SELECTED?
7102 040672 001500          BEQ     6$                    ;; GO TO SWR CHANGE
7103
7104
7105 040674 022737 000001 040542      2$:    CMP      #1,$TKCNT        ;; IS THE QUEUE FULL?
7106 040702 001004          BNE     3$                    ;; BRANCH IF NO
7107 040704 104401 001204      TYPE     $BELL                 ;; RING THE TTY BELL
7108 040710 005726          TST     (SP)+                  ;; CLEAN CHARACTER OFF OF STACK
7109 040712 000451          BR      5$                    ;; EXIT
7110 040714 021627 000023      3$:    CMP      (SP), #23        ;; IS IT A CONTROL-S?
7111 040720 001021          BNE     32$                   ;; BRANCH IF NO
7112 040722 005077 140216      CLR     $TKS                   ;; DISABLE TTY KEYBOARD INTERRUPTS
7113 040726 005726          TST     (SP)+                  ;; CLEAN CHAR OFF STACK
7114 040730 105777 140210      31$:   TSTB    $TKS                   ;; WAIT FOR A CHAR
7115 040734 100375          BPL     31$                   ;; LOOP UNTIL ITS THERE
    
```

E11

```

7116 040736 117746 140204      MOVB    2STKB,-(SP)      ;;GET THE CHARACTER
7117 040742 042716 177600      BIC     #1C177,(SP)    ;;MAKE IT 7-BIT ASCII
7118 040746 022627 000021      CMP     (SP)+,21      ;;IS IT A CONTROL-G?
7119 040752 001366          BNE     31$           ;;BRANCH IF NO
7120 040754 012777 000100 140162      MOV     #100,2STKS    ;;REENABLE TTY KEYBOARD INTERRUPTS
7121 040762 000002          RTI                    ;;RETURN
7122 040764 005237 040542      32$:   INC     $TKCNT    ;;COUNT THIS CHARACTER
7123 040770 021627 000140      CMP     (SP),#140    ;;IS IT UPPER CASE?
7124 040774 002405          BLT     4$           ;;BRANCH IF YES
7125 040776 021627 000175      CMP     (SP),#175    ;;IS IT A SPECIAL CHAR?
7126 041002 003002          BGT     4$           ;;BRANCH IF YES
7127 041004 042716 000040      BIC     #40,(SP)     ;;MAKE IT UPPER CASE
7128 041010 112677 177530      4$:   MOVB    (SP)+,2STKQIN ;;AND PUT IT IN QUEUE
7129 041014 005237 040544      INC     $TKQIN      ;;UPDATE THE POINTER
7130 041020 023727 040544 040551      CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
7131 041026 001003          BNE     5$           ;;BRANCH IF NO
7132 041030 012737 040550 040544      MOV     #2STKQSRRT,$TKQIN ;;RESET THE POINTER
7133 041036 000002      5$:   RTI                    ;;RETURN
  
```

```

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

```

7140 041040 022737 000176 001140 $CKSWR: CMP     #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
7141 041046 001124          BNE     15$         ;;EXIT IF NOT
7142 041050 105777 140070      TSTB   2STKS       ;;IS A CHAR WAITING?
7143 041054 100121          BPL     15$         ;;IF NOT, EXIT
7144 041056 117746 140064      MOVB   2STKB,-(SP) ;;YES
7145 041062 042716 177600      BIC    #1C177,(SP) ;;MAKE IT 7-BIT ASCII
7146 041066 021627 000007      CMP    (SP),#7     ;;IS IT A CONTROL-G?
7147 041072 001300          BNE     2$         ;;IF NOT, PUT IT IN THE TTY QUEUE
7148          ;;AND EXIT
  
```

```

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

```

7154 041074 123727 001134 000001 6$:   CMPB   $AUTOB,#1    ;;ARE WE RUNNING IN AUTO-MODE?
7155 041102 001674          BEQ     2$         ;;BRANCH IF YES
7156 041104 005726          TST    (SP)+      ;;CLEAR CONTROL-G OFF STACK
7157 041106 004737 040552      JSR    PC,$TKINT  ;;FLUSH THE TTY INPUT QUEUE
7158 041112 005077 140026          CLR    2STKS     ;;DISABLE TTY KEYBOARD INTERRUPTS
7159 041116 112737 000001 001135      MOVB   #1,$INTAG  ;;SET INTERRUPT MODE INDICATOR
  
```

```

7161 041124 104401 041750      TYPE   ,SNTLGL     ;;ECHO THE CONTROL-G (1G)
7162 041130 104401 041755      SGTSWR: TYPE   $MSWR    ;;TYPE CURRENT CONTENTS
7163 041134 013746 000176      MOV    $SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
7164 041140 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7165 041142 104401 041766      TYPE   ,SMNEW     ;;PROMPT FOR NEW SWR
7166 041146 005046      19$:   CLR    -(SP)     ;;CLEAR COUNTER
7167 041150 005046          CLR    -(SP)     ;;THE NEW SWR
7168 041152 105777 137766      7$:   TSTB   2STKS     ;;CHAR THERE?
7169 041156 100375          BPL     7$       ;;IF NOT TRY AGAIN
  
```

```

7171 041160 117746 137762      MOVB   2STKB,-(SP) ;;PICK UP CHAR
  
```



```

7228
7229 041402 011646          SRDCHR: MOV      (SP),-(SP,      ;; PUSH DOWN THE PC AND
7230 041404 016666 000004 000002 MOV      4(SP),2(SP,  ;; THE PS
7231 041412 005066 000004 CLR      4(SP)      ;; GET READY FOR A CHARACTER
7232 041416 005046 CLR      -(SP)      ;; PUT NEW PS ON STACK
7233 041420 012746 041426 MOV      #645,-(SP)  ;; PUT NEW PC ON STACK
7234 041424 000002 RTI          ;; POP NEW PC AND PS
7235 041426
7236 041426 005737 040542 64$: TST      $TKCNT      ;; WAIT ON A CHARACTER
7237 041430 001775 1$: BEQ      1$
7238 041434 005337 040542 DEC      $TKCNT      ;; DECREMENT THE COUNTER
7239 041440 117766 177102 000004 MOV      $TKQOUT,4(SP) ;; GET ONE CHARACTER
7240 041446 005237 040546 INC      $TKQOUT      ;; UPDATE THE POINTER
7241 041450 023727 040546 040551 CMP      $TKQOUT,$TKQEND ;; DID IT GO OFF OF THE END?
7242 041456 001003 BNE      2$        ;; BRANCH IF NO
7243 041460 012737 040550 040546 MOV      $TKQSRRT,$TKQOUT ;; RESET THE POINTER
7244 041470 000002 2$: RTI          ;; RETURN
7245
7246 *****
7247 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
7248 *CALL:
7249 *
7250 * RDLIN          ;; INPUT A STRING FROM THE TTY
7251 * RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
7252 *              ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
7253
7254 SRDLIN: MOV      R3,-(SP)  ;; SAVE R3
7255 CLR      -(SP)          ;; CLEAR THE RUBOUT KEY
7256 1$: MOV      #STTYIN,R3   ;; GET ADDRESS
7257 2$: CMP      #STTYIN+8.,R3 ;; BUFFER FULL?
7258 BLOS      4$           ;; BR IF YES
7259 RDCHR      ;; GO READ ONE CHARACTER FROM THE TTY
7260 MOV      (SP)+,(R3)     ;; GET CHARACTER
7261 CMP      #177,(R3)     ;; IS IT A RUBOUT
7262 BNE      5$           ;; BR IF NO
7263 TST      (SP)          ;; IS THIS THE FIRST RUBOUT?
7264 BNE      6$           ;; BR IF NO
7265 MOV      #'\.9$,R3     ;; TYPE A BACK SLASH
7266 10$: MOV      #-1,(SP)  ;; SET THE RUBOUT KEY
7267 DEC      R3           ;; BACKUP BY ONE
7268 CMP      R3,#STTYIN   ;; STACK EMPTY?
7269 BLO      4$           ;; BR IF YES
7270 MOV      (R3),9$      ;; SETUP TO TYPEOUT THE DELETED CHAR.
7271 TYPE     9$          ;; GO TYPE
7272 BR      2$           ;; GO READ ANOTHER CHAR.
7273 5$: TST      (SP)      ;; RUBOUT KEY SET?
7274 BEQ      7$          ;; BR IF NO
7275 MOV      #'\.9$,R3     ;; TYPE A BACK SLASH
7276 TYPE     9$
7277 CLR      (SP)         ;; CLEAR THE RUBOUT KEY
7278 7$: CMP      #25,(R3)  ;; IS CHARACTER A CTRL U?
7279 BNE      8$          ;; BR IF NO
7280 TYPE     $CNTLU      ;; TYPE A CONTROL "U"
7281 BR      1$          ;; GO START OVER
7282 8$: CMP      #22,(R3)  ;; IS CHARACTER A "r"?
7283 BNE      3$          ;; BRANCH IF NO
7284 CLAB     (R3)        ;; CLEAR THE CHARACTER

```

```

7284 041632 104401 001211 TYPE .SCLF ;;TYPE A "CR" & "LF"
7285 041636 104401 041726 TYPE $TTYIN ;;TYPE THE INPUT STRING
7286 041642 000717 BR 2$ ;;GO PICKUP ANOTHER CHARACTER
7287 041644 104401 001210 4$: TYPE $QJES ;;TYPE A "?"
7288 041650 000712 BR 1$ ;;CLEAR THE BUFFER AND LOOP
7289 041652 111337 041724 3$: MOVB (R3),9$ ;;ECHO THE CHARACTER
7290 041656 104401 041724 TYPE 9$
7291 041662 122723 000015 CMPB #15,(R3)+ ;;CHECK FOR RETURN
7292 041666 001305 BNE 2$ ;;LOOP IF NOT RETURN
7293 041670 105063 177777 CLRB -1(R3) ;;CLEAR RETURN (THE 15)
7294 041674 104401 001212 TYPE $LF ;;TYPE A LINE FEED
7295 041700 035726 TST (SP)+ ;;CLEAN RUBOUT KEY FROM THE STACK
7296 041702 012603 MOV (SP)+,R3 ;;RESTORE R3
7297 041704 011646 MOV (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
7298 041706 016666 000004 000002 MOV 4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
7299 041714 012766 041726 000004 MOV $TTYIN,4(SP)
7300 041722 000002 RTI ;;RETURN
7301 041724 000 9$: .BYTE 0 ;;STORAGE FOR ASCII CHAR. TO TYPE
7302 041725 000 .BYTE 0 ;;TERMINATOR
7303 041726 000010 $TTYIN: .BLKB 8. ;;RESERVE 8 BYTES FOR TTY INPUT
7304 041736 041536 005015 000 $CNTLC: .ASCIZ /@C/<15><12> ;;CONTROL "C"
7305 041743 136 006525 000012 $CNTLU: .ASCIZ /@U/<15><12> ;;CONTROL "U"
7306 041750 043536 005015 000 $CNTLG: .ASCIZ /@G/<15><12> ;;CONTROL "G"
7307 041755 015 051412 051127 $MSWR: .ASCIZ <15><12>/SWR = /
7308 041762 036440 000040
7309 041766 020040 042516 020127 $MNEW: .ASCIZ / NEW = /
7310 041774 020075 000
7311 .EVEN
7312 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
7313
7314 ;;*****
7315 ;;THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
7316 ;;CHANGE IT TO BINARY.
7317 ;;THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
7318 ;;OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
7319 ;;FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
7320 ;;THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
7321 ;;CALL:
7322 ;; * RDOCT ;;READ AN OCTAL NUMBER
7323 ;; * RETURN HERE ;;LOW ORDER BITS ARE ON TOP OF THE STACK
7324 ;; * ;;HIGH ORDER BITS ARE IN $HIOCT
7325
7326 042000 011646 000004 000002 $RDOCT: MOV (SP),-(SP) ;;PROVIDE SPACE FOR THE
7327 042002 016666 000004 000002 MOV 4(SP),2(SP) ;;INPUT NUMBER
7328 042010 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
7329 042012 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
7330 042014 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
7331 042016 104411 1$: RDLIN ;;READ AN ASCIZ LINE
7332 042020 012600 MOV (SP)+,R0 ;;GET ADDRESS OF 1ST CHARACTER
7333 042022 010037 042126 MOV R0,$$ ;;AND SAVE IT
7334 042026 005001 CLR R1 ;;CLEAR DATA WORD
7335 042030 005002 CLR R2
7336 042032 112046 2$: MOVB (R0)+,-(SP) ;;PICKUP THIS CHARACTER
7337 042034 001420 BEQ 3$ ;;IF ZERO GET OUT
7338 042036 122716 000060 CMPB #'0,(SP) ;;MAKE SURE THIS CHARACTER
7339 042042 003026 BGT 4$ ;;IS AN OCTAL DIGIT

```

```

7340 042044 122716 000067
7341 042050 002423
7342 042052 006301
7343 042054 006102
7344 042056 006301
7345 042060 006102
7346 042062 006301
7347 042064 006102
7348 042066 042716 177770
7349 042072 062601
7350 042074 000756
7351 042076 005726
7352 042100 010166 000012
7353 042104 010237 042136
7354 042110 012602
7355 042112 012601
7356 042114 012600
7357 042116 000002
7358 042120 005726
7359 042122 105010
7360 042124 104401
7361 042126 000000
7362 042130 104401 001210
7363 042132 000730
7364 042134 000000
7365 042136 000000
7366
7367
7368
7369
7370
7371
7372
7373
7374
7375
7376
7377
7378
7379
7380
7381
7382 042140
7383 042140 010046
7384 042142 010146
7385 042144 010246
7386 042146 010346
7387 042150 010446
7388 042152 010546
7389 042154 016646 000022
7390 042160 016646 000022
7391 042164 016646 000022
7392 042170 016646 000022
7393 042174 000002
7394
7395

```

```

CMPB #7,(SP)
BLT 45
ASL R1 ::*2
ROL R2
ASL R1 ::*4
ROL R2
ASL R1 ::*8
ROL R2
BIC #07,(SP) ::STRIP THE ASCII JUNK
ADD (SP)+,R1 ::ADD IN THIS DIGIT
BR 25 ::LOOP
35: TST (SP)+ ::CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ::SAVE THE RESULT
MOV R2,$HIOCT
MOV (SP)+,R2 ::POP STACK INTO R2
MOV (SP)+,R1 ::POP STACK INTO R1
MOV (SP)+,R0 ::POP STACK INTO R0
RTI ::RETURN
45: TST (SP)+ ::CLEAN PARTIAL FROM STACK
CLRB (R0) ::SET A TERMINATOR
TYPE TYPE UP THRU THE BAD CHAR.
55: .WORD 0
TYPE $QUES ::"? " "CR" & "LF"
BR 15 ::TRY AGAIN
$HIOCT: .WORD 0 ::HIGH ORDER BITS GO HERE
.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

*SAVE R0-R5
*CALL:

SAVE REG
\$SAVREG THE STACK WILL LOOK LIKE:

- *TOP---
- * +2---(+18)
- * +4---R5
- * +6---R4
- * +8---R3
- * +10---R2
- * +12---R1
- * +14---R0

```

$SAVREG:
MOV R0,-(SP) JSH R0 ON STACK
MOV R1,-(SP) JSH R1 ON STACK
MOV R2,-(SP) JSH R2 ON STACK
MOV R3,-(SP) JSH R3 ON STACK
MOV R4,-(SP) JSH R4 ON STACK
MOV R5,-(SP) JSH R5 ON STACK
MOV 22(SP),-(SP) ::SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ::SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ::SAVE PC OF CALL
MOV 22(SP),-(SP) ::SAVE PC OF CALL
RTI

