

RK611
RK06, RK07

RK611 DSKLS PRT 4
CZR6DD0

AH-9110D-MC
FICHE 1 OF 1

MAR 1982
COPYRIGHT © 76-81
MADE IN USA



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

.REM % IDENTIFICATION

PRODUCT CODE: AC-9108D-MC
PRODUCT NAME: CZR6DDO RK611 DSKLS PRT4
DATE: AUGUST 10 1981
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: BRIAN LEBLANC

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

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

COPYRIGHT (C) 1976,1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86

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 (^C) OPERATION
 - 4.3 CONTROL S (^S) OPERATION
 - 4.4 CONTROL Q (^Q) OPERATION

5.0 PROGRAM DESCRIPTION

6.0 ERROR REPORTING

1.0 ABSTRACT

- THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 4
- A. TESTS THE LOADING OF THE DRIVE BUS MESSAGE SHIFT REGISTERS FOR CLASS C COMMANDS.
 - B. TESTS HEADER GENERATION FOR SEARCH OPERATIONS.
 - C. TESTS WRITE DATA NPR TRANSFERS TO SILO.
 - D. TESTS HEADER RECOGNITION.
 - E. TESTS CYLINDER, TRACK, HEADER SEARCH.
 - F. TESTS DETECTION OF ALL HEADER TYPE ERRORS.
 - G. TESTS ECC GENERATION AND WRITING.
 - H. TESTS PARTIAL SECTOR WRITE (ZERO FILL).
 - I. TESTS 18 BIT FORMAT ECC GENERATION AND DATA WRITE.

NO RK06 DRIVE IS REQUIRED FOR PROGRAM EXECUTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 SYSTEM (16K CORE MEMORY)
CONSOLE TERMINAL
DECTAPE, PAPER TAPE READER, OR DECDISK
RK611 CONTROLLER

2.2 PRELIMINARY PROGRAMS

AND SECTOR INCREMENT AFTER SUCCESSFUL

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

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 1 (CZR6A)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (CZR6B)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3 (CZR6C)

3.0 OPERATING PROCEDURES

3.1 LOADING PROCEDURE

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING ABSOLUTE LOADER
OR FROM XXDP MEDIA SUPPORTED BY XXDP.

3.2 STARTING PROCEDURE

LOCATION 200 - START PROGRAM

LOCATION 204 - RESTART PROGRAM

LOCATION 214 - REQUEST BUS ADDRESS, VECTOR ADDRESS, AND
PRIORITY MODIFICATION

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 ='

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

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

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

199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254

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.

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 TH
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
310

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 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 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 68 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 CONTROLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ON 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

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
364
365
366

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.

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
420
421
422

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

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

FOLLOWING 3 WORDS WITH A BAD HEADER VRC:

423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478

479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534

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:

000140

535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590

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 HEADFR 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

THEN SIMULATE A 3 WORD HEADER CONSISTING ON

591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646

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.

647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702

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.

**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 FC
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.

703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758

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.

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.

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

759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814

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

NOTE: SINCE 18 BIT FUNCTIONALITY OF THE RK611 IS NOT
UTILIZED ON THE PDP-11, THE FOLLOWING TESTS, INTENDED
FOR MANUFACTURING USE, WILL NOT BE RUN UNLESS LOCATION
'SEC24' IS PATCHED TO A NON-ZERO VALUE.

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.

TEST 47 18 BIT WRITE DATA (PART 2)

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

815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862

XXXXXX VYYYYY
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.

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.