```

*RESTORE R0-R5

```

7396 ;*CALL:
7397 ;* RESREG
7398 $RESREG:
7399 042176 012666 000022 MOV (SP)+,R2(SP) ;:RESTORE PC OF CALL
7400 042202 012666 000022 MOV (SP)+,R2(SP) ;:RESTORE PS OF CALL
7401 042206 012666 000022 MOV (SP)+,R2(SP) ;:RESTORE PC OF MAIN FLOW
7402 042212 012666 000022 MOV (SP)+,R2(SP) ;:RESTORE PS OF MAIN FLOW
7403 042216 012605 MOV (SP)+,R5 ;:POP STACK INTO R5
7404 042220 012604 MOV (SP)+,R4 ;:POP STACK INTO R4
7405 042222 012603 MOV (SP)+,R3 ;:POP STACK INTO R3
7406 042224 012602 MOV (SP)+,R2 ;:POP STACK INTO R2
7407 042226 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
7408 042230 012600 MOV (SP)+,R0 ;:POP STACK INTO R0
7409 042232 000002 RTI
7410
7411 .SBTTL POWER DOWN AND UP ROUTINE
7412
7413 ;:*****
7414
7415 :POWER DOWN ROUTINE
7416 042234 017737 136700 003312 $PWRDN: MOV @SWR,S@SWR ;SAVE SWITCH REGISTER
7417 042242 012737 042262 000024 MOV @PWRUP,P@RVEC ;SET UP VECTOR
7418 042250 012737 000340 000026 MOV @PR7,P@RVEC+2
7419 042256 000000 HALT
7420 042260 000776 BR -2 ;HANG UP
7421
7422 ;:*****
7423
7424 :POWER UP ROUTINE
7425 042262 005037 042356 $PWRUP: CLR $PWRCT ;LOAD WAIT COUNT
7426 042266 012737 000144 042360 MOV #100,$PWRCT+2
7427 042274 005237 042356 1$: INC $PWRCT ;WAIT FOR TELETYPE
7428 042300 001375 BNE 1$
7429 042302 005337 042360 DEC $PWRCT+2
7430 042306 001372 BNE 1$
7431 042310 012737 042234 000024 MOV @PWRDN,P@RVEC ;SET UP FOR POWER DOWN VECTOR
7432 042316 012737 000340 000026 MOV @PR7,P@RVEC+2
7433 042324 012706 001100 MOV @STACK,SP ;FORCE STACK
7434 042330 104401 042362 TYPE $POWER ;TYPE POWER
7435 042334 004737 034402 JSR PC,PARCHK ;REINITIALIZE MEMORY CHECK ENABLE
7436 042340 013777 003312 136572 MOV @S@SWR,@SWR ;RESTORE SWITCH REGISTER
7437 042346 013702 001270 MOV $BASE,R2 ;REINITIALISE R2 FOR '611 BASE
7438 042352 000177 136530 JMP @SLPADR ;GO BACK TO LAST TEST
7439
7440 042356 000000 000000 $PWRCT: .WORD 0,0 ;TELETYPE TIME OUT
7441 042362 047520 042527 000122 $POWER: .ASCIZ /POWER/
7442 .EVEN
7443 .SBTTL TRAP DECODER
7444
7445 ;:*****
7446 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
7447 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
7448 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
7449 ;*GO TO THAT ROUTINE.
7450
7451 042370 010046 $TRAP: MOV R0,-(SP) ;:SAVE R0

```

K11

```

7452 042372 016600 000002      MOV      2(SP),RO      ;;GET TRAP ADDRESS
7453 042376 005740             TST      -(RO)        ;;BACKUP BY 2
7454 042400 111000             MOVVB   (RO),RO       ;;GET RIGHT BYTE OF TRAP
7455 042402 006300             ASL     RO            ;;POSITION FOR INDEXING
7456 042404 016000 042424      MOV     $TRPAD(RO),RO ;;INDEX TO TABLE
7457 042410 000200             RTS      RO           ;;GO TO ROUTINE
  
```

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

7462 042412 011646 000004 000002 $TRAP2: MOV   (SP),-(SP)  ;;MOVE THE PC DOWN
7463 042414 016666             MOV     4(SP),2(SP)  ;;MOVE THE PSW DOWN
7464 042422 000002             RTI                    ;;RESTORE THE PSW
  
```

.SBTTL TRAP TABLE

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

			ROUTINE		

7473	042424	042412	\$TRPAD: .WORD \$TRAP2	TRAP+1(104401)	TTY TYPEOUT ROUTINE
7474	042426	037606	\$TYPE	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7475	042430	040114	\$TYPOC	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
7476	042432	040070	\$TYPOS	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
7477	042434	040130	\$TYPON	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
7478	042436	040316	\$TYPDS		
7479					
7480	042440	041130	\$GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
7481					
7482	042442	041040	\$CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
7483	042444	041402	\$RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
7484	042446	041472	\$RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
7485	042450	042030	\$RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
7486	042452	042140	\$SAVREG	TRAP+13(104413)	SAVE RO-R5 ROUTINE
7487	042454	042176	\$RESREG	TRAP+14(104414)	RESTORE RO-R5 ROUTINE
7488	042456	035304	\$SCOPI\$	TRAP+15(104415)	INTERNAL LOOP ON ERROR

.SBTTL DATA TABLE FOR PRINT OUT

7489									
7490									
7491	042460	001220	003244		DT000:	.WORD	STESTN, TRAPPC		
7492	042464	001220	001116	003224	DT002:	.WORD	STESTN, SERRPC, E.MR1, T.MR1, P1.BIT, PR.BIT, M1.BIT, M2.BIT, BITCNT		
7493	042472	003164	003252	003254					
7494	042500	003256	003260	003262					
7495	042506	001220	001116	003200	DT003:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.MR2, T.MR2, E.MR3, T.MR3		
7496	042514	003140	003226	003166					
7497	042522	003230	003170						
7498	042526	001220	001116	003200	DT041:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.CS2, T.CS2, E.ER, T.ER, E.BA, T.BA, E.WC, T.WC		
7499	042534	003140	003210	003150					
7500	042538	003214	003154	003204					
7501	042550	003144	003202	003142					
7502	042556	001220	001116	003200	DT046:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.CS2, T.CS2, E.ER, T.ER		
7503	042564	003140	003210	003150					
7504	042572	003214	003154						
7505	042576	001220	001116	003222	DT051:	.WORD	STESTN, SERRPC, E.DB, T.DB, WRDCNT		
7506	042604	003162	003264						
7507	042610	001220	001116	003200	DT052:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, T.CS2, T.ER		
7508	042616	003140	003150	003154					
7509	042624	001220	001116	003224	DT053:	.WORD	STESTN, SERRPC, E.MR1, T.MR1		
7510	042632	003164							
7511	042634	001220	001116	003220	DT054:	.WORD	STESTN, SERRPC, E.DCYL, T.DCYL, E.DA, T.DA		
7512	042642	003160	003206	003146					
7513	042650	001220	001116	003200	DT071:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.CS2, T.CS2, E.ER, T.ER		
7514	042656	003140	003210	003150					
7515	042664	003214	003154						
7516	042670	055176	055200	055202		.WORD	VRCHDR, VRCHDR+2, VRCHDR+4		
7517	042676	001220	001116	003200	DT074:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.CS2, T.CS2, E.ER, T.ER, HDRCNT		
7518	042704	003140	003210	003150					
7519	042712	003214	003154	003310					
7520	042720	001220	001116	003224	DT077:	.WORD	STESTN, SERRPC, E.MR1, T.MR1, HDRCNT		
7521	042726	003164	003310						
7522	042732	001220	001116	003234	DT134:	.WORD	STESTN, SERRPC, E.ECPT, T.ECPT, BITCNT		
7523	042740	003174	003262						
7524	042744	001220	001116	003200	DT144:	.WORD	STESTN, SERRPC, E.CS1, T.CS1, E.CS2, T.CS2, E.ER, T.ER		
7525	042752	003140	003210	003150					
7526	042760	003214	003154						
7527	042764	003204	003144	003202		.WORD	E.BA, T.BA, E.WC, T.WC, E.DCYL, T.DCYL, E.DA, T.DA		
7528	042772	003142	003220	003160					
7529	043000	003206	003146						
7530	043004	001220	001116	003234	DT151:	.WORD	STESTN, SERRPC, E.ECPT, T.ECPT		
7531	043012	003174							

.SBTTL DATA FORMAT FOR PRINT OUT

7532					
7533					
7534	043014	000001		DF000:	.WORD 1
7535	043016	002	000		.BYTE 2,0
7536	043020	000005		DF002:	.WORD 5,0
7537	043022	000	000		.BYTE 0,0
7538	043024	044067			.WORD DH000A
7539	043026	000	000		.BYTE 0,0
7540	043030	044105			.WORD DH000B
7541	043032	002	000		.BYTE 2,0
7542	043034	044151			.WORD DH002A
7543	043036	000	000		.BYTE 0,0
7544	043040	044235			.WORD DH002B
7545	043042	007	000		.BYTE 7,0
7546	043044	000005		DF003:	.WORD 5
7547	043046	000	000		.BYTE 0,0
7548	043050	044067			.WORD DH000A
7549	043052	000	000		.BYTE 0,0
7550	043054	044105			.WORD DH000B
7551	043056	002	000		.BYTE 2,0
7552	043060	044323			.WORD DH003A
7553	043062	000	000		.BYTE 0,0
7554	043064	044402			.WORD DH003B
7555	043066	006	000		.BYTE 6,0
7556	043070	000005		DF025:	.WORD 5
7557	043072	000	000		.BYTE 0,0
7558	043074	044067			.WORD DH000A
7559	043076	000	000		.BYTE 0,0
7560	043100	044105			.WORD DH000B
7561	043102	002	000		.BYTE 2,0
7562	043104	044461			.WORD DH025A
7563	043106	000	000		.BYTE 0,0
7564	043110	044540			.WORD DH025B
7565	043112	006	000		.BYTE 6,0
7566	043114	000007		DF041:	.WORD 7
7567	043116	000	000		.BYTE 0,0
7568	043120	044067			.WORD DH000A
7569	043122	000	000		.BYTE 0,0
7570	043124	044105			.WORD DH000B
7571	043126	002	000		.BYTE 2,0
7572	043130	044620			.WORD DH041A
7573	043132	000	000		.BYTE 0,0
7574	043134	044677			.WORD DH041B
7575	043136	006	000		.BYTE 6,0
7576	043140	044754			.WORD DH041C
7577	043142	000	000		.BYTE 0,0
7578	043144	045013			.WORD DH041D
7579	043146	004	000		.BYTE 4,0
7580	043150	000005		DF046:	.WORD 5
7581	043152	000	000		.BYTE 0,0
7582	043154	044067			.WORD DH000A
7583	043156	000	000		.BYTE 0,0
7584	043160	044105			.WORD DH000B
7585	043162	002	000		.BYTE 2,0
7586	043164	044620			.WORD DH041A
7587	043166	000	000		.BYTE 0,0

:ERROR 3-24

:ERROR 25-40

7588	043170	044677		.WORD	DH041B	
7589	043172	006	000	.BYTE	6,0	
7590	043174	000005		DF051: .WORD	5	
7591	043176	000	000	.BYTE	0,0	
7592	043200	044067		.WORD	DH000A	
7593	043202	000	000	.BYTE	0,0	
7594	043204	044105		.WORD	DH000B	
7595	043206	002	000	.BYTE	2,0	
7596	043210	045050		.WORD	DH051A	
7597	043212	000	000	.BYTE	0,0	
7598	043214	045075		.WORD	DH051B	
7599	043216	003	000	.BYTE	3,0	
7600	043220	000005		DF052: .WORD	5	
7601	043222	000	000	.BYTE	0,0	
7602	043224	044067		.WORD	DH000A	
7603	043226	000	000	.BYTE	0,0	
7604	043230	044105		.WORD	DH000B	
7605	043232	002	000	.BYTE	2,0	
7606	043234	045123		.WORD	DH052A	
7607	043236	000	000	.BYTE	0,0	
7608	043240	045142		.WORD	DH052B	
7609	043242	004	000	.BYTE	4,0	
7610	043244	000005		DF053: .WORD	5	
7611	043246	000	000	.BYTE	0,0	
7612	043250	044067		.WORD	DH000A	
7613	043252	000	000	.BYTE	0,0	
7614	043254	044105		.WORD	DH000B	
7615	043256	002	000	.BYTE	2,0	
7616	043260	045177		.WORD	DH053A	
7617	043262	000	000	.BYTE	0,0	
7618	043264	045216		.WORD	DH053B	
7619	043266	002	000	.BYTE	2,0	
7620	043270	000005		DF054: .WORD	5	;ERROR-54-61
7621	043272	000	000	.BYTE	0,0	
7622	043274	044067		.WORD	DH000A	
7623	043276	000	000	.BYTE	0,0	
7624	043300	044105		.WORD	DH000B	
7625	043302	002	000	.BYTE	2,0	
7626	043304	045234		.WORD	DH054A	
7627	043306	000	000	.BYTE	0,0	
7628	043310	045273		.WORD	DH054B	
7629	043312	004	000	.BYTE	4,0	
7630	043314	000007		DF071: .WORD	7	;ERRORS 71-73
7631	043316	000	000	.BYTE	0,0	
7632	043320	044067		.WORD	DH000A	
7633	043322	000	000	.BYTE	0,0	
7634	043324	044105		.WORD	DH000B	
7635	043326	002	000	.BYTE	2,0	
7636	043330	044620		.WORD	DH041A	
7637	043332	000	000	.BYTE	0,0	
7638	043334	044677		.WORD	DH041B	
7639	043336	006	000	.BYTE	6,0	
7640	043340	045330		.WORD	DH071A	
7641	043342	000	000	.BYTE	0,0	
7642	043344	045351		.WORD	DH071B	
7643	043346	003	000	.BYTE	3,0	

7644	043350	000005		DF074:	.WORD	5
7645	043352	000	000		.BYTE	0,0
7646	043354	044067			.WORD	DH000A
7647	043356	044067	000		.BYTE	0,0
7648	043358	044067	000		.WORD	DH000B
7649	043360	044067	000		.BYTE	2,0
7650	043362	045408			.WORD	DH074A
7651	043364	045408	000		.BYTE	0,0
7652	043366	045467			.WORD	DH074B
7653	043368	045467	000		.BYTE	3,0
7654	043370	000005		DF077:	.WORD	5
7655	043372	000	000		.BYTE	0,0
7656	043374	044067			.WORD	DH000A
7657	043376	044067	000		.BYTE	0,0
7658	043378	044067	000		.WORD	DH000B
7659	043380	044067	000		.BYTE	2,0
7660	043382	045553			.WORD	DH077A
7661	043384	045553	000		.BYTE	0,0
7662	043386	045602			.WORD	DH077B
7663	043388	045602	000		.BYTE	3,0
7664	043390	000005		DF134:	.WORD	5
7665	043392	000	000		.BYTE	0,0
7666	043394	044067			.WORD	DH000A
7667	043396	044067	000		.BYTE	0,0
7668	043398	044067	000		.WORD	DH000B
7669	043400	044067	000		.BYTE	2,0
7670	043402	045626			.WORD	DH134A
7671	043404	045626	000		.BYTE	0,0
7672	043406	045652			.WORD	DH134B
7673	043408	045652	000		.BYTE	3,0
7674	043410	000007		DF144:	.WORD	7
7675	043412	000	000		.BYTE	0,0
7676	043414	044067			.WORD	DH000A
7677	043416	044067	000		.BYTE	0,0
7678	043418	044067	000		.WORD	DH000B
7679	043420	044067	000		.BYTE	2,0
7680	043422	044620			.WORD	DH041A
7681	043424	044620	000		.BYTE	0,0
7682	043426	044677			.WORD	DH041B
7683	043428	044677	000		.BYTE	6,0
7684	043430	045700			.WORD	DH144A
7685	043432	045700	000		.BYTE	0,0
7686	043434	045777			.WORD	DH144B
7687	043436	045777	000		.BYTE	10,0
7688	043438	010		DF151:	.WORD	5
7689	043440	000005	000		.BYTE	0,0
7690	043442	000	000		.WORD	DH000A
7691	043444	044067			.BYTE	0,0
7692	043446	000	000		.WORD	DH000B
7693	043448	044067			.BYTE	2,0
7694	043450	044105			.WORD	DH052A
7695	043452	044105	000		.BYTE	0,0
7696	043454	045123			.WORD	DH151A
7697	043456	045123	000		.BYTE	2,0
7698	043458	046074			.WORD	DH151A
7699	043460	046074	000		.BYTE	2,0

7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744

005015 045522 030466
020061 052502 020123
042101 051104 051505
020123 020050 000
040 020051 020075
000
122 033113 030461
053040 041505 047524
020122 042101 051104
051505 020123 020050
000
122 033113 030461
050040 044522 051117
052111 020131 020050
000
015 025012 025052
025052 020040 051120
043517 040522 020115
040510 052114 042105
025040 025052 025052
005015 000
015 051412 041505
047117 020104 040520
051523 051040 047125
052040 046511 020105
051511 040440 050120
047522 020130 035063
032461 046440 047111
052125 051505 005015
000
040 000040
005015 051120 043517
040522 020115 041101
051117 042524 020104
042502 040503 051525
020105 051105 047522
020122 044124 042522
044123 046117 020104
054105 042503 042105
042105 005015 000
015 052012 051505
020124 000
04C 054502 040520
051523 042105 005015
000

.SBTTL ASCII MESSAGES

OPR001: .ASCIZ <15><12>/RK611 BUS ADDRESS (/
OPR002: .ASCIZ /) = /
OPR003: .ASCIZ /RK611 VECTOR ADDRESS (/
OPR004: .ASCIZ /RK611 PRIORITY (/
OPR005: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
OPR006: .ASCIZ <15><12>/SECOND PASS RUN TIME IS APPROX 3:15 MINUTES/<15><12>
SPACE2: .ASCIZ / /
ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
TSTBY1: .ASCIZ <15><12>/TEST /
TSTBY2: .ASCIZ / BYPASSED/<15><12>

		.SBTTL DATA HEADERS													
7745															
7746															
7747	044067	124	051505	020124	DH000A: .ASCIZ	/TEST	ERRGR/								
7748	044074	020040	042440	051122											
7749	044102	051117	000												
7750	044105	116	046525	020040	DH000B: .ASCIZ	/NUM	PC/								
7751	044112	020040	050040	000103											
7752	044120	042524	052123	020040	DH000C: .ASCII	/TEST	TRAP<<15><12>								
7753	044126	020040	051124	050101											
7754	044134	005015													
7755	044136	052516	020115	020040		.ASCIZ	/NUM	PC/							
7756	044144	020040	041520	000											
7757	044151	105	050130	041505	DH002A: .ASCIZ	/EXPECT	ACTUAL	PRESENT	PRESENT	PRESENT	PRESENT	PRESENT	PRESENT	BIT/	
7758	044156	020124	040440	052103											
7759	044164	040525	020114	050040											
7760	044172	042522	042523	052116											
7761	044200	050040	042522	042523											
7762	044206	052116	050040	042522											
7763	044214	042523	052116	050040											
7764	044222	042522	042523	052116											
7765	044230	041040	052111	000											
7766	044235	122	046513	030522	DH002B: .ASCIZ	/RKMR1	RKMR1	BIT+1	BIT	BIT-1	BIT-2	COUNT/			
7767	044242	020040	051040	046513											
7768	044250	030522	020040	041040											
7769	044256	052111	030453	020040											
7770	044264	041040	052111	020040											
7771	044272	020040	041040	052111											
7772	044300	030455	020040	041040											
7773	044306	052111	031055	020040											
7774	044314	041440	052517	052116											
7775	044322	000													
7776	044323	105	050130	041505	DH003A: .ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT	ACTUAL/				
7777	044330	020124	040440	052103											
7778	044336	040525	020114	042440											
7779	044344	050130	041505	020124											
7780	044352	040440	052103	040525											
7781	044360	020114	042440	050130											
7782	044366	041505	020124	040440											
7783	044374	052103	040525	000114											
7784	044402	045522	051503	020061	DH003B: .ASCIZ	/RKCS1	RKCS1	MESS A	MESS A	MESS B	MESS B/				
7785	044410	020040	045522	051503											
7786	044416	020061	020040	042515											
7787	044424	051523	040440	020040											
7788	044432	042515	051523	040440											
7789	044440	020040	042515	051523											
7790	044446	041040	020040	042515											
7791	044454	051523	041040	000											
7792	044461	105	050130	041505	DH025A: .ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT	ACTUAL/				
7793	044466	020124	040440	052103											
7794	044474	040525	020114	042440											
7795	044502	050130	041505	020124											
7796	044510	040440	052103	040525											
7797	044516	020114	042440	050130											
7798	044524	041505	020124	040440											
7799	044532	052103	040525	000114											
7800	044540	045522	051503	020061	DH025B: .ASCIZ	/RKCS1	RKCS1	HD WD 1	HD WD 1	HD WD 2	HD WD 2/				

7913	045722	042520	052103	020040
7914	045730	041501	052524	046101
7915	045736	020040	054105	042520
7916	045744	052103	020040	041501
7917	045752	052524	046101	020040
7918	045760	054105	042520	052103
7919	045766	020040	041501	052524
7920	045774	046101	000	
7921	045777	122	041113	020101
7922	046004	020040	051040	041113
7923	046012	020101	020040	051040
7924	046020	053513	020103	020040
7925	046026	051040	053513	020103
7926	046034	020040	051040	042113
7927	046042	054503	020114	051040
7928	046050	042113	054503	020114
7929	046056	051040	042113	020101
7930	046064	020040	051040	042113
7931	046072	000101		
7932	046074	045522	041505	052120
7933	046102	020040	045522	041505
7934	046110	052120	000	

DH144B: .ASCIZ /RKBA RKBA RKWC RKWC RKDCYL RKDCYL RKDA RKDA/