%

```
863      ; *** REV 003 ***
864      .NLIST  CND,MD,MC
865      .LIST   ME
866      .ENABL  ABS,AMA
867      $SWR=  167400
868      $TN=   1
869      .TITLE  CZR6DD0 RK611 DSKLS PRT4
870      ;*COPYRIGHT (C) 1976,1981
871      ;*DIGITAL EQUIPMENT CORP.
872      ;*MAYNARD, MASS. 01754
873      ;*
874      ;*
875      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
876      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
877      ;*
878      .SBTTL  OPERATIONAL SWITCH SETTINGS
879      ;*
880      ;*      SWITCH                      USE
881      ;*      -----
882      ;*      15                      HALT ON ERROR
883      ;*      14                      LOOP ON TEST
884      ;*      13                      INHIBIT ERROR TYPEOUTS
885      ;*      12                      ABORT PROGRAM AFTER 20 ERRORS
886      ;*      11                      INHIBIT ITERATIONS
887      ;*      10                      BELL ON ERROR
888      ;*      9                       LOOP ON ERROR
889      ;*      8                       LOOP ON TEST IN SWR<7:0>
890      .SBTTL  BASIC DEFINITIONS
891      ;*
892      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
893      STACK= 1100
894      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
895      .EQUIV  IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL
896      ;*
897      ;*MISCELLANEOUS DEFINITIONS
898      HT=    11              ;;CODE FOR HORIZONTAL TAB
899      LF=    12              ;;CODE FOR LINE FEED
900      CR=    15              ;;CODE FOR CARRIAGE RETURN
901      CRLF=  200            ;;CODE FOR CARRIAGE RETURN-LINE FEED
902      PS=    177776         ;;PROCESSOR STATUS WORD
903      .EQUIV  PS,PSW
904      STKLMi= 177774         ;;STACK LIMIT REGISTER
905      PIRQ=   177772         ;;PROGRAM INTERRUPT REQUEST REGISTER
906      DSWR=   177570         ;;HARDWARE SWITCH REGISTER
907      DDISP=  177570         ;;HARDWARE DISPLAY REGISTER
908      ;*
909      ;*GENERAL PURPOSE REGISTER DEFINITIONS
910      R0=    %0              ;;GENERAL REGISTER
911      R1=    %1              ;;GENERAL REGISTER
912      R2=    %2              ;;GENERAL REGISTER
913      R3=    %3              ;;GENERAL REGISTER
914      R4=    %4              ;;GENERAL REGISTER
915      R5=    %5              ;;GENERAL REGISTER
916      R6=    %6              ;;GENERAL REGISTER
917      R7=    %7              ;;GENERAL REGISTER
918      SP=    %6              ;;STACK POINTER
```

```
919          000007          PC=      %7          ;;PROGRAM COUNTER
920
921          ;*PRIORITY LEVEL DEFINITIONS
922          000000          PR0=      0          ;;PRIORITY LEVEL 0
923          000040          PR1=      40         ;;PRIORITY LEVEL 1
924          000100          PR2=     100         ;;PRIORITY LEVEL 2
925          000140          PR3=     140         ;;PRIORITY LEVEL 3
926          000200          PR4=     200         ;;PRIORITY LEVEL 4
927          000240          PR5=     240         ;;PRIORITY LEVEL 5
928          000300          PR6=     300         ;;PRIORITY LEVEL 6
929          000340          PR7=     340         ;;PRIORITY LEVEL 7
930
931          ;*'SWITCH REGISTER' SWITCH DEFINITIONS
932          100000          SW15=    100000
933          040000          SW14=    40000
934          020000          SW13=    20000
935          010000          SW12=    10000
936          004000          SW11=    4000
937          002000          SW10=    2000
938          001000          SW09=    1000
939          000400          SW08=    400
940          000200          SW07=    200
941          000100          SW06=    100
942          000040          SW05=    40
943          000020          SW04=    20
944          000010          SW03=    10
945          000004          SW02=    4
946          000002          SW01=    2
947          000001          SW00=    1
948          .EQUIV SW09,SW9
949          .EQUIV SW08,SW8
950          .EQUIV SW07,SW7
951          .EQUIV SW06,SW6
952          .EQUIV SW05,SW5