DH151A: .ASCIZ {RKECPT RKECPT.

.SBTTL ERROR MESSAGES

7935						
7936						
7937	046113	125	042516	050130	EM000:	.ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
7938	046120	041505	042524	020104		
7939	046126	042515	047515	054522		
7940	046134	050040	051101	052111		
7941	046142	020131	047105	041101		
7942	046150	042514	052040	040522		
7943	046156	000120				
7944	046160	052101	042524	050115	EM300:	.ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM READ DATA/
7945	046166	044524	043516	052040		
7946	046174	020117	044103	041505		
7947	046202	020113	042523	045505		
7948	046210	046440	051505	020123		
7949	046216	051106	046517	051040		
7950	046224	040505	020104	040504		
7951	046232	040524	000			
7952	046235	101	052124	046505	EM301:	.ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE DATA/
7953	046242	052120	047111	020107		
7954	046250	047524	041440	042510		
7955	046256	045503	051440	042505		
7956	046264	020113	042515	051523		
7957	046272	043040	047522	020115		
7958	046300	051127	052111	020105		
7959	046306	040504	040524	000		
7960	046313	101	052124	046505	EM302:	.ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE CHECK/
7961	046320	052120	047111	020107		
7962	046326	047524	041440	042510		
7963	046334	045503	051440	042505		
7964	046342	020113	042515	051523		
7965	046350	043040	047522	020115		
7966	046356	051127	052111	020105		
7967	046364	044103	041505	000113		
7968	046372	052101	042524	050115	EM303:	.ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM READ DATA/
7969	046400	044524	043516	052040		
7970	046406	020117	044103	041505		
7971	046414	020113	046103	040505		
7972	046422	020122	042515	051523		
7973	046430	043040	047522	020115		
7974	046436	042522	042101	042040		
7975	046444	052101	000101			
7976	046450	052101	042524	050115	EM304:	.ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE DATA/
7977	046456	044524	043516	052040		
7978	046464	020117	044103	041505		
7979	046472	020113	046103	040505		
7980	046500	020122	042515	051523		
7981	046506	043040	047522	020115		
7982	046514	051127	052111	020105		
7983	046522	040504	040524	000		
7984	046527	101	052124	046505	EM305:	.ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE CHECK/
7985	046534	052120	047111	020107		
7986	046542	047524	041440	042510		
7987	046550	045503	041440	042514		
7988	046556	051101	046440	051505		
7989	046564	020123	051106	046517		
7990	046572	053440	044522	042524		

7991	046600	041440	042510	045503	
7992	046606	000			
7993	046607	101	052124	046505	EM306: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
7994	046614	052120	047111	020107	
7995	046622	047524	041440	042510	
7996	046630	045503	044040	040505	
7997	046636	020104	042507	042516	
7998	046644	040522	044524	047117	
7999	046652	005015			
8000	046654	044527	044124	053040	.ASCIZ /WITH VARIOUS CYLINDER VALUES/
8001	046662	051101	047511	051525	
8002	046670	041440	046131	047111	
8003	046676	042504	020122	040526	
8004	046704	052514	051505	000	
8005	046711	101	052124	046505	EM307: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
8006	046716	052120	047111	020107	
8007	046724	047524	041440	042510	
8008	046732	045503	044040	040505	
8009	046740	020104	042507	042516	
8010	046746	040522	044524	047117	
8011	046754	005015			
8012	046756	044527	044124	053040	.ASCIZ /WITH VARIOUS TRACK VALUES/
8013	046764	051101	047511	051525	
8014	046772	052040	040522	045503	
8015	047000	053040	046101	042525	
8016	047006	000123			
8017	047010	052101	042524	050115	EM308: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
8018	047016	044524	043516	052040	
8019	047024	020117	044103	041505	
8020	047032	020113	042510	042101	
8021	047040	043440	047105	051105	
8022	047046	052101	047511	006516	
8023	047054	012			
8024	047055	127	052111	020110	.ASCIZ /WITH VARIOUS SECTOR VALUES/
8025	047062	040526	044522	052517	
8026	047070	020123	042523	052103	
8027	047076	051117	053040	046101	
8028	047104	042525	000123		
8029	047110	052101	042524	050115	EM309: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
8030	047116	044524	043516	052040	
8031	047124	020117	044103	041505	
8032	047132	020113	042510	042101	
8033	047140	043440	047105	051105	
8034	047146	052101	047511	006516	
8035	047154	012			
8036	047155	127	052111	020110	.ASCIZ /WITH VARIOUS FORMAT VALUES/
8037	047162	040526	044522	052517	
8038	047170	020123	047506	046522	
8039	047176	052101	053040	046101	
8040	047204	042525	000123		
8041	047210	052101	042524	050115	EM310: .ASCIZ /ATTEMPTING TO CHECK NPR DATA TRANSFER FOR WRITE DATA/
8042	047216	044524	043516	052040	
8043	047224	020117	044103	041505	
8044	047232	020113	050116	020122	
8045	047240	040504	040524	052040	
8046	047246	040522	051516	042506	

8047	047254	020122	047506	020122
8048	047262	051127	052111	020105
8049	047270	040504	040524	000
8050	047275	101	052124	046505
8051	047302	052120	047111	020107
8052	047310	047524	041440	042510
8053	047316	045503	044040	040505
8054	047324	042504	020122	042522
8055	047332	047503	047107	052111
8056	047340	047511	000116	
8057	047344	052101	042524	050115
8058	047352	044524	043516	052040
8059	047359	020117	044103	041505
8060	047367	020113	042523	052103
8061	047374	044440	044440	041516
8062	047402	051115	052116	
8063	047410			
8064	047411	101	046505	EM313: .ASCIZ /ATTEMPTING TO CHECK TRACK INCREMENT/
8065	047416	052120	047111	020107
8066	047424	047524	041440	042510
8067	047432	045503	052040	
8068	047440	045503	044440	041516
8069	047446	042522	042515	052116
8070	047454	000		
8071	047455	101	052124	046505
8072	047462	052120	047111	020107
8073	047470	047524	041440	042510
8074	047476	045503	041440	046131
8075	047504	047111	042504	020122
8076	047512	047111	051103	046505
8077	047520	047105	000124	
8078	047524	052101	042524	050115
8079	047532	044524	043516	052040
8080	047540	020117	044103	041505
8081	047546	020113	042523	052103
8082	047554	051117	050040	046125
8083	047562	042523	042040	052105
8084	047570	041505	044524	047117
8085	047576	005015		
8086	047600	044527	044124	053440
8087	047606	044522	042524	042040
8088	047614	052101	000101	
8089	047620	052101	042524	050115
8090	047626	044524	043516	052040
8091	047634	020117	047506	041522
8092	047642	020105	040502	020104
8093	047650	042523	052103	051117
8094	047656	042440	051122	051117
8095	047664	000		
8096	047665	101	052124	046505
8097	047672	052120	047111	020107
8098	047700	047524	043040	051117
8099	047706	042503	044040	040505
8100	047714	042504	020122	051126
8101	047722	020103	051105	047522
8102	047730	006522	012	

EM311: .ASCIZ /ATTEMPTING TO CHECK HEADER RECOGNITION/

EM312: .ASCIZ /ATTEMPTING TO CHECK SECTOR INCREMENT/

EM313: .ASCIZ /ATTEMPTING TO CHECK TRACK INCREMENT/

EM314: .ASCIZ /ATTEMPTING TO CHECK CYLINDER INCREMENT/

EM315: .ASCII /ATTEMPTING TO CHECK FOR PULSE DETECTION/<15><12>

.ASCIZ /WITH WRITE DATA/

EM316: .ASCIZ /ATTEMPTING TO FORCE BAD SECTOR ERROR

EM317: .ASCII /ATTEMPTING TO FORCE HEADER VRC ERROR/<15><12>

8103	047733	127	052111	020110	.ASCIZ	/WITH BAD SECTOR PRESENT/
8104	047740	040502	020104	042523		
8105	047746	052103	051117	050040		
8106	047754	042522	042523	052116		
8107	047762	000				
8108	047763	101	052124	046505	EM318:	.ASCIZ /ATTEMPTING TO FORCE HVRC ERROR/
8109	047770	052120	047111	020107		
8110	047776	047524	043040	051117		
8111	050004	042503	044040	051126		
8112	050012	020103	051105	047522		
8113	050020	000122				
8114	050022	052101	042524	050115	EM319:	.ASCIZ /ATTEMPTING TO FORCE OPERATION INCOMPLETE/
8115	050030	044524	043516	052040		
8116	050036	020117	047506	041522		
8117	050044	020105	050117	051105		
8118	050052	052101	047511	020116		
8119	050060	047111	047503	050115		
8120	050066	042514	042524	000		
8121	050073	101	052124	046505	EM320:	.ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37/
8122	050100	044520	043516	052040		
8123	050106	020117	047506	041522		
8124	050114	020105	050117	020111		
8125	050122	044527	044124	044040		
8126	050130	051126	020103	051105		
8127	050136	047522	020122	047117		
8128	050144	044040	040505	042504		
8129	050152	020122	033463	000		
8130	050157	101	052124	046505	EM321:	.ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36/
8131	050164	044520	043516	052040		
8132	050172	020117	047506	041522		
8133	050200	020105	050117	020111		
8134	050206	044527	044124	044040		
8135	050214	051126	020103	051105		
8136	050222	047522	020122	047117		
8137	050230	044040	040505	042504		
8138	050236	020122	033063	000		
8139	050243	101	052124	046505	EM322:	.ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0/
8140	050250	044520	043516	052040		
8141	050256	020117	047506	041522		
8142	050264	020105	050117	020111		
8143	050272	044527	044124	044040		
8144	050300	051126	020103	051105		
8145	050306	047522	020122	047117		
8146	050314	044040	040505	042504		
8147	050322	020122	000060			
8148	050326	044103	041505	044513	EM323:	.ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS BAD SECTOR ERROR/
8149	050334	043516	044040	040505		
8150	050342	047504	020122	042522		
8151	050350	047503	047107	052111		
8152	050356	047511	020116	044527		
8153	050364	044124	050040	042522		
8154	050372	044526	052517	020123		
8155	050400	040502	020104	042523		
8156	050406	052103	051117	042440		
8157	050414	051122	051117	000		
8158	050421	103	042510	045503	EM324:	.ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER VRC ERROR

ERROR MESSAGES

8215	051137	101	052124	046505	EM331: .ASCIZ /ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR/
8216	051144	052120	047111	020107	
8217	051152	040502	020104	042523	
8218	051160	052103	051117	042440	
8219	051166	051122	051117	051440	
8220	051174	052105	044524	043516	
8221	051202	041440	047117	051124	
8222	051210	046117	042514	020122	
8223	051216	051105	047522	000122	
8224	051224	052101	042524	050115	EM332: .ASCIZ /ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR/
8225	051232	044524	043516	044040	
8226	051240	040505	042504	020122	
8227	051246	051126	020103	051105	
8228	051254	047522	020122	042523	
8229	051262	052124	047111	020107	
8230	051270	047503	052116	047522	
8231	051276	046114	051105	042440	
8232	051304	051122	051117	000	
8233	051311	101	052124	046505	EM333: .ASCIZ /ATTEMPTING TO WRITE ECC WORDS/
8234	051316	052120	047111	020107	
8235	051324	047524	053440	044522	
8236	051332	042524	042440	041503	
8237	051340	053440	051117	051504	
8238	051346	000			
8239	051347	101	052124	046505	EM334: .ASCIZ /ATTEMPTING TO WRITE POSTAMBLE/
8240	051354	052120	047111	020107	
8241	051362	047524	053440	044522	
8242	051370	042524	050040	051517	
8243	051376	040524	041115	042514	
8244	051404	000			
8245	051405	101	052124	046505	EM335: .ASCIZ /ATTEMPTING COMPLETE EXECUTION OF WRITE DATA/
8246	051412	052120	047111	020107	
8247	051420	047503	050115	042514	
8248	051426	042524	042440	042530	
8249	051434	052503	044524	047117	
8250	051442	047440	020106	051127	
8251	051450	052111	020105	040504	
8252	051456	040524	000		
8253	051461	101	052124	046505	EM336: .ASCIZ /ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR/
8254	051466	052120	047111	020107	
8255	051474	051105	047522	020122	
8256	051502	046103	040505	020122	
8257	051510	044527	044124	041440	
8258	051516	047117	051124	046117	
8259	051524	042514	020122	046103	
8260	051532	040505	000122		
8261	051536	052101	042524	050115	EM337: .ASCIZ /ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL/
8262	051544	044524	043516	050040	
8263	051552	051101	044524	046101	
8264	051560	051440	041505	047524	
8265	051566	020122	051127	052111	
8266	051574	020105	044527	044124	
8267	051602	055040	051105	020117	
8268	051610	044506	046114	000	
8269	051615	101	052124	046505	EM338: .ASCIZ /ATTEMPTING TO WRITE 18 BIT DATA FIELD WITH WRITE DATA/
8270	051622	052120	047111	020107	

8271	051630	047524	053440	044522	
8272	051636	042524	030440	020070	
8273	051644	044502	020124	040504	
8274	051652	040524	043040	042511	
8275	051660	042114	053440	052111	
8276	051666	020110	051127	052111	
8277	051674	020105	040504	040524	
8278	051702	000			
8279	051703	101	052124	046505	EM339: .ASCII /ATTEMPTING TO WRITE BIT 16-17 OF 18 BIT/
8280	051710	052120	047111	020107	
8281	051716	047524	053440	044522	
8282	051724	042524	041040	052111	
8283	051732	030440	026466	033461	
8284	051740	047440	020106	034061	
8285	051746	041040	052111		
8286	051752	005015	040504	040524	.ASCIZ <15><12>/DATA FIELD WITH WRITE DATA/
8287	051760	043040	042511	042114	
8288	051766	053440	052111	020110	
8289	051774	051127	052111	020105	
8290	052002	040504	040524	000	
8291	052007	101	052124	046505	EM340: .ASCIZ /ATTEMPTING WRITE DATA IN 18 BIT MODE/
8292	052014	052120	047111	020107	
8293	052022	051127	052111	020105	
8294	052030	040504	040524	044440	
8295	052036	020116	034061	041040	
8296	052044	052111	046440	042117	
8297	052052	000105			
8298	052054	051503	020061	047111	EM4000: .ASCIZ /CS1 INCORRECT/
8299	052062	047503	051122	041505	
8300	052070	000124			
8301	052072	042515	051523	040440	EM4001: .ASCIZ /MESS A INCORRECT/
8302	052100	044440	041516	051117	
8303	052106	042522	052103	000	
8304	052113	115	051505	020123	EM4002: .ASCIZ /MESS B INCORRECT/
8305	052120	020102	047111	047503	
8306	052126	051122	041505	000124	
8307	052134	042510	042101	051105	EM4003: .ASCIZ /HEADER WORD 1 INCORRECT/
8308	052142	053440	051117	020104	
8309	052150	020061	047111	047503	
8310	052156	051122	041505	000124	
8311	052164	042510	042101	051105	EM4004: .ASCIZ /HEADER WORD 2 INCORRECT/
8312	052172	053440	051117	020104	
8313	052200	020062	047111	047503	
8314	052206	051122	041505	000124	
8315	052214	051503	020062	047111	EM4005: .ASCIZ /CS2 INCORRECT/
8316	052222	047503	051122	041505	
8317	052230	000124			
8318	052232	051105	047522	020122	EM4006: .ASCIZ /ERROR REG INCORRECT/
8319	052240	042522	020107	047111	
8320	052246	047503	051122	041505	
8321	052254	000124			
8322	052256	052502	020123	042101	EM4007: .ASCIZ /BUS ADDRESS INCORRECT/
8323	052264	051104	051505	020123	
8324	052272	047111	047503	051122	
8325	052300	041505	000124		
8326	052304	047527	042122	041440	EM4008: .ASCIZ /WORD COUNT INCORRECT/

0327	052312	052517	052116	044440	
0328	052322	041516	051117	042522	
0329	052326	052103	000		
0330	052331	033523	044440	EM4009: .ASCIZ /CSI INCORRECT AFTER READING DATA BUFFER/	
0331	052336	051516	051117	042522	
0332	052344	052103	040440	052106	
0333	052352	051105	051040	040505	
0334	052360	044504	043516	042040	
0335	052366	052101	020101	052502	
0336	052374	043106	051105	000	
0337	052401	041006	031123	044440	EM4010: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
0338	052406	041516	051117	042522	
0339	052414	052103	040440	052106	
0340	052422	051105	051040	040505	
0341	052430	044504	043516	042040	
0342	052436	052101	020101	052502	
0343	052444	043106	051105	000	
0344	052451	051122	051117	EM4011: .ASCIZ /ERROR REG INCORRECT AFTER READING DATA BUFFER/	
0345	052456	051040	043505	044440	
0346	052464	041516	051117	042522	
0347	052472	052103	040440	052106	
0348	052500	051105	051040	040505	
0349	052506	044504	043516	042040	
0350	052514	052101	020101	052502	
0351	052522	043106	051105	000	
0352	052527	041006	052101	020101	EM4012: .ASCIZ /DATA READ FROM MEMORY INCORRECT
0353	052534	042522	042101	043040	
0354	052542	047522	020115	042515	
0355	052550	047515	054522	044440	
0356	052556	041516	051117	042522	
0357	052564	052103	000		
0358	052567	04115	030522	044440	EM4013: .ASCIZ /MRI INCORRECT AFTER GAP IN WRITE DATA
0359	052574	041516	051117	042522	
0360	052602	052103	040440	052106	
0361	052610	051105	043440	050101	
0362	052616	044440	020116	051127	
0363	052624	052111	020105	040504	
0364	052632	040524	000		
0365	052635	04104	051511	020113	EM4014: .ASCIZ /DISK ADDRESS REG. INCORRECT
0366	052642	042101	051104	051505	
0367	052650	020123	042522	027107	
0368	052656	044440	041516	051117	
0369	052664	042522	052103	000	
0370	052671	04103	046131	047111	EM4015: .ASCIZ /CYLINDER ADDRESS REG. INCORRECT/
0371	052676	042504	020122	042101	
0372	052704	051104	051505	020123	
0373	052712	042522	027107	044440	
0374	052720	041516	051117	042522	
0375	052726	052103	000		
0376	052731	04115	030522	044440	EM4016: .ASCIZ /MRI INCORRECT/
0377	052736	041516	051117	042522	
0378	052744	052103	000		
0379	052747	04105	041503	050040	EM4017: .ASCIZ /ECC PAT REG INCORRECT/
0380	052754	052101	051040	043505	
0381	052762	044440	041516	051117	
0382	052770	042522	052103	000	

000000	000000	000000	000000
000001	000001	000001	000001
000002	000002	000002	000002
000003	000003	000003	000003
000004	000004	000004	000004
000005	000005	000005	000005
000006	000006	000006	000006
000007	000007	000007	000007
000008	000008	000008	000008
000009	000009	000009	000009
000010	000010	000010	000010
000011	000011	000011	000011
000012	000012	000012	000012
000013	000013	000013	000013
000014	000014	000014	000014
000015	000015	000015	000015
000016	000016	000016	000016
000017	000017	000017	000017
000018	000018	000018	000018
000019	000019	000019	000019
000020	000020	000020	000020
000021	000021	000021	000021
000022	000022	000022	000022
000023	000023	000023	000023
000024	000024	000024	000024
000025	000025	000025	000025
000026	000026	000026	000026
000027	000027	000027	000027
000028	000028	000028	000028
000029	000029	000029	000029
000030	000030	000030	000030
000031	000031	000031	000031
000032	000032	000032	000032
000033	000033	000033	000033
000034	000034	000034	000034
000035	000035	000035	000035
000036	000036	000036	000036
000037	000037	000037	000037
000038	000038	000038	000038
000039	000039	000039	000039
000040	000040	000040	000040
000041	000041	000041	000041
000042	000042	000042	000042
000043	000043	000043	000043
000044	000044	000044	000044
000045	000045	000045	000045
000046	000046	000046	000046
000047	000047	000047	000047
000048	000048	000048	000048
000049	000049	000049	000049
000050	000050	000050	000050
000051	000051	000051	000051
000052	000052	000052	000052
000053	000053	000053	000053
000054	000054	000054	000054
000055	000055	000055	000055
000056	000056	000056	000056
000057	000057	000057	000057
000058	000058	000058	000058
000059	000059	000059	000059
000060	000060	000060	000060
000061	000061	000061	000061
000062	000062	000062	000062
000063	000063	000063	000063
000064	000064	000064	000064
000065	000065	000065	000065
000066	000066	000066	000066
000067	000067	000067	000067
000068	000068	000068	000068
000069	000069	000069	000069
000070	000070	000070	000070
000071	000071	000071	000071
000072	000072	000072	000072
000073	000073	000073	000073
000074	000074	000074	000074
000075	000075	000075	000075
000076	000076	000076	000076
000077	000077	000077	000077
000078	000078	000078	000078
000079	000079	000079	000079
000080	000080	000080	000080
000081	000081	000081	000081
000082	000082	000082	000082
000083	000083	000083	000083
000084	000084	000084	000084
000085	000085	000085	000085
000086	000086	000086	000086
000087	000087	000087	000087
000088	000088	000088	000088
000089	000089	000089	000089
000090	000090	000090	000090
000091	000091	000091	000091
000092	000092	000092	000092
000093	000093	000093	000093
000094	000094	000094	000094
000095	000095	000095	000095
000096	000096	000096	000096
000097	000097	000097	000097
000098	000098	000098	000098
000099	000099	000099	000099
000100	000100	000100	000100

EMW2: .ASCIZ MRI INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/

EMW4: .ASCIZ MRI INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/

.SBTTL DATA BUFFERS

.EVEN
NPREUF:

053156	000	000
053156	001	001
053160	002	002
053162	003	003
053164	004	004
053166	005	005
053170	006	006
053172	007	007
053174	010	010
053176	011	011
053200	012	012
053202	013	013
053204	014	014
053206	015	015
053210	016	016
053212	017	017
053214	020	020
053216	021	021
053220	022	022
053222	023	023
053224	024	024
053226	025	025
053230	026	026
053232	027	027
053234	030	030
053236	031	031
053240	032	032
053242	033	033
053244	034	034
053246	035	035
053250	036	036
053252	037	037
053254	040	040
053256	041	041
053260	042	042
053262	043	043
053264	044	044
053266	045	045
053270	046	046
053272	047	047
053274	050	050
053276	051	051
053300	052	052
053302	053	053
053304	054	054
053306	055	055
053310	056	056
053312	057	057
053314	060	060
053316	061	061
053320	062	062
053322	063	063

.BYTE	0,0
.BYTE	1,1
.BYTE	2,2
.BYTE	3,3
.BYTE	4,4
.BYTE	5,5
.BYTE	6,6
.BYTE	7,7
.BYTE	10,10
.BYTE	11,11
.BYTE	12,12
.BYTE	13,13
.BYTE	14,14
.BYTE	15,15
.BYTE	16,16
.BYTE	17,17
.BYTE	20,20
.BYTE	21,21
.BYTE	22,22
.BYTE	23,23
.BYTE	24,24
.BYTE	25,25
.BYTE	26,26
.BYTE	27,27
.BYTE	30,30
.BYTE	31,31
.BYTE	32,32
.BYTE	33,33
.BYTE	34,34
.BYTE	35,35
.BYTE	36,36
.BYTE	37,37
.BYTE	40,40
.BYTE	41,41
.BYTE	42,42
.BYTE	43,43
.BYTE	44,44
.BYTE	45,45
.BYTE	46,46
.BYTE	47,47
.BYTE	50,50
.BYTE	51,51
.BYTE	52,52
.BYTE	53,53
.BYTE	54,54
.BYTE	55,55
.BYTE	56,56
.BYTE	57,57
.BYTE	60,60
.BYTE	61,61
.BYTE	62,62
.BYTE	63,63

0459	053326	064	064	.BYTE	64,64
0460	053330	065	065	.BYTE	65,65
0461	053332	066	066	.BYTE	66,66
0462	053334	067	067	.BYTE	67,67
0463	053336	070	070	.BYTE	70,70
0464	053340	071	071	.BYTE	71,71
0465	053342	072	072	.BYTE	72,72
0466	053344	073	073	.BYTE	73,73
0467	053346	074	074	.BYTE	74,74
0468	053350	075	075	.BYTE	75,75
0469	053352	076	076	.BYTE	76,76
0470	053354	077	077	.BYTE	77,77
0471	053356	100	100	.BYTE	100,100
0472	053360	101	101	.BYTE	101,101
0473	053362	102	102	.BYTE	102,102
0474	053364	103	103	.BYTE	103,103
0475	053366	000000		HEAD1: .WORD	000000
0476	053370	140000		.WORD	140000
0477	053372	140000		.WORD	140000
0478	053374	000000		HEAD2: .WORD	000000
0479	053376	040000		.WORD	040000
0480	053400	040000		.WORD	040000
0481	053402	000000		HEAD3: .WORD	000000
0482	053404	100000		.WORD	100000
0483	053406	100000		.WORD	100000
0484	053410	000300		HEAD4: .WORD	000300
0485	053412	040057		.WORD	040057
0486	053414	040356		.WORD	040356
0487	053416	000100		HEAD5: .WORD	000100
0488	053420	040000		.WORD	040000
0489	053422	040100		.WORD	040100
0490	053424	000100		.WORD	000100
0491	053426	140001		.WORD	140001
0492	053430	140101		.WORD	140101
0493	053432	000200		HEAD6: .WORD	000200
0494	053434	140000		.WORD	140000
0495	053436	140000		.WORD	140000
0496	053440	000200		.WORD	000200
0497	053442	140001		.WORD	140001
0498	053444	140201		.WORD	140201
0499	053446	000400		HEAD7: .WORD	000400
0500	053450	140000		.WORD	140000
0501	053452	140000		.WORD	140000
0502	053454	000400		.WORD	000400
0503	053456	040001		.WORD	040001
0504	053460	040401		.WORD	040401
0505	053462	000140		HEAD8: .WORD	000140
0506	053464	040000		.WORD	040000
0507	053466	040140		.WORD	040140
0508	053470	000140		.WORD	000140
0509	053472	140001		.WORD	140001
0510	053474	140101		.WORD	140101
0511	053476	000300		HEAD10: .WORD	000300
0512	053500	140000		.WORD	140000
0513	053502	140000		.WORD	140000
0514	053504	000000		HEAD11: .WORD	000000

8515	053506	141000			.WORD	141000
8516	053510	141000			.WORD	141000
8517	053512	000000	000000	000000	ECC1: .WORD	0.0.0.0.0.0.0.0
8518	053520	000000	000000	000000		
8519	053526	000000	000000			
8520	053532	005001			ECC2: .WORD	005001
8521	053534	040040			.WORD	040040
8522	053536	020004			.WORD	020004
8523	053540	000064			.WORD	000064
8524	053542	000000			.WORD	000000
8525	053544	000000			.WORD	000000
8526	053546	000000			.WORD	000000
8527	053550	000000			.WORD	000000
8528	053552	177777			ECC3: .WORD	177777
8529	053554	177777			.WORD	177777
8530	053556	177777			.WORD	177777
8531	053560	177777			.WORD	177777
8532	053562	177777			.WORD	177777
8533	053564	177777			.WORD	177777
8534	053566	177777			.WORD	177777
8535	053570	177777			.WORD	177777
8536	053572	177777			.WORD	177777
8537	053574	177777			.WORD	177777
8538	053576	001252			OPI1: .WORD	001252
8539	053600	141123			.WORD	141123
8540	053602	140371			.WORD	140371
8541	053604	001251			.WORD	001251
8542	053606	141123			.WORD	141123
8543	053610	140372			.WORD	140372
8544	053612	001257			.WORD	001257
8545	053614	141123			.WORD	141123
8546	053616	140374			.WORD	140374
8547	053620	001243			.WORD	001243
8548	053622	141123			.WORD	141123
8549	053624	140360			.WORD	140360
8550	053626	001273			.WORD	001273
8551	053630	141123			.WORD	141123
8552	053632	140350			.WORD	140350
8553	053634	001213			.WORD	001213
8554	053636	141123			.WORD	141123
8555	053640	140330			.WORD	140330
8556	053642	001353			.WORD	001353
8557	053644	141123			.WORD	141123
8558	053646	140270			.WORD	140270
8559	053650	001053			.WORD	001053
8560	053652	141123			.WORD	141123
8561	053654	140170			.WORD	140170
8562	053656	001653			.WORD	001653
8563	053660	141123			.WORD	141123
8564	053662	140770			.WORD	140770
8565	053664	000253			.WORD	000253
8566	053666	141123			.WORD	141123
8567	053670	141370			.WORD	141370
8568	053672	003253			.WORD	003253
8569	053674	141123			.WORD	141123
8570	053676	142370			.WORD	142370

8571	053700	005253	.WORD	005253
8572	053702	141123	.WORD	141123
8573	053704	144370	.WORD	144370
8574	053706	011253	.WORD	011253
8575	053710	141123	.WORD	141123
8576	053712	150370	.WORD	150370
8577	053714	021253	.WORD	021253
8578	053716	141123	.WORD	141123
8579	053720	160370	.WORD	160370
8580	053722	041253	.WORD	041253
8581	053724	141123	.WORD	141123
8582	053726	100370	.WORD	100370
8583	053730	101253	.WORD	101253
8584	053732	141123	.WORD	141123
8585	053734	040370	.WORD	040370
8586	053736	001253	.WORD	001253
8587	053740	141122	.WORD	141122
8588	053742	140371	.WORD	140371
8589	053744	001253	.WORD	001253
8590	053746	141121	.WORD	141121
8591	053750	140372	.WORD	140372
8592	053752	001253	.WORD	001253
8593	053754	141127	.WORD	141127
8594	053756	140374	.WORD	140374
8595	053760	001253	.WORD	001253
8596	053762	141133	.WORD	141133
8597	053764	140360	.WORD	140360
8598	053766	001253	.WORD	001253
8599	053770	141103	.WORD	141103
8600	053772	140350	.WORD	140350
8601	053774	001253	.WORD	001253
8602	053776	141163	.WORD	141163
8603	054000	140330	.WORD	140330
8604	054002	001253	.WORD	001253
8605	054004	141023	.WORD	141023
8606	054006	140270	.WORD	140270
8607	054010	001253	.WORD	001253
8608	054012	141323	.WORD	141323
8609	054014	140170	.WORD	140170
8610	054016	001253	.WORD	001253
8611	054020	141523	.WORD	141523
8612	054022	140770	.WORD	140770
8613	054024	001253	.WORD	001253
8614	054026	140123	.WORD	140123
8615	054030	141370	.WORD	141370
8616	054032	001253	.WORD	001253
8617	054034	143123	.WORD	143123
8618	054036	142370	.WORD	142370
8619	054040	001253	.WORD	001253
8620	054042	145123	.WORD	145123
8621	054044	144370	.WORD	144370
8622	054046	001253	.WORD	001253
8623	054050	151123	.WORD	151123
8624	054052	150370	.WORD	150370
8625	054054	001253	.WORD	001253
8626	054056	161123	.WORD	161123

8627	054060	160370	.WORD	160370
8628	054062	001250	.WORD	001250
8629	054064	141123	.WORD	141123
8630	054066	140373	.WORD	140373
8631	054070	141253	.WORD	141253
8632	054072	141123	.WORD	141123
8633	054074	000370	.WORD	000370
8634	054076	000240	.WORD	000240
8635	054100	140000	.WORD	140000
8636	054102	140240	.WORD	140240
8637	054104	000240	.WORD	000240
8638	054106	140000	.WORD	140000
8639	054110	140240	.WORD	140240
8640	054112	000240	.WORD	000240
8641	054114	140000	.WORD	140000
8642	054116	140240	.WORD	140240
8643	054120	000240	.WORD	000240
8644	054122	140000	.WORD	140000
8645	054124	140240	.WORD	140240
8646	054126	000240	.WORD	000240
8647	054130	140000	.WORD	140000
8648	054132	140240	.WORD	140240
8649	054134	000240	.WORD	000240
8650	054136	140000	.WORD	140000
8651	054140	140240	.WORD	140240
8652	054142	000240	.WORD	000240
8653	054144	140000	.WORD	140000
8654	054146	140240	.WORD	140240
8655	054150	000240	.WORD	000240
8656	054152	140000	.WORD	140000
8657	054154	140240	.WORD	140240
8658	054156	000240	.WORD	000240
8659	054160	140000	.WORD	140000
8660	054162	140240	.WORD	140240
8661	054164	000240	.WORD	000240
8662	054166	140000	.WORD	140000
8663	054170	140240	.WORD	140240
8664	054172	000240	.WORD	000240
8665	054174	140000	.WORD	140000
8666	054176	140240	.WORD	140240
8667	054200	000240	.WORD	000240
8668	054202	140000	.WORD	140000
8669	054204	140240	.WORD	140240
8670	054206	000240	.WORD	000240
8671	054210	140000	.WORD	140000
8672	054212	140240	.WORD	140240
8673	054214	000240	.WORD	000240
8674	054216	140000	.WORD	140000
8675	054220	140240	.WORD	140240
8676	054222	000240	.WORD	000240
8677	054224	140000	.WORD	140000
8678	054226	140240	.WORD	140240
8679	054230	000240	.WORD	000240
8680	054232	140000	.WORD	140000
8681	054234	140240	.WORD	140240
8682	054236	000240	.WORD	000240

CFI2:

8683	054240	140000	.WORD	140000
8684	054242	140240	.WORD	140240
8685	054244	000240	.WORD	000240
8686	054246	140000	.WORD	140000
8687	054250	140240	.WORD	140240
8688	054252	000240	.WORD	000240
8689	054254	140000	.WORD	140000
8690	054256	140240	.WORD	140240
8691	054260	000240	.WORD	000240
8692	054262	140000	.WORD	140000
8693	054264	140240	.WORD	140240
8694	054266	000240	.WORD	000240
8695	054270	140000	.WORD	140000
8696	054272	140240	.WORD	140240
8697	054274	000240	.WORD	000240
8698	054276	140000	.WORD	140000
8699	054300	140240	.WORD	140240
8700	054302	000240	.WORD	000240
8701	054304	140000	.WORD	140000
8702	054306	140240	.WORD	140240
8703	054310	000240	.WORD	000240
8704	054312	140000	.WORD	140000
8705	054314	140240	.WORD	140240
8706	054316	000240	.WORD	000240
8707	054320	140000	.WORD	140000
8708	054322	140240	.WORD	140240
8709	054324	000240	.WORD	000240
8710	054326	140000	.WORD	140000
8711	054330	140240	.WORD	140240
8712	054332	000240	.WORD	000240
8713	054334	140000	.WORD	140000