953          .EQUIV SW04,SW4
954          .EQUIV SW03,SW3
955          .EQUIV SW02,SW2
956          .EQUIV SW01,SW1
957          .EQUIV SW00,SW0
958
959          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
960          100000          BIT15=  100000
961          040000          BIT14=  40000
962          020000          BIT13=  20000
963          010000          BIT12=  10000
964          004000          BIT11=  4000
965          002000          BIT10=  2000
966          001000          BIT09=  1000
967          000400          BIT08=  400
968          000200          BIT07=  200
969          000100          BIT06=  100
970          000040          BIT05=  40
971          000020          BIT04=  20
972          000010          BIT03=  10
973          000004          BIT02=  4
974          000002          BIT01=  2
```

```
975          000001          BIT00= 1
976          .EQUIV BIT09,BIT9
977          .EQUIV BIT08,BIT8
978          .EQUIV BIT07,BIT7
979          .EQUIV BIT06,BIT6
980          .EQUIV BIT05,BIT5
981          .EQUIV BIT04,BIT4
982          .EQUIV BIT03,BIT3
983          .EQUIV BIT02,BIT2
984          .EQUIV BIT01,BIT1
985          .EQUIV BIT00,BIT0
986
987          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
988          000004          ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
989          000010          RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
990          000014          TBITVEC=14        ;; 'T' BIT
991          000014          TRTVEC= 14         ;; TRACE TRAP
992          000014          BPTVEC= 14         ;; BREAKPOINT TRAP (BPT)
993          000020          IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
994          000024          PWRVEC= 24         ;; POWER FAIL
995          000030          EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
996          000034          TRAPVEC=34        ;; "TRAP" TRAP
997          000060          TKVEC= 60          ;; TTY KEYBOARD VECTOR
998          000064          TPVEC= 64         ;; TTY PRINTER VECTOR
999          000240          PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
1000         000210          AVECT1= 210       ;DEFINE RK611 VECTOR ADDRESS
1001         000240          APRIOR= PR5       ;DEFINE RK611 PRIORITY
1002         177440          ABASE= 177440     ;DEFINE BASE OF RK611 REGISTERS
1003
1004          .SBTTL RK611 CONTROLLER REGISTER DEFINITION
1005
1006         000000          RKCS1= 0          ;CONTROL AND STATUS REGISTER 1
1007         000002          RKWC= 2           ;WORD COUNT REGISTER
1008         000004          RKBA= 4           ;BUS ADDRESS REGISTER
1009         000006          RKDA= 6           ;DESIRED TRACK SECTOR REGISTER
1010         000010          RKCS2= 10         ;CONTROL AND STATUS REGISTER 2
1011         000012          RKDS= 12          ;DRIVE STATUS REGISTER
1012         000014          RKER= 14          ;ERROR REGISTER
1013         000016          RKASOF= 16         ;ATTENTION SUMMARY AND OFFSET REGISTER
1014         000020          RKDCYL= 20        ;DESIRED CYLINDER REGISTER
1015         000024          RKDB= 24          ;DATA BUFFER
1016         000026          RKMR1= 26         ;MAINTENANCE REGISTER 1
1017         000034          RKMR2= 34         ;MAINTENANCE REGISTER 2
1018         000036          RKMR3= 36         ;MAINTENANCE REGISTER 3
1019         000030          RKECPS= 30        ;ECC POSITION INFORMATION
1020         000032          RKECPT= 32        ;ECC PATTERN INFORMATION
1021         000022          RKSPAR= 22        ;SPARE REGISTER
1022
1023          .SBTTL DRIVE COMMANDS
1024
1025         000001          SELDRV= 01         ;SELECT DRIVE
1026         000003          PACK= 03          ;PACK ACKNOWLEDGE
1027         000005          CLEAR= 05         ;DRIVE CLEAR
1028         000007          UNLOAD= 07        ;UNLOAD
1029         000011          SRTSPL= 11        ;START SPINDLE
1030         000013          RECAL= 13         ;RECALIBRATE
```

1031	000015	OFFSET= 15	:OFFSET
1032	000017	SEEK= 17	:SEEK
1033	000021	RDDATA= 21	:READ DATA
1034	000023	WRDATA= 23	:WRITE DATA
1035	000025	RDHEAD= 25	:READ HEADER
1036	000027	WRHEAD= 27	:WRITE HEADER AND DATA
1037	000031	WRTCHK= 31	:WRITE CHECK
1038	000300	INTR= 300	:GENERATE INTERRUPT TO CPU
1039			
1040		.SBTTL CONTROL AND STATUS REGISTER 1 BITS	
1041			
1042	000001	GO= BIT0	:GO BIT
1043	000100	IE= BIT6	:INTERRUPT ENABLE
1044	000200	RDY= BIT7	:CONTROLLER READY
1045	000400	BA16= BIT8	:BUS ADDRESS BIT 16
1046	001000	BA17= BIT9	:BUS ADDRESS BIT 17
1047	002000	CDT= BIT10	:CONTROLLER DRIVE TYPE (0=RK06)
1048	004000	CTO= BIT11	:CONTROLLER TIMED OUT WAITING FOR : DRIVE RESPONSE
1049			
1050	010000	CFMT= BIT12	:CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1051	020000	SPAR= BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1052	040000	DI= BIT14	:DRIVE INTERRUPT
1053	100000	CERR= BIT15	:CONTROLLER ERROR
1054	100000	CCLR= BIT15	:CONTROLLER CLEAR
1055			
1056		.SBTTL CONTROL AND STATUS REGISTER 2 BITS	
1057			
1058	000007	DRVMSK= 7	:MASK FOR DRIVE SELECTION CODE
1059	000010	RLS= BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
1060	000020	BAI= BIT4	:BUS ADDRESS INCREMENT INHIBIT
1061	000040	SCLR= BIT5	:CLEAR CONTROLLER AND ALL DRIVES
1062	000100	IR= BIT6	:INPUT READY
1063	000200	OR= BIT7	:OUTPUT READY
1064	000400	UFE= BIT8	:UNIT FIELD ERROR
1065	001000	MDS= BIT9	:MULTIPLE DRIVE SELECT
1066	002000	PGE= BIT10	:PROGRAMMING ERROR
1067	004000	NEM= BIT11	:NON-EXISTENT MEMORY
1068	010000	NED= BIT12	:NON-EXISTENT DRIVE
1069	020000	UPE= BIT13	:UNIBUS PARITY ERROR
1070	040000	WCE= BIT14	:WRITE CHECK ERROR
1071	100000	DLT= BIT15	:DATA LATE ERROR
1072			
1073		.SBTTL ERROR REGISTER BIT DEFINITION	
1074			
1075	000001	ILF= BIT0	:ILLEGAL FUNCTION CODE
1076	000002	SKI= BIT1	:SEEK INCOMPLETE
1077	000004	NXF= BIT2	:NON-EXECUTABLE DRIVE FUNCTION
1078	000010	DRPAR= BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
1079	000020	FMTE= BIT4	:FORMAT ERROR
1080	000040	DTYPE= BIT5	:DRIVE TYPE ERROR
1081	000100	ECH= BIT6	:ECC HARD
1082	000200	BSE= BIT7	:BAD SECTOR ERROR
1083	000400	HVRC= BIT8	:HEADER VRC ERROR
1084	001000	COE= BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1085	002000	IDAE= BIT10	:INVALID DISK ADDRESS ERROR
1086	004000	WLE= BIT11	:WRITE LOCK ERROR


```
1087      010000      DTE=   BIT12      ;DRIVE TIMING ERROR
1088      020000      OPI=   BIT13      ;OPERATION (SEARCH) INCOMPLETE
1089      040000      UNS=   BIT14      ;DRIVE UNSAFE
1090      100000      DCK=   BIT15      ;DATA CHECK

1091
1092      .SBTTL  STATUS REGISTER BIT DEFINITION
1093
1094      000001      DRA=   BIT0       ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1095      ; THIS BIT IS RESET)
1096      000004      OFST=  BIT2       ;DRIVE OFFSET
1097      000010      ACLO=  BIT3       ;AC LOW
1098      000020      SPDLSS= BIT4       ;SPEED LOSS
1099      000040      DROT=  BIT5       ;DRIVE OFF TRACK
1100      000100      VV=    BIT6       ;VOLUME VALID
1101      000200      DRDY=  BIT7       ;DRIVE READY