8714	054336	140240	.WORD	140240
8715	054340	000240	.WORD	000240
8716	054342	140000	.WORD	140000
8717	054344	140240	.WORD	140240
8718	054346	000240	.WORD	000240
8719	054350	140000	.WORD	140000
8720	054352	140240	.WORD	140240
8721	054354	000240	.WORD	000240
8722	054356	140000	.WORD	140000
8723	054360	140240	.WORD	140240
8724	054362	000240	.WORD	000240
8725	054364	140000	.WORD	140000
8726	054366	140040	.WORD	140040
8727	054370	000240	.WORD	000240
8728	054372	140000	.WORD	140000
8729	054374	140040	.WORD	140040
8730	054376	000300	.WORD	000300
8731	054400	140000	.WORD	140000
8732	054402	140300	.WORD	140300
8733	054404	000300	.WORD	000300
8734	054406	140000	.WORD	140000
8735	054410	140300	.WORD	140300
8736	054412	000300	.WORD	000300
8737	054414	140000	.WORD	140000
8738	054416	140300	.WORD	140300

OP13:

8739	054420	000300	.WORD	000300
8740	054422	140000	.WORD	140000
8741	054424	140300	.WORD	140300
8742	054426	000300	.WORD	000300
8743	054430	140000	.WORD	140000
8744	054432	140300	.WORD	140300
8745	054434	000300	.WORD	000300
8746	054436	140000	.WORD	140000
8747	054440	140300	.WORD	140300
8748	054442	000300	.WORD	000300
8749	054444	140000	.WORD	140000
8750	054446	140300	.WORD	140300
8751	054450	000300	.WORD	000300
8752	054452	140000	.WORD	140000
8753	054454	140300	.WORD	140300
8754	054456	000300	.WORD	000300
8755	054460	140000	.WORD	140000
8756	054462	140300	.WORD	140300
8757	054464	000300	.WORD	000300
8758	054466	140000	.WORD	140000
8759	054470	140300	.WORD	140300
8760	054472	000300	.WORD	000300
8761	054474	140000	.WORD	140000
8762	054476	140300	.WORD	140300
8763	054500	000300	.WORD	000300
8764	054502	140000	.WORD	140000
8765	054504	140300	.WORD	140300
8766	054506	000300	.WORD	000300
8767	054510	140000	.WORD	140000
8768	054512	140300	.WORD	140300
8769	054514	000300	.WORD	000300
8770	054516	140000	.WORD	140000
8771	054520	140300	.WORD	140300
8772	054522	000300	.WORD	000300
8773	054524	140000	.WORD	140000
8774	054526	140300	.WORD	140300
8775	054530	000300	.WORD	000300
8776	054532	140000	.WORD	140000
8777	054534	140300	.WORD	140300
8778	054536	000300	.WORD	000300
8779	054540	140000	.WORD	140000
8780	054542	140300	.WORD	140300
8781	054544	000300	.WORD	000300
8782	054546	140000	.WORD	140000
8783	054548	140300	.WORD	140300
8784	054550	000300	.WORD	000300
8785	054552	140000	.WORD	140000
8786	054556	140300	.WORD	140300
8787	054560	000300	.WORD	000300
8788	054562	140000	.WORD	140000
8789	054564	140300	.WORD	140300
8790	054566	000300	.WORD	000300
8791	054570	140000	.WORD	140000
8792	054572	140300	.WORD	140300
8793	054574	000300	.WORD	000300
8794	054576	140000	.WORD	140000

08795	054608	140300	.WORD	140300
08796	054608	000300	.WORD	000300
08797	054604	140000	.WORD	140000
08798	054606	140300	.WORD	140300
08799	054610	000300	.WORD	000300
08800	054612	140000	.WORD	140000
08801	054614	140300	.WORD	140300
08802	054616	000300	.WORD	000300
08803	054620	140000	.WORD	140000
08804	054622	140300	.WORD	140300
08805	054624	000300	.WORD	000300
08806	054628	140000	.WORD	140000
08807	054630	140300	.WORD	140300
08808	054632	000300	.WORD	000300
08809	054634	140000	.WORD	140000
08810	054636	140300	.WORD	140300
08811	054640	000300	.WORD	000300
08812	054642	140000	.WORD	140000
08813	054644	140300	.WORD	140300
08814	054646	000300	.WORD	000300
08815	054650	140000	.WORD	140000
08816	054652	140300	.WORD	140300
08817	054654	000300	.WORD	000300
08818	054656	140000	.WORD	140000
08819	054660	140300	.WORD	140300
08820	054662	000300	.WORD	000300
08821	054664	140000	.WORD	140000
08822	054666	140200	.WORD	140200
08823	054670	000300	.WORD	000300
08824	054672	140000	.WORD	140000
08825	054674	140300	.WORD	140300
08826	054676	000340	.WORD	000340
08827	054700	140000	.WORD	140000
08828	054702	140000	.WORD	140000
08829	054704	000040	.WORD	000040
08830	054706	140000	.WORD	140000
08831	054710	000040	.WORD	000040
08832	054712	000040	.WORD	000040
08833	054714	140000	.WORD	140000
08834	054716	000040	.WORD	000040
08835	054720	000040	.WORD	000040
08836	054722	140000	.WORD	140000
08837	054724	140040	.WORD	140040
08838	054726	000040	.WORD	000040
08839	054730	140000	.WORD	140000
08840	054732	140040	.WORD	140040
08841	054734	000040	.WORD	000040
08842	054736	140000	.WORD	140000
08843	054738	140040	.WORD	140040
08844	054740	000040	.WORD	000040
08845	054742	140000	.WORD	140000
08846	054744	140040	.WORD	140040
08847	054746	000040	.WORD	000040
08848	054748	140000	.WORD	140000
08849	054750	140040	.WORD	140040
08850	054752	000040	.WORD	000040
08851	054754	140000	.WORD	140000
08852	054756	140040	.WORD	140040
08853	054758	000040	.WORD	000040
08854	054760	140000	.WORD	140000
08855	054762	140040	.WORD	140040
08856	054764	000040	.WORD	000040
08857	054766	140000	.WORD	140000
08858	054768	140040	.WORD	140040
08859	054770	000040	.WORD	000040
08860	054772	140000	.WORD	140000
08861	054774	140040	.WORD	140040
08862	054776	000040	.WORD	000040
08863	054778	140000	.WORD	140000
08864	054780	140040	.WORD	140040
08865	054782	000040	.WORD	000040
08866	054784	140000	.WORD	140000
08867	054786	140040	.WORD	140040
08868	054788	000040	.WORD	000040
08869	054790	140000	.WORD	140000
08870	054792	140040	.WORD	140040
08871	054794	000040	.WORD	000040
08872	054796	140000	.WORD	140000
08873	054798	140040	.WORD	140040
08874	054800	000040	.WORD	000040
08875	054802	140000	.WORD	140000
08876	054804	140040	.WORD	140040
08877	054806	000040	.WORD	000040
08878	054808	140000	.WORD	140000
08879	054810	140040	.WORD	140040
08880	054812	000040	.WORD	000040
08881	054814	140000	.WORD	140000
08882	054816	140040	.WORD	140040
08883	054818	000040	.WORD	000040
08884	054820	140000	.WORD	140000
08885	054822	140040	.WORD	140040
08886	054824	000040	.WORD	000040
08887	054826	140000	.WORD	140000
08888	054828	140040	.WORD	140040
08889	054830	000040	.WORD	000040
08890	054832	140000	.WORD	140000
08891	054834	140040	.WORD	140040
08892	054836	000040	.WORD	000040
08893	054838	140000	.WORD	140000
08894	054840	140040	.WORD	140040
08895	054842	000040	.WORD	000040
08896	054844	140000	.WORD	140000
08897	054846	140040	.WORD	140040
08898	054848	000040	.WORD	000040
08899	054850	140000	.WORD	140000
08900	054852	140040	.WORD	140040
08901	054854	000040	.WORD	000040
08902	054856	140000	.WORD	140000
08903	054858	140040	.WORD	140040
08904	054860	000040	.WORD	000040
08905	054862	140000	.WORD	140000
08906	054864	140040	.WORD	140040
08907	054866	000040	.WORD	000040
08908	054868	140000	.WORD	140000
08909	054870	140040	.WORD	140040
08910	054872	000040	.WORD	000040
08911	054874	140000	.WORD	140000
08912	054876	140040	.WORD	140040
08913	054878	000040	.WORD	000040
08914	054880	140000	.WORD	140000
08915	054882	140040	.WORD	140040
08916	054884	000040	.WORD	000040
08917	054886	140000	.WORD	140000
08918	054888	140040	.WORD	140040
08919	054890	000040	.WORD	000040
08920	054892	140000	.WORD	140000
08921	054894	140040	.WORD	140040
08922	054896	000040	.WORD	000040
08923	054898	140000	.WORD	140000
08924	054900	140040	.WORD	140040
08925	054902	000040	.WORD	000040
08926	054904	140000	.WORD	140000
08927	054906	140040	.WORD	140040
08928	054908	000040	.WORD	000040
08929	054910	140000	.WORD	140000
08930	054912	140040	.WORD	140040
08931	054914	000040	.WORD	000040
08932	054916	140000	.WORD	140000
08933	054918	140040	.WORD	140040
08934	054920	000040	.WORD	000040
08935	054922	140000	.WORD	140000
08936	054924	140040	.WORD	140040
08937	054926	000040	.WORD	000040
08938	054928	140000	.WORD	140000
08939	054930	140040	.WORD	140040
08940	054932	000040	.WORD	000040
08941	054934	140000	.WORD	140000
08942	054936	140040	.WORD	140040
08943	054938	000040	.WORD	000040
08944	054940	140000	.WORD	140000
08945	054942	140040	.WORD	140040
08946	054944	000040	.WORD	000040
08947	054946	140000	.WORD	140000
08948	054948	140040	.WORD	140040
08949	054950	000040	.WORD	000040
08950	054952	140000	.WORD	140000
08951	054954	140040	.WORD	140040
08952	054956	000040	.WORD	000040
08953	054958	140000	.WORD	140000
08954	054960	140040	.WORD	140040
08955	054962	000040	.WORD	000040
08956	054964	140000	.WORD	140000
08957	054966	140040	.WORD	140040
08958	054968	000040	.WORD	000040
08959	054970	140000	.WORD	140000
08960	054972	140040	.WORD	140040
08961	054974	000040	.WORD	000040
08962	054976	140000	.WORD	140000
08963	054978	140040	.WORD	140040
08964	054980	000040	.WORD	000040
08965	054982	140000	.WORD	140000
08966	054984	140040	.WORD	140040
08967	054986	000040	.WORD	000040
08968	054988	140000	.WORD	140000
08969	054990	140040	.WORD	140040
08970	054992	000040	.WORD	000040
08971	054994	140000	.WORD	140000
08972	054996	140040	.WORD	140040
08973	054998	000040	.WORD	000040
08974	055000	140000	.WORD	140000
08975	055002	140040	.WORD	140040
08976	055004	000040	.WORD	000040
08977	055006	140000	.WORD	140000
08978	055008	140040	.WORD	140040
08979	055010	000040	.WORD	000040
08980	055012	140000	.WORD	140000
08981	055014	140040	.WORD	140040
08982	055016	000040	.WORD	000040
08983	055018	140000	.WORD	140000
08984	055020	140040	.WORD	140040
08985	055022	000040	.WORD	000040
08986	055024	140000	.WORD	140000
08987	055026	140040	.WORD	140040
08988	055028	000040	.WORD	000040
08989	055030	140000	.WORD	140000
08990	055032	140040	.WORD	140040
08991	055034	000040	.WORD	000040
08992	055036	140000	.WORD	140000
08993	055038	140040	.WORD	140040
08994	055040	000040	.WORD	000040
08995	055042	140000	.WORD	140000
08996	055044	140040	.WORD	140040
08997	055046	000040	.WORD	000040
08998	055048	140000	.WORD	140000
08999	055050	140040	.WORD	140040

CP11:

8907	055140	000040				.WORD	000040
8908	055142	140000				.WORD	140000
8909	055144	140040				.WORD	140040
8910	055146	000040				.WORD	000040
8911	055150	140000				.WORD	140000
8912	055152	140040				.WORD	140040
8913	055154	000040				.WORD	000040
8914	055156	140000				.WORD	140000
8915	055160	140040				.WORD	140040
8916	055162	000040				.WORD	000040
8917	055164	140000				.WORD	140000
8918	055166	140040				.WORD	140040
8919	055170	000040				.WORD	000040
8920	055172	140000				.WORD	140000
8921	055174	140040				.WORD	140040
8922	055176	001253			VRCHDR:	.WORD	001253
8923	055200	141123				.WORD	141123
8924	055202	140370				.WORD	140370
8925	055204	177777	177777	177777	MOD2BF:	.WORD	177777,177777,177777,177777,177777,177777
8926	055212	177777	177777	177777			
8927	055220	000400			BUFF:	.BLKW	400
8928	056220	125252	125252	125252	ECCBJF:	.WORD	125252,125252,125252,125252,125252,125252
8929	056226	125252	125252	125252			
8930	056234	125252	125252	177777		.WORD	125252,125252,177777,177777,177777,177777
8931	056242	177777	177777	177777			
8932	056250	177777	177777	177777		.WORD	177777,177777,177777,177777,000000,000000
8933	056256	177777	000000	000000			
8934	056264	000000	000000	000000		.WORD	000000,000000,000000,000000,000000,000000
8935	056272	000000	000000	000000			
8936	056300	052525	052525	052525		.WORD	052525,052525,052525,052525,052525,052525
8937	056306	025252	052525	052525			
8938	056314	052525	052525	121105		.WORD	052525,052525,121105,150442,064221,132110
8939	056322	150442	064221	132110			
8940	056330	055044	026422	013211		.WORD	055044,026422,013211,105504,042642,021321
8941	056336	105504	042642	021321			
8942	056344	110550	044264	022132		.WORD	110550,044264,022132,011055,104426,042213
8943	056352	011055	104426	042213			
8944	056360	133333	066666	155555		.WORD	133333,066666,155555,155555,133333,066666
8945	056366	155555	133333	066666			
8946	056374	066666	155555	155555		.WORD	066666,155555,155555,133333,133333,133333
8947	056402	133333	133333	133333			
8948	056410	133333	133333	133333		.WORD	133333,133333,133333,133333,177777,177777
8949	056416	133333	177777	177777			
8950	056424	177777	052525	052525		.WORD	177777,052525,052525,052525,177777,177777
8951	056432	052525	177777	177777			
8952	056440	052525	052525	177777		.WORD	052525,052525,177777,052525,177252,177252
8953	056446	052525	177252	177252			
8954	056454	172765	172765	072307		.WORD	172765,172765,072307,135143,156461,167230
8955	056462	135143	156461	167230			
8956	056470	073514	035646	016723		.WORD	073514,035646,016723,107351,143564,061672
8957	056476	107351	143564	061672			
8958	056504	030735	114356	046167		.WORD	030735,114356,046167,123073,151453,164616
8959	056512	123073	151453	164616			
8960	056520	125252	125252	125252		.WORD	125252,125252,125252,125252,125252,125252
8961	056526	125252	125252	125252			
8962	056534	125252	125252	177777		.WORD	125252,125252,177777,177777,177777,177777

N13

8963	056542	177777	177777	177777	
8964	056550	177777	177777	177777	.WORD 177777,177777,177777,177777,000000,000000
8965	056556	177777	000000	000000	
8966	056564	000000	000000	000000	.WORD: 000000,000000,000000,000000,000000,000000
8967	056572	000000	000000	000000	
8968	056600	052525	052525	052525	.WORD 052525,052525,052525,052525,052525,052525
8969	056606	025252	052525	052525	
8970	056614	052525	052525	121105	.WORD 052525,052525,121105,150442,064221,132110
8971	056622	150442	064221	132110	
8972	056630	055044	026422	013211	.WORD 055044,026422,013211,105504,042642,021321
8973	056636	105504	042642	021321	
8974	056644	110550	044264	022132	.WORD 110550,044264,022132,011055,104426,042213
8975	056652	011055	104426	042213	
8976	056660	133333	066666	155555	.WORD 133333,066666,155555,155555,133333,066666
8977	056666	155555	133333	066666	
8979	056674	066666	155555	155555	.WORD 066666,155555,155555,133333,133333,133333
8979	056702	133333	133333	133333	
8980	056710	133333	133333	133333	.WORD 133333,133333,133333,133333,177777,177777
8981	056716	133333	177777	177777	
8982	056724	177777	052525	052525	.WORD 177777,052525,052525,052525,177777,177777
8983	056732	052525	177777	177777	
8984	056740	052525	052525	177777	.WORD 052525,052525,177777,052525,177252,177252
8985	056746	052525	177252	177252	
8986	056754	172765	172765	072307	.WORD 172765,172765,072307,135143,156461,167230
8987	056762	135143	156461	167230	
8988	056770	073514	035646	016723	.WORD 073514,035646,016723,107351,143564,061672
8989	056776	107351	143564	061672	
8990	057004	030735	114356	046167	.WORD 030735,114356,046167,123073,151453,164616
8991	057012	123073	151453	164616	
8992	057020	125252	125252	125252	.WORD 125252,125252,125252,125252,125252,125252
8993	057026	125252	125252	125252	
8994	057034	125252	125252	177777	.WORD 125252,125252,177777,177777,177777,177777
8995	057042	177777	177777	177777	
8996	057050	177777	177777	177777	.WORD 177777,177777,177777,177777,000000,000000
8997	057056	177777	000000	000000	
8998	057064	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000
8999	057072	000000	000000	000000	
9000	057100	052525	052525	052525	.WORD 052525,052525,052525,052525,052525,052525