1102      000400      DDT=   BIT8       ;DRIVE TYPE (0=RK06)
1103      004000      WRL=   BIT11      ;WRITE LOCK
1104      020000      PIP=   BIT13      ;POSITIONING IN PROGRESS
1105      040000      DSC=   BIT14      ;DRIVE STATUS CHANGE
1106      100000      SVAL=  BIT15      ;STATUS VALID

1107
1108      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1109
1110      000017      MESMSK= 17      ;MESSAGE MASK
1111
1112      000020      PAT=   BIT4       ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1113      000040      DMD=   BIT5       ;DIAGNOSTIC MODE
1114      000100      MSP=   BIT6       ;MAINTENANCE SECTOR PULSE
1115      000200      MIND=  BIT7       ;MAINTENANCE INDEX
1116      000400      MCLK=  BIT8       ;MAINTENANCE CLOCK
1117      001000      MERD=  BIT9       ;MAINTENANCE ENCODED READ DATA
1118      002000      MEWD=  BIT10      ;MAINTENANCE ENCODED WRITE DATA
1119      004000      PCA=   BIT11      ;PRECOMPENSATION ADVANCE
1120      010000      PCD=   BIT12      ;PRECOMPENSATION DELAY
1121      020000      ECCW=  BIT13      ;ECC WORD IS BEING READ OR WRITTEN
1122      040000      WRTGAT= BIT14      ;WRITE GATE
1123      100000      RDGATE= BIT15      ;READ GATE

1124
1125      .SBTTL  TRANSMITTED MESSAGE A
1126
1127      000020      S.SEK=  BIT4       ;SEEK COMMAND
1128      000040      S.RECL= BIT5       ;RECALIBRATE COMMAND
1129      000100      S.STSP= BIT6       ;START SPINDLE COMMAND
1130      000200      S.RTC=  BIT7       ;DRIVE RETURN TO CENTERLINE COMMAND
1131      000400      S.CLR=  BIT8       ;CLEAR ERROR AND DSC
1132      001000      S.FMT=  BIT9       ;FORMAT
1133      002000      S.UNLD= BIT10      ;UNLOAD
1134      004000      S.PACK= BIT11      ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1135
1136
1137      000000      .=0
1138      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1139      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1140      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1141
1142      000174      000174      .=174
000174  000000      DISPREG: .WORD 0      ;:SOFTWARE DISPLAY REGISTER
```

```
1143 000176 000000 SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
1144 .SBTTL STARTING ADDRESS(ES)
1145 000200 000137 003336 JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1146 000204 000137 003326 JMP RESTRT ;;JUMP TO RESTART ROUTINE
1147 000214 000214 .=214
1148 000214 000137 003316 JMP PARM ;;JUMP TO OPERATOR ASSIGNED PARMETERS
1149 .SBTTL ACT11 HOOKS
1150
1151 ;:*****
1152 ;:HOOKS REQUIRED BY ACT11
1153 000220 $SVPC=. ;SAVE PC
1154 000046 .=46
1155 000046 033150 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1156 000052 .=52
1157 000052 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
1158 000220 .=$SVPC ;; RESTORE PC
1159 001000 .-1000
1160 .SBTTL APT PARAMETER BLOCK
1161
1162 ;:*****
1163 ;:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1164 ;:*****
1165 001000 .SX=. ;;SAVE CURRENT LOCATION
1166 000024 .-24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1167 000024 000200 200 ;;FOR APT START UP
1168 000044 .=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1169 000044 001000 $APTHDR ;;POINT TO APT HEADER BLOCK
1170 001000 .=$X ;;RESET LOCATION COUNTER
1171 ;:*****
1172 ;:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1173 ;:INTERFACE SPEC.
1174
1175 001000 $APTHD:
1176 001000 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1177 001002 001214 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1178 001004 000000 $STM: .WORD ;;RUN TIM OF LONGEST TEST
1179 001006 000000 $PASTM: .WORD ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1180 001010 000000 $UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1181 001012 000032 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237

.SBTTL COMMON TAGS

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

.=1100

\$CMTAG: .WORD 0 ;:START OF COMMON TAGS
\$TSTNM: .BYTE 0 ;:CONTAINS THE TEST NUMBER
\$ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
\$ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
\$LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
\$ERTTL: .WORD 0 ;: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 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
\$BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
\$GDDAT: .WORD 0 ;:CONTAINS 'GOOD' DATA
\$BDDAT: .WORD 0 ;:CONTAINS 'BAD' DATA
 .WORD 0 ;:RESERVED--NOT TO BE USED
\$AUTOB: .BYTE 0 ;:AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0 ;:INTERRUPT MODE INDICATOR
 .WORD 0
\$SWR: .WORD DSWR ;:ADDRESS OF SWITCH REGISTER
\$DISP_AY: .WORD DDISP ;:ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 ;:TTY KBD STATUS
\$TKB: 177562 ;:TTY KBD BUFFER
\$TPS: 177564 ;:TTY PRINTER STATUS REG. ADDRESS
\$TPB: 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'
\$TPFLG: .BYTE 0 ;: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
\$TMP0: .WORD 0 ;:USER DEFINED
\$TMP1: .WORD 0 ;:USER DEFINED
\$TMP2: .WORD 0 ;:USER DEFINED
\$TMP3: .WORD 0 ;:USER DEFINED
\$TMP4: .WORD 0 ;:USER DEFINED
\$TMP5: .WORD 0 ;:USER DEFINED
\$TMP6: .WORD 0 ;:USER DEFINED
\$TMP7: .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

000377

1238	001214		\$MAIL:		:: APT MAILBOX
1239	001214	000000	\$MSGTY:	.WORD	AMSGTY :: MESSAGE TYPE CODE
1240	001216	000000	\$FATAL:	.WORD	AFATAL :: FATAL ERROR NUMBER
1241	001220	000000	\$TESTN:	.WORD	ATESTN :: TEST NUMBER
1242	001222	000000	\$PASS:	.WORD	APASS :: PASS COUNT
1243	001224	000000	\$DEVCT:	.WORD	ADEVCT :: DEVICE COUNT
1244	001226	000000	\$UNIT:	.WORD	AUNIT :: I/O UNIT NUMBER
1245	001230	000000	\$MSGAD:	.WORD	AMSGAD :: MESSAGE ADDRESS
1246	001232	000000	\$MSGLG:	.WORD	AMSGLG :: MESSAGE LENGTH
1247	001234		\$ETABLE:		:: APT ENVIRONMENT TABLE
1248	001234	000	\$ENV:	.BYTE	AENV :: ENVIRONMENT BYTE
1249	001235	000	\$ENVM:	.BYTE	AENVM :: ENVIRONMENT MODE BITS
1250	001236	000000	\$SWREG:	.WORD	ASWREG :: APT SWITCH REGISTER
1251	001240	000000	\$USWR:	.WORD	AUSWR :: USER SWITCHES
1252	001242	000000	\$CPUOP:	.WORD	ACPUOP :: CPU TYPE, OPTIONS