9001	057106	025252	052525	052525	
9002	057114	052525	052525	121105	.WORD 052525,052525,121105,150442,064221,132110
9003	057122	150442	064221	132110	
9004	057130	055044	026422	013211	.WORD 055044,026422,013211,105504,042642,021321
9005	057136	105504	042642	021321	
9006	057144	110550	044264	022132	.WORD 110550,044264,022132,011055,104426,042213
9007	057152	011055	104426	042213	
9008	057160	177777	177777	177777	.WORD 177777,177777,177777,177777,177777,177777
9009	057166	177777	177777	177777	
9010	057174	177777	177777	177777	.WORD 177777,177777,177777,177777,177777,177777
9011	057202	177777	177777	177777	
9012	057210	177777	177777	177777	.WORD 177777,177777,177777,177777
9013	057216	177777			
9014	057220	177777	177777	177777	BADPAR: .WORD 177777,177777,177777,177777,177777,177777
9015	057226	177777	177777	177777	
9016		000001			.END

ABASE = 177440	1021#	1330	1371
ABORT = 043756	6504	7731#	
ACDM1 = 000000	1330	1373	
ACDM2 = 000000	1330	1374	
ACLO = 000010	1116#		
ACPLUP = 000000	1330	1345	
ROOM0 = 000000	1330		
ROOM1 = 000000	1330		
ROOM10 = 000000	1330		
ROOM11 = 000000	1330		
ROOM12 = 000000	1330		
ROOM13 = 000000	1330		
ROOM14 = 000000	1330		
ROOM15 = 000000	1330		
ROOM2 = 000000	1330		
ROOM3 = 000000	1330		
ROOM4 = 000000	1330		
ROOM5 = 000000	1330		
ROOM6 = 000000	1330		
ROOM7 = 000000	1330		
ROOM8 = 000000	1330		
ROOM9 = 000000	1330		
ROEVCT = 000000	1330	1336	
ROEVN = 000000	1330	1372	
ROENV = 000000	1330	1341	
ROENVH = 000000	1330	1342	
RFATAL = 000000	1330	1333	
RMAOR1 = 000000	1330	1358	
RMAOR2 = 000000	1330	1362	
RMAOR3 = 000000	1330	1365	
RMAOR4 = 000000	1330	1368	
RMAHS1 = 000000	1330	1352	
RMAHS2 = 000000	1330	1360	
RMAHS3 = 000000	1330	1363	
RMAHS4 = 000000	1330	1366	
RMSGAD = 000000	1330	1338	
RMSGLG = 000000	1330	1339	
RMSGTY = 000000	1330	1332	
RMTYP1 = 000000	1330	1353	
RMTYP2 = 000000	1330	1361	
RMTYP3 = 000000	1330	1364	
RMTYP4 = 000000	1330	1367	
RPASS = 000000	1330	1335	
RPRIOR = 000240	1020#	1330	
RPTCSU = 000040	6777#	6863	
RPTENV = 000001	6733	6775#	6809 6856
RPTSIZ = 000200	2223	6774#	
RPTSPO = 000100	6735	6776#	6858
RESWREG = 000000	1330	1343	
RTESTN = 000000	1330	1334	
RUNIT = 000000	1330	1337	
RUSWR = 000000	1330	1344	
RYECT1 = 000210	1019#	1330	1369
RYECT2 = 000000	1330	1370	
RZCPAR = 057220	5774	5788	5900 9014#
ERT = 000020	1079#		

DH077A	045553	7660	7894#															
DH077B	045602	7662	7898#															
DH134A	045626	7670	7902#															
DH134B	045652	7672	7906#															
DH144A	045700	7684	7910#															
DH144B	045777	7686	7921#															
DH151A	046074	7696	7932#															
DI	= 040000	1071#																
DISPLA	001142	1303#	2211#	2219*	6379*	6796*												
DISPRE	000174	1161#	1232#	2219														
DLT	= 100000	1090#																
DPC	= 000040	1132#																
		2509	2324	2333	2334	2370	2379	2380	2417	2426	2427	2463	2472	2473	2473	2473	2473	2473
		2732	2518	2519	2556	2565	2566	2614	2621	2622	2671	2678	2673	2673	2673	2673	2673	2673
		2980	2733	2780	2787	2788	2830	2835	2836	2923	2929	2930	2933	2933	2933	2933	2933	2933
		3104	3014	3020	3021	3025	3026	3027	3030	3062	3091	3099	3100	3100	3103	3103	3103	3103
		3341	3172	3180	3181	3184	3185	3253	3261	3262	3265	3266	3332	3332	3340	3340	3340	3340
		3515	3344	3345	3414	3422	3423	3426	3427	3495	3503	3504	3512	3512	3514	3514	3514	3514
		3795	3609	3623	3624	3627	3628	3696	3704	3705	3708	3709	3786	3786	3794	3794	3794	3794
		4001	3903	3805	3806	3854	3889	3897	3898	3906	3908	3909	3957	3957	3993	3993	3993	3993
		4211	4002	4010	4012	4013	4097	4105	4106	4114	4116	4117	4202	4202	4210	4210	4210	4210
		4464	4219	4221	4222	4328	4336	4337	4345	4347	4348	4447	4455	4455	4456	4456	4456	4456
		4653	4466	4467	4551	4559	4560	4563	4564	4596	4597	4641	4649	4649	4650	4650	4650	4650
		4930	4654	4686	4687	4731	4739	4740	4743	4744	4776	4777	4821	4821	4829	4829	4829	4829
		5033	4833	4834	4866	4867	4914	4920	4921	4924	4925	4963	5026	5026	5032	5032	5032	5032
		5257	5033	5037	5075	5140	5146	5147	5150	5151	5189	5247	5253	5253	5254	5254	5254	5254
		5638	5257	5296	5379	5380	5440	5446	5447	5450	5451	5489	5576	5576	5577	5577	5577	5577
		5933	5638	5644	5649	5650	5692	5787	5793	5794	5798	5799	5841	5841	5927	5927	5927	5927
		6681	5933	5938	5939	5981	6609	6626	6627	6640	6641	6667	6668	6668	6680	6680	6680	6680
		1113#	6707	6710	6711	6712	6713	6716	6717	6718	6719							
DRA	= 000001	1120#																
DRDY	= 000200	1118#																
DRDT	= 000040	1097#																
DRPAR	= 000010	1077#																
DRVMSK	= 000007	1124#																
DSC	= 040000	921#	1302	2210														
DSMR	= 177570	1106#																
DTE	= 010000	1099#																
DTYPE	= 000040	1395	7491#															
DT000	042460	1400	7492#															
DT002	042464	1406	1412	1418	1424	1430	1436	1442	1448	1454	1460	1466	1472	1472	1478	1478	1478	1478
DT003	042506	1484	1490	1496	1502	1508	1515	1522	1529	1536	1543	1550	1557	1557	1564	1564	1564	1564
		1571	1578	1585	1592	7495#												
DT041	042526	1598	1604	1610	1616	1622	7498#											
DT046	042556	1628	1634	1640	1707	1713	1719	1726	1733	1740	1968	1974	1980	1980	1986	1986	1986	1986
		1992	1998	7502#														
DT051	042576	1646	7505#															
DT052	042610	1652	2046	7507#														
DT053	042624	1658	1701	7509#														
DT054	042634	1664	1670	1676	1682	1688	1694	7511#										
DT071	042650	1746	1752	1758	7513#													
DT074	042676	1764	1770	1776	1788	1794	1800	1812	1818	1824	1836	1842	1848	1848	1860	1860	1860	1860
		1866	1872	1884	1890	1896	1908	1914	1920	1932	1938	1944	7517#	7517#				
DT077	042720	1782	1806	1830	1854	1878	1902	1926	1950	7520#								
DT134	042732	1956	1962	2100	7522#													
DT144	042744	2004	2010	2016	2022	2028	2034	2040	2052	2058	2064	2070	2076	2076	2082	2082	2082	2082

EM334	051347	5366	5563	6090	8239*												
EM335	051405	2002	2008	2014	2020	2026	2032	2038	8245*								
EM336	051461	2044	8253*														
EM337	051536	2050	2056	2062	2068	2074	2080	2086	8261*								
EM338	051615	5707	5856	5996	6029	8269*											
EM339	051703	5733	5882	6022	6060	8279*											
EM340	052007	2092	2098	8291*													
EM400	052054	1405	1423	1441	1459	1477	1495	1514	1535	1556	1577	1597	1651	1706			
		1725	1745	1763	1787	1811	1835	1859	1883	1907	1931	1967	1985	2003			
		2045	2051	8298*													
EM4001	052072	1411	1429	1447	1465	1483	1501	8301*									
EM4002	052113	1417	1435	1453	1471	1489	1507	8304*									
EM4003	052134	1521	1542	1563	1584	8307*											
EM4004	052164	1528	1549	1570	1591	8311*											
EM4005	052214	1603	1712	1732	1751	1769	1793	1817	1841	1865	1889	1913	1937	1973			
		1991	2009	2057	8315*												
EM4006	052232	1609	1718	1739	1757	1775	1799	1823	1847	1871	1895	1919	1943	1979			
		1997	2015	2063	8318*												
EM4007	052256	1615	2021	2069	8322*												
EM4008	052304	1621	2027	2075	8326*												
EM4009	052331	1627	8330*														
EM4010	052401	1633	8337*														
EM4011	052451	1639	8344*														
EM4012	052527	1645	8352*														
EM4013	052567	1657	1781	1805	1829	1853	1877	1901	1925	1949	8358*						
EM4014	052635	1663	1675	1687	2039	2087	8365*										
EM4015	052671	1669	1681	1693	2033	2081	8370*										
EM4016	052731	1700	8376*														
EM4017	052747	1955	1961	2093	2099	8379*											
ERRCNT	003250	2149*	2227*	6499*	6502												
ERRVEC=	000004	1003*	2208	2209*	2220*	6248*	6249*	6268*	6269*	6330	6331*	6333*	6336*				
E.ASOF	003216	2131*															
E.BA	003204	2126*	2847*	2858	5395*	5406	5592*	5603	7498	7527							
E.CS1	003200	2124*	2340*	2343	2386*	2389	2433*	2436	2479*	2482	2525*	2528	2572*	2575			
		2607*	2619	2628	2664*	2676	2685	2718*	2730	2739	2773*	2785	2794	2807			
		2809*	2844*	2849	2873	2935*	2942	2950	2966	3379*	3382	3461*	3464	3509*			
		3550	3567*	3568*	3576*	3577	3597*	3662	3743*	3746	3800*	3840	3903*	3943			
		4007*	4047	4111*	4151	4216*	4257	4274*	4275*	4293*	4284	4342*	4383	4400*			
		4401*	4409*	4410	4461*	4502	4519*	4520*	4528*	4529	4603*	4606	4619*	4620			
		4693*	4696	4709*	4710	4783*	4786	4799*	4800	4873*	4876	4889*	4890	5386*			
		5397	5583*	5594	5562	7495	7498	7502	7507	7513	7517	7524					
E.CS2	003210	2128*	2845*	2855	2870*	2885	2897*	3380*	3385	3462*	3467	3510*	3553	3598*			
		3665	3744*	3749	3801*	3843	3904*	3946	4008*	4050	4112*	4154	4217*	4260			
		4343*	4386	4462*	4505	4604*	4609	4694*	4699	4784*	4789	4874*	4879	5387*			
		5400	5584*	5597	7498	7502	7513	7517	7524								
E.DA	003206	2127*	2606*	2618	2663*	2675	2698*	2699	2717*	2729	2752*	2753	2772*	2784			
		3137	3219	3299	5394*	5415	5591*	5612	6530*	6531*	6532	6534*	6535*	6536			
		6538*	6553	6560	7511	7527											
E.DB	003222	2133*	2871*	2891	7505												
E.DCYL	003220	2132*	2605*	2617	2641*	2642	2644*	2645	2662*	2674	2716*	2728	2771*	2783			
		3140	3221	3302	5393*	5412	5590*	5609	6529*	6539*	6545	7511	7527				
E.DS	003212	2129*															
E.ECFC	003232	2137*															
E.ECPT	003234	2138*	4974*	4991	5105	5219	5326	5519	5723	5872	6012	6046	6102*	6104			
		6603*	6604*	522	7530												
E.EF	003214	2130*	2848*	2864	2888	3381*	3388	3463*	3470	3511*	3556	3566*	3599*	3668			

MESMSK = 000017	1129#													
MEMD = 002000	1137#	1208#	2980	3062	3512	3803	3854	3906	3957	4010	4114	4219	4345	
	4464	4963	5075	5189	5296	5485	5692	5841	5981	6609	6615	6639	6679	
MIND = 000200	1134#	1205#	3025	3026	3027									
MOD28F 055204	5639	5704	5853	8925#										
MSP = 000100	1133#	1204#	2933	3103	3184	3265	3344	3426	3514	3627	3708	3805	3908	
	4012	4116	4221	4347	4466	4563	4653	4743	4833	4924	5036	5150	5257	
	5450	5649	5798	5938										
M1.BIT 003256	2152#	2937*	2957*	2974*	3032*	3043*	3056*	3106*	3117*	3130*	3187*	3198*	3211*	
	3268*	3279*	3292*	3347*	3358*	3371*	3429*	3440*	3453*	3519*	3529*	3542*	3630*	
	3641*	3654*	3711*	3722*	3735*	3808*	3818*	3832*	3911*	3921*	3935*	4015*	4025*	
	4039*	4119*	4129*	4143*	4226*	4236*	4249*	4352*	4362*	4375*	4471*	4481*	4494*	
	4566*	4577*	4590*	4656*	4667*	4680*	4746*	4757*	4770*	4836*	4847*	4860*	4927*	
	4938*	4951*	4967*	4979	4980*	5039*	5050*	5063*	5072*	5092	5093*	5153*	5164*	
	5177*	5196*	5206	5207*	5260*	5271*	5284*	5291*	5313	5314*	5345	5346*	5369	
	5370*	5453*	5464*	5477*	5484*	5506	5507*	5542	5543*	5566	5567*	5653*	5667*	
	5680*	5689*	5709	5710*	5802*	5816*	5829*	5838*	5858	5859*	5942*	5956*	5969*	
	5978*	5998	5999*	6032	6033*	6069	6070*	6093	6094*	6613	6659	6664	6705	
	7492													
M2.BIT 003260	2153#	4961*	4979*	5073*	5092*	5187*	5206*	5292*	5313*	5345*	5369*	5485*	5506*	
	5542*	5566*	5690*	5709*	5839*	5858*	5979*	5998*	6032*	6069*	6093*	6618	6622	
	7492													
NED = 010000	1087#													
NEM = 004000	1086#													
NEWPAS 004332	2304#													
NPRBUF 053156	2832	2847	2867	8406#										
NXF = 000004	1096#													
OFFSET = 000015	1050#													
OFST = 000004	1115#													
OPI = 020000	1107#	3566	4273	4399	4518									
OPI1 053576	3508	8538#												
OPI2 054076	4215	8634#												
OPI3 054376	4341	8730#												
OPI4 054676	4460	8826#												
OPR001 043524	2254	7700#												
OPR002 043553	2257	2268	2284	7704#										
OPR003 043561	2264	7706#												
OPR004 043611	2275	7711#												
OPR005 043633	6297	7715#												
OPR006 043673	2251	7721#												
OR = 000200	1082#	2845	2870	3380	3462	3510	3598	3744	3801	3904	4008	4112	4217	
	4343	4462	4604	4694										
	1045#													
PACK = 000003	2304	6248#	7435											
PARCHK 034402	1167	1238	2168#											
PARM 003314	1017#	6195	6221	6254										
PAR.EN = 000001	1131#													
PAT = 000020	1138#	1209#	6610	6625	6661									
PCA = 004000	1139#	1210#	6610	6620	6666									
PCD = 010000	1085#													
PGE = 002000	1123#													
PIP = 020000	920#													
PIRG = 177772	1014#													
PIRGVE = 000240	2151#	2936*	2947*	2957	2960*	2963*	2974	2975*	3031*	3037*	3043	3046*	3049*	
PR.BIT 003254	3056	3057*	3105*	3111*	3117	3120*	3123*	3130	3131*	3186*	3192*	3198	320*	
	3204*	3211	3212*	3267*	3273*	3279	3282*	3285*	3292	3293*	3346*	3352*	3358	

		3361*	3364*	3371	3372*	3428*	3434*	3440	3443*	3446*	3453	3454*	3518*	3524*
		3529	3532*	3535*	3542	3543*	3629*	3635*	3641	3644*	3647*	3654	3655*	3710*
		3716*	3722	3725*	3728*	3735	3736*	3807*	3813*	3818	3821*	3824*	3832	3833*
		3910*	3916*	3921	3924*	3927*	3935	3936*	4014*	4020*	4025	4028*	4031*	4039
		4040*	4118*	4124*	4129	4132*	4135*	4143	417*	4225*	4231*	4236	4239*	4242*
		4249	4250*	4351*	4357*	4362	4365*	4368*	4375	4376*	4470*	4476*	4481	4484*
		4487*	4494	4495*	4565*	4571*	4577	4580*	4583*	4590	4591*	4655*	4661*	4667
		4670*	4673*	4680	4681*	4745*	4751*	4757	4760*	4763*	4770	4771*	4835*	4841*
		4847	4850*	4853*	4860	4861*	4926*	4932*	4938	4941*	4944*	4951	4952*	4959*
		4980	4981*	5038*	5044*	5050	5053*	5056*	5063	5064*	5071*	5093	5094*	5152*
		5158*	5164	5167*	5170*	5177	5178*	5185*	5207	5208*	5259*	5265*	5271	5274*
		5277*	5284	5285*	5290*	5314	5315*	5346	5347*	5370	5371*	5452*	5458*	5464
		5467*	5470*	5477	5478*	5483*	5507	5508*	5543	5544*	5567	5569*	5652*	5660*
		5667	5670*	5673*	5680	5681*	5688*	5710	5711*	5801*	5809*	5816	5819*	5822*
		5829	5830*	5837*	5859	5860*	5941*	5949*	5956	5959*	5962*	5969	5970*	5977*
		5999	6000*	6033	6034*	6070	6071*	6094	6095*	6611	6703	7492		
		937*	2305	6228										
PRO	=	000000												
PR1	=	000040												
PR2	=	000100												
PR3	=	000140												
PR4	=	000200												
PR5	=	000240												
PR6	=	000300												
PR7	=	000340												
PS	=	177776												
PSW	=	177776												
PWRVEC	=	000024												
P1.BIT	=	003252												
		1009*	2198*	2199*	7417*	7418*	7431*	7432*						
		2150*	4958*	4969*	4981	4984*	4987*	5070*	5081*	5094	5097*	5100*	5184*	5195*
		5208	5211*	5214*	5289*	5302*	5315	5318*	5321*	5347	5350*	5353*	5371	5372*
		5482*	5495*	5508	5511*	5514*	5544	5547*	5550*	5568	5569*	5687*	5698*	5711
		5715*	5718*	5836*	5847*	5860	5864*	5867*	5976*	5987*	6000	6004*	6007*	6034
		6038*	6041*	6071	6074*	6077*	6095	6096*	6579	6616	6657	7492		
ROBIT	037030	2940	2948	2964	2976	3034	3038	3050	3058	3108	3112	3124	3132	3189
		3193	3205	3213	3270	3274	3286	3294	3349	3353	3365	3373	3431	3435
		3447	3455	3521	3525	3536	3544	3632	3636	3648	3656	3713	3717	3729
		3737	3810	3814	3825	3834	3913	3917	3928	3937	4017	4021	4032	4041
		4121	4125	4136	4145	4228	4232	4243	4251	4354	4358	4369	4377	4473
		4477	4488	4496	4568	4572	4584	4592	4658	4662	4674	4682	4748	4752
		4764	4772	4838	4842	4854	4862	4929	4933	4945	4953	5041	5045	5057
		5065	5155	5159	5171	5179	5262	5266	5278	5286	5455	5459	5471	5479