1253			*		BITS 15-11=CPU TYPE
1254			*		11/04=01,11/05=02,11/20=03,11/40-04,11/45=05
1255			*		11/70=06,PDQ=07,Q=10
1256			*		BIT 10=REAL TIME CLOCK
1257			*		BIT 9=FLOATING POINT PROCESSOR
1258			*		BIT 8=MEMORY MANAGEMENT
1259	001244	000	\$MAMS1:	.BYTE	AMAMS1 :: HIGH ADDRESS,M.S. BYTE
1260	001245	000	\$MTYP1:	.BYTE	AMTYP1 :: MEM. TYPE,BLK#1
1261			*		MEM.TYPE BYTE -- (HIGH BYTE)
1262			*		900 NSEC CORE=001
1263			*		300 NSEC BIPOLAR=002
1264			*		500 NSEC MOS=003
1265	001246	000000	\$MADR1:	.WORD	AMADR1 :: HIGH ADDRESS,BLK#1
1266			*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1267	001250	000	\$MAMS2:	.BYTE	AMAMS2 :: HIGH ADDRESS,M.S. BYTE
1268	001251	000	\$MTYP2:	.BYTE	AMTYP2 :: MEM.TYPE,BLK#2
1269	001252	000000	\$MADR2:	.WORD	AMADR2 :: MEM.LAST ADDRESS,BLK#2
1270	001254	000	\$MAMS3:	.BYTE	AMAMS3 :: HIGH ADDRESS,M.S.BYTE
1271	001255	000	\$MTYP3:	.BYTE	AMTYP3 :: MEM.TYPE,BLK#3
1272	001256	000000	\$MADR3:	.WORD	AMADR3 :: MEM.LAST ADDRESS,BLK#3
1273	001260	000	\$MAMS4:	.BYTE	AMAMS4 :: HIGH ADDRESS,M.S.BYTE
1274	001261	000	\$MTYP4:	.BYTE	AMTYP4 :: MEM.TYPE,BLK#4
1275	001262	000000	\$MADR4:	.WORD	AMADR4 :: MEM.LAST ADDRESS,BLK#4
1276	001264	000210	\$VECT1:	.WORD	AVECT1 :: INTERRUPT VECTOR#1,BUS PRIORITY#1
1277	001266	000000	\$VECT2:	.WORD	AVECT2 :: INTERRUPT VECTOR#2BUS PRIORITY#2
1278	001270	177440	\$BASE:	.WORD	ABASE :: BASE ADDRESS OF EQUIPMENT UNDER TEST
1279	001272	000000	\$DEVM:	.WORD	ADEVCT :: DEVICE MAP
1280	001274	000000	\$CDW1:	.WORD	ACDW1 :: CONTROLLER DESCRIPTION WORD#1
1281	001276	000000	\$CDW2:	.WORD	ACDW2 :: CONTROLLER DESCRIPTION WORD#2
1282	001300		\$ETEND:		
1283			.MEXIT		

1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298 001300
1299
1300 001300 044565
1301 001302 042572
1302 001304 041132
1303 001306 041466
1304
1305 001310 000000
1306 001312 000000
1307 001314 041136
1308 001316 041472
1309
1310
1311 001320 044632
1312 001322 050526
1313 001324 041160
1314 001326 041516
1315
1316
1317 001330 044632
1318 001332 050544
1319 001334 041160
1320 001336 041516
1321
1322
1323 001340 044632
1324 001342 050565
1325 001344 041160
1326 001346 041516
1327
1328
1329 001350 044707
1330 001352 050526
1331 001354 041160
1332 001356 041516
1333
1334
1335 001360 044707
1336 001362 050544
1337 001364 041160
1338 001366 041516
1339

.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
CS1 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
CS1 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

1340			:	MESS B INCORRECT
1341	001370	044707	:	EM301
1342	001372	050565	:	EM4002
1343	001374	041160	:	DT003
1344	001376	041516	:	DF003
1345			:	ERROR 11: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1346			:	CS1 INCORRECT
1347	001400	044765	:	EM302
1348	001402	050526	:	EM4000
1349	001404	041160	:	DT003
1350	001406	041516	:	DF003
1351			:	ERROR 12: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1352			:	MESS A INCORRECT
1353	001410	044765	:	EM302
1354	001412	050544	:	EM4001
1355	001414	041160	:	DT003
1356	001416	041516	:	DF003