		5656	5661	5674*	5682	5805	5810	5823	5831	5945	5950	5963	5971	6703*
		7257	7483*											
RDCHR	=	104410												
RDDATA	=	000021												
RDGATE	=	100000												
RDHEAD	=	000025												
RDLIN	=	104411												
RDOCT	=	104412												
RDY	=	000200												
		1052*	2330	2340	2469	2479	2607	2664	2718	2773	4736	4783		
		1142*	1213*	6610										
		1054*												
		7331	7484*											
		2258	2269	2285	7485*									
		1063*	3568	3576	4275	4283	4401	4409	4520	4528	4603	4693	4783	4873
		5386	5583											
		1049*												
RECAL	=	000013												
RESREG	=	104414												
RESTR	=	003324												
RESVEC	=	000010												
RKASOF	=	000016												
RKBA	=	000004												
		1027*	2326*	2372*	2419*	2465*	2511*	2558*	2616*	2673*	2727*	2782*	2832*	2842
		2925*	3016*	3095*	3176*	3257*	3336*	3418*	3496*	3617*	3700*	3787*	3890*	3994*

	4098*	4203*	4329*	4448*	4552*	4642*	4732*	4822*	4915*	5027*	5141*	5248*	5391
RKCS1 = 000000	5441*	5588	5639*	5788*	5928*								
	1025*	2323*	2330*	2337	2346*	2365*	2376*	2383	2392*	2416*	2423*	2430	2439*
	2462*	2469*	2476	2485*	2508*	2515*	2522	2531*	2555*	2562*	2569	2578*	2613*
	2619*	2625	2631*	2670*	2676*	2682	2688*	2724*	2730*	2736	2742*	2779*	2785*
	2791	2797*	2829*	2833*	2839	2852*	2876	2882*	2922*	2926*	2941	2949	2965
	3010*	3017*	3090*	3096*	3171*	3177*	3252*	3258*	3331*	3337*	3376	3413*	3419*
	3458	3491*	3500*	3547	3572*	3573	3605*	3621*	3659	3695*	3701*	3740	3785*
	3791*	3837	3898*	3894*	3940	3992*	3998*	4044	4096*	4102*	4148	4198*	4207*
	4254	4279*	4280	4324*	4333*	4380	4405*	4406	4443*	4452*	4499	4524*	4525
	4550*	4556*	4600	4615*	4616	4640*	4646*	4690	4705*	4706	4730*	4736*	4780
	4795*	4796	4820*	4826*	4870	4885*	4886	4913*	4917*	5025*	5029*	5139*	5143*
RKCS2 = 000010	5246*	5250*	5283	5439*	5443*	5580	5633*	5641*	5782*	5790*	5922*	5930*	6296*
	1029*	2329*	2375*	2422*	2468*	2514*	2561*	2840	2877	2986	3377	3459	3548
	3574	3660	3741	3838	3941	4045	4149	4255	4281	4381	4407	4500	4526
	4501	4617	4691	4707	4781	4797	4871	4887	5384	5581			
RKDA = 000006	1028*	2328*	2374*	2421*	2467*	2513*	2550*	2618*	2675*	2729*	2784*	3093*	3135
	3174*	3216	3255*	3297	3334*	3416*	3499*	3619*	3698*	3789*	3892*	3996*	4100*
	4206*	4332*	4451*	4555*	4645*	4735*	4825*	5390	5587				
RKDB = 000024	1034*	2872											
RKDCYL = 000020	1033*	2327*	2373*	2420*	2466*	2512*	2559*	2617*	2674*	2728*	2783*	3092*	3136
	3173*	3217	3254*	3298	3333*	3415*	3498*	3520*	3697*	3790*	3893*	3997*	4101*
	4205*	4331*	4450*	4554*	4644*	4734*	4824*	5389	5586				
RKDS = 000012	1030*												
RKECPS = 000030	1038*												
RKECPT = 000032	1039*	4430	5103	5217	5325	5518	5721	5870	6010	6044	6103		
RKER = 000014	1031*	2873	2878	2987	3378	3460	3549	3575	3661	3742	3839	3942	4046
	4150	4256	4282	4382	4408	4501	4527	4602	4618	4692	4708	4782	4798
	4872	4988	5385	5582									
RKMF1 = 000026	1035*	2324*	2333*	2334*	2370*	2379*	2380*	2417*	2426*	2427*	2463*	2472*	2473*
	2509*	2518*	2519*	2556*	2565*	2566*	2614*	2621*	2622*	2671*	2678*	2679*	2725*
	2732*	2733*	2790*	2787*	2788*	2830*	2835*	2836*	2923*	2929*	2930*	2933*	2934*
	2979	3014*	3020*	3021*	3025*	3026*	3027*	3030*	3061	3091*	3099*	3100*	3103*
	3104*	3172*	3180*	3181*	3184*	3185*	3253*	3261*	3262*	3265*	3266*	3332*	3340*
	3341*	3344*	3345*	3414*	3422*	3423*	3426*	3427*	3495*	3503*	3504*	3514*	3515*
	3559	3609*	3623*	3624*	3627*	3628*	3696*	3704*	3705*	3708*	3709*	3786*	3794*
	3795*	3805*	3806*	3849	3889*	3897*	3898*	3908*	3909*	3952	3993*	4001*	4002*
	4012*	4013*	4056	4097*	4105*	4106*	4116*	4117*	4160	4202*	4210*	4211*	4221*
	4222*	4266	4328*	4336*	4337*	4347*	4348*	4392	4447*	4455*	4456*	4466*	4467*
	4511	4551*	4559*	4560*	4563*	4564*	4596*	4597*	4641*	4649*	4650*	4653*	4654*
	4686*	4687*	4731*	4739*	4740*	4743*	4744*	4776*	4777*	4821*	4829*	4830*	4833*
	4834*	4866*	4867*	4914*	4920*	4921*	4924*	4925*	5026*	5032*	5033*	5036*	5037*
	5140*	5146*	5147*	5150*	5151*	5247*	5253*	5254*	5257*	5258*	5379*	5380*	5440*
	5446*	5447*	5450*	5451*	5576*	5577*	5638*	5644*	5645*	5649*	5650*	5787*	5793*
	5794*	5798*	5799*	5927*	5933*	5934*	5938*	5939*	6626*	6627*	6628	6640*	6641*
	6642	6667*	6668*	6669	6680*	6681*	6682	6707*	6710*	6711*	6712*	6713*	6716*
	6717*	6718*	6719*										
	1036*	2338	2384	2431	2477	2523	2570	2626	2683	2737	2792		
RKMF2 = 000034	1037*	2339	2385	2432	2478	2524	2571	2627	2684	2738	2793		
RKMF3 = 000036	2143*	2301*											
RKPP1 = 003242	1040*												
RKSPAR = 000022	2142*	2299*	2300*										
RKVEC = 003240	1026*	2325*	2371*	2418*	2464*	2510*	2557*	2615*	2672*	2726*	2791*	2831*	2841
RKWC = 000002	2924*	3015*	3094*	3175*	3256*	3335*	3417*	3497*	3618*	3699*	3789*	3891*	3995*
	4099*	4204*	4330*	4449*	4553*	4643*	4733*	4823*	4916*	5028*	5142*	5249*	5392
	5442*	5589	5640*	5789*	5929*								

S. STSP =	000100	1148#	1219#																	
S. UNLD =	002000	1152#	1223#																	
TBITVE =	000014	1005#																		
TKVEC =	000060	1012#	7078*	7079*																
TPVEC =	000064	1013#																		
TRACK =	003277	2161#	3225*	3226	6519															
TRAPP =	003244	2144#	6280*	7491																
TRAPVE =	000034	1011#	2196*	2197*																
TRTVEC =	000014	1006#																		
TSTBY1 =	044043	5766	7740#																	
TSTBY2 =	044053	5769	7742#																	
TST1 =	004350	2306	2320#	6154	6384															
TST10 =	006462	2659#	6391																	
TST11 =	006732	2713#	6392																	
TST12 =	007202	2768#	6393																	
TST13 =	007456	2808	2826#	6394																
TST14 =	010126	2853	2883	2919#	6395															
TST15 =	010552	2982	2984	3007#	6396															
TST16 =	011112	3064	3080#	6397																
TST17 =	011504	3161#	6398																	
TST2 =	004602	2347	2353	2366#	6385															
TST20 =	012100	3242#	6399																	
TST21 =	012474	3328#	6400																	
TST22 =	013070	3389	3410#	6401																
TST23 =	013464	3471	3488#	6402																
TST24 =	014252	3578	3594#	6403																
TST25 =	014742	3673	3692#	6404																
TST26 =	015336	3753	3782#	6405																
TST27 =	016010	3856	3885#	6406																
TST3 =	005034	2393	2399	2413#	6386															
TST30 =	016462	3959	3989#	6407																
TST31 =	017134	4063	4093#	6408																
TST32 =	017606	4167	4195#	6409																
TST33 =	020374	4285	4321#	6410																
TST34 =	021162	4411	4440#	6411																
TST35 =	021750	4530	4547#	6412																
TST36 =	022440	4621	4637#	6413																
TST37 =	023130	4711	4727#	6414																
TST4 =	005266	2440	2446	2459#	6387															
TST40 =	023620	4801	4817#	6415																
TST41 =	024310	4891	4910#	6416																
TST42 =	025024	5022#	6417																	
TST43 =	025550	5136#	6418																	
TST44 =	026274	5242#	6419																	
TST45 =	027532	5416	5435#	6420																
TST46 =	031002	5613	5629#	6421																
TST47 =	031566	5757#	6422																	
TST5 =	005516	2486	2492	2505#	6388															
TST50 =	032440	5770	5897#	6423																
TST51 =	032470	5917#	6424																	
TST6 =	005746	2532	2538	2552#	6389															
TST7 =	006176	2579	2585	2602#	6390															
TYPOS =	104405	6135	6142	7478#																
TYPE =	104401	2234	2251	2254	2257	2264	2268	2275	2284	5766	5769	6129	6136	6143						
		6297	6457	6458	6462	6463	6470	6481	6483	6493	6494	6496	6504	6799						
		6807	6876	6972	7047	7095	7107	7161	7162	7165	7176	7186	7197	7216						

N14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 183
 DZR6DA.P11 28-SEP-76 15:50 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0192

		7264	7270	7275	7279	7284	7285	7287	7290	7294	7360	7362	7434	7474*
TYPERR	035330	6445*	6806											
TYP0C =	104402	2256	2267	2283	5768	6478	7164	7475*						
TYP0N =	104404	7477*												
TYP0S =	104403	7476*												
T.ASOF	003156	2112*												
T.BA	003144	2107*	2842*	2858	5391*	5406	5588*	5603	7498	7527				
T.CS1	003140	2105*	2337*	2343	2383*	2389	2430*	2436	2476*	2482	2522*	2528	2569*	2575
		2625*	2628	2682*	2685	2736*	2739	2791*	2794	2839*	2849	2876*	2879	2941*
		2942	2949*	2950	2965*	2966	3376*	3382	3458*	3464	3547*	3550	3573*	3577
		3659*	3662	3740*	3746	3837*	3840	3940*	3943	4044*	4047	4148*	4151	4254*
		4257	4280*	4284	4380*	4383	4406*	4410	4499*	4502	4525*	4529	4600*	4606
		4616*	4620	4690*	4696	4706*	4710	4780*	4786	4796*	4800	4870*	4876	4886*
		4890	5383*	5397	5580*	5594	7495	7498	7502	7507	7513	7517	7524	
T.CS2	003150	2109*	2840*	2855	2877*	2885	2986*	3377*	3385	3459*	3467	3548*	3553	3574*
		3660*	3665	3741*	3749	3838*	3843	3941*	3946	4045*	4050	4149*	4154	4255*
		4260	4281*	4381*	4386	4407*	4500*	4505	4526*	4601*	4609	4617*	4691*	4699
		4707*	4781*	4789	4797*	4871*	4879	4887*	5384*	5400	5581*	5597	7498	7502
		7507	7513	7517	7524									
T.DA	003146	2108*	3135*	3137	3216*	3218	3297*	3299	5390*	5415	5587*	5612	7511	7527
T.DB	003162	2114*	2872*	2891	7505									
T.DCYL	003160	2113*	3136*	3140	3217*	3221	3298*	3302	5389*	5412	5586*	5609	7511	7527
T.DS	003152	2110*												
T.ECPS	003172	2118*												
T.ECPT	003174	2119*	4990*	4991	5103*	5105	5217*	5219	5325*	5326	5518*	5519	5721*	5723
		5870*	5872	6010*	6012	6044*	6046	6103*	6104	7522	7530			
T.ER	003154	2111*	2843*	2864	2878*	2888	2987*	3378*	3388	3450*	3470	3549*	3556	3575*
		3661*	3668	3742*	3752	3839*	3846	3942*	3949	4046*	4053	4150*	4157	4256*
		4263	4282*	4382*	4389	4408*	4501*	4508	4527*	4602*	4612	4618*	4692*	4702
		4708*	4782*	4792	4798*	4872*	4882	4888*	5385*	5403	5582*	5600	7498	7502
		7507	7513	7517	7524									
T.MR1	003164	2115*	2979*	2981	3061*	3063	3559*	3560	3849*	3850	3952*	3953	4056*	4057
		4160*	4161	4266*	4267	4392*	4393	4511*	4512	6628*	6629	6642*	6643	6669*
		6670	6682*	6683	7492	7509	7520							
T.MR2	003166	2116*	2338*	2349	2384*	2395	2431*	2442	2477*	2498	2523*	2534	2570*	2581
		2626*	2634	2683*	2691	2737*	2745	2792*	2800	7495				
T.MR3	003170	2117*	2339*	2352	2385*	2398	2432*	2445	2478*	2491	2524*	2537	2571*	2584
		2627*	2637	2684*	2694	2738*	2748	2793*	2803	7495				
T.SPAP	003176	2120*												
T.WC	003142	2106*	2841*	2861	5392*	5409	5589*	5606	7498	7527				
UFE =	000400	1083*												
UNLOAD =	000007	1047*												
UNS =	040000	1108*												
UPE =	020000	1088*												
VRCHDR	055176	3610*	3611	3613*	3616*	3637	7516	8922*						
VV =	000100	1119*												
WCE =	040000	1089*												
WLE =	004000	1105*												
WRDATA =	000023	1053*	2376	2386	2515	2525	2833	2844	2926	2935	3017	3096	3177	3258
		3337	3379	3419	3461	3500	3509	3597	3621	3701	3743	3791	3800	3894
		3903	3998	4007	4102	4111	4207	4216	4333	4342	4452	4461	4556	4603
		4646	4693	4917	5029	5143	5250	5386	5443	5583	5641	5790	5930	
WRDCNT	003264	2155*	2869*	2894*	7505									
WRHEAD =	000027	1055*												
WRL =	004000	1122*												
WRTBIT	036302	4964	4970	4988	5076	5082	5101	5190	5196	5215	5297	5303	5322	5354

SPURDN	042234	2198	7416#	7431																
SPURUP	042262	7417	7425#																	
SOLES	001210	1323#	6832	6912	7216	7287	7304	7362	7365											
SROCHR	041402	7229#	7483																	
SRODEC=	***** U	7486																		
SROLIN	041472	7252#	7484																	
SROCT	042000	7326#	7485																	
SROSZ =	000C10	7245#																		
SRESRE	042176	7398#	7487																	
SRTNOC	034110	6154#																		
SRTA =	***** U	7488																		
SSAVRE	042140	7382#	7486																	
SSCOPE	034650	2192	6323#																	
SSETUP=	000137	2168#	2191	2192	2194	2196	2198	2200	2201	2202	2204	2232	2235	6119						
		6324	6793	6819	6827	7099	7104	7105	7135	7311										
		2168#																		
SSTUP =	177777	6334	6373#																	
SSVLAD	035110	1172#	1177	1243#	1248															
SSVPC =	000220	881#	893	897	898	899	900	901	902	903	904	1320	1321	1322						
SSWR	= 167400	2201	2202	2204	2205	2321	2367	2414	2460	2506	2552	2603	2660	2714						
		2769	2827	2920	3008	3081	3162	3243	3329	3411	3489	3595	3693	3783						
		3886	3990	4094	4196	4322	4441	4548	4638	4728	4818	4911	5023	5137						
		5243	5436	5630	5758	5898	5918	6114	6120	6147	6153	6155	6315	6316						
		6317	6318	6319	6325	6337	6339	6340	6353	6354	6355	6362	6363	6364						
		6376	6379	6382	6784	6785	6786	6787	6798	6797	6804	6816	6820	6832						
SSWREG	001236	1343#	2225																	
SSWRMK=	000000	904	905	6319	6320	6343														
SSWOBT	035162	6349	6383#																	
STESTN	001220	1334#	5767	6374*	6446*	6447*	7491	7492	7495	7498	7502	7505	7507	7509						
		7511	7513	7517	7520	7522	7524	7530												
STIMES	001200	1320#	2201*	2321*	2367*	2414*	2460*	2506*	2553*	2603*	2660*	2714*	2769*	2827*						
		2920*	3008*	3081*	3162*	3243*	3329*	3411*	3489*	3595*	3693*	3783*	3886*	3990*						
		4094*	4196*	4322*	4441*	4548*	4638*	4728*	4818*	4911*	5023*	5137*	5243*	5436*						
		5630*	5758*	5763*	5898*	5918*	6120*	6362*	6369	6372*	6382									
STKB	001146	1305#	7059	7080	7091	7116*	7144	7171												
STKCNT	040542	7060#	7075*	7105	7122*	7236	7238*													
STKINT	040552	2178	7075#	7096	7157															
STKQEN=	040551	7064#	7130	7241																
STKQIN	040544	7061#	7076*	7077	7128*	7129*	7130	7132*												
STKQOU	040546	7062#	7077*	7239	7240*	7241	7243*													
STKQSR	040550	7063#	7076	7132	7243															
STKS	001144	1304#	7059	7081*	7112*	7114	7120*	7142	7158*	7168	7180*	7200*								
STKSRV	040622	7078	7091#																	
STMPO	001160	1312#	2259*	2261	2263	2270*	2272	2274	2286*	2288	2290	2292*	2293*	2294*						
		2295#	2296*	2298																
STMP1	001162	1313#																		
STMP2	001164	1314#																		
STMP3	001166	1315#																		
STMP4	001170	1316#	3600*	3611	3613	3616	3672*													
STMP5	001172	1317#																		
STMP6	001174	1318#																		
STMP7	001176	1319#																		
STN =	000052	882#	893	2310	2321#	2347	2353	2356	2367#	2393	2399	2402	2414#	2440						
		2446	2449	2460#	2486	2492	2495	2506#	2532	2538	2541	2553#	2579	2585						
		2590	2603#	2648	2660#	2702	2714#	2756	2769#	2808	2814	2827#	2853	2883						
		2902	2920#	2982	2984	2990	3008#	3064	3067	3081#	3148	3162#	3229	3243#						

H15

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P4 MD-11-DZR6DA MACY11 27(1006) 05-OCT-76 09:06 PAGE 191
DZR6DA.P11 28-SEP-76 15:50 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0199

. ABS. 057234 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DZR6DA, DSKZ:DZR6DA.SEQ/CRF/SOL/DOC=DZR6DA
RUN-TIME: 91 86 8 SECONDS
RUN-TIME RATIO: 292/186=1.5
CORE USED: 35K (69 PAGES)

DOCUMENT PAGES: 189