1357			:	ERROR 13: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1358			:	MESS B INCORRECT
1359	001420	044765	:	EM302
1360	001422	050565	:	EM4002
1361	001424	041160	:	DT003
1362	001426	041516	:	DF003
1363			:	ERROR 14: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1364			:	CS1 INCORRECT
1365	001430	045044	:	EM303
1366	001432	050526	:	EM4000
1367	001434	041160	:	DT003
1368	001436	041516	:	DF003
1369			:	ERROR 15: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1370			:	MESS A INCORRECT
1371	001440	045044	:	EM303
1372	001442	050544	:	EM4001
1373	001444	041160	:	DT003
1374	001446	041516	:	DF003
1375			:	ERROR 16: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1376			:	MESS B INCORRECT
1377	001450	045044	:	EM303
1378	001452	050565	:	EM4002
1379	001454	041160	:	DT003
1380	001456	041516	:	DF003
1381			:	ERROR 17: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1382			:	CS1 INCORRECT
1383	001460	045122	:	EM304
1384	001462	050526	:	EM4000
1385	001464	041160	:	DT003
1386	001466	041516	:	DF003
1387			:	ERROR 20: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1388			:	MESS A INCORRECT
1389	001470	045122	:	EM304
1390	001472	050544	:	EM4001
1391	001474	041160	:	DT003
1392	001476	041516	:	DF003
1393			:	ERROR 21: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1394			:	MESS B INCORRECT
1395	001500	045122	:	EM304

1396	001502	050565	EM4002
1397	001504	041160	DT003
1398	001506	041516	DF003
1399	:	:	ERROR 22: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1400	:	:	CS1 INCORRECT
1401	001510	045201	EM305
1402	001512	050526	EM4000
1403	001514	041160	DT003
1404	001516	041516	DF003
1405	:	:	ERROR 23: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1406	:	:	MESS A INCORRECT
1407	001520	045201	EM305
1408	001522	050544	EM4001
1409	001524	041160	DT003
1410	001526	041516	DF003
1411	:	:	ERROR 24: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1412	:	:	MESS B INCORRECT
1413	001530	045201	EM305
1414	001532	050565	EM4002
1415	001534	041160	DT003
1416	001536	041516	DF003
1417	:	:	ERROR 25: ATTEMPTING TO CHECK HEADER GENERATION
1418	:	:	WITH VARIOUS CYLINDER VALUES
1419	:	:	CS1 INCORRECT
1420	001540	045261	EM306
1421	001542	050526	EM4000
1422	001544	041160	DT003
1423	001546	041542	DF025
1424	:	:	ERROR 26: ATTEMPTING TO CHECK HEADER GENERATION
1425	:	:	WITH VARIOUS CYLINDER VALUES
1426	:	:	FIRST WORD OF HEADER INCORRECT
1427	001550	045261	EM306
1428	001552	050606	EM4003
1429	001554	041160	DT003
1430	001556	041542	DF025
1431	:	:	ERROR 27: ATTEMPTING TO CHECK HEADER GENERATION
1432	:	:	WITH VARIOUS CYLINDER VALUES
1433	:	:	SECOND WORD OF HEADER INCORRECT
1434	001560	045261	EM306
1435	001562	050636	EM4004
1436	001564	041160	DT003
1437	001566	041542	DF025
1438	:	:	ERROR 30: ATTEMPTING TO CHECK HEADER GENERATION
1439	:	:	WITH VARIOUS TRACK VALUES
1440	:	:	CS1 INCORRECT
1441	001570	045363	EM307
1442	001572	050526	EM4000
1443	001574	041160	DT003
1444	001576	041542	DF025
1445	:	:	ERROR 31: ATTEMPTING TO CHECK HEADER GENERATION
1446	:	:	WITH VARIOUS TRACK VALUES
1447	:	:	FIRST WORD OF HEADER INCORRECT
1448	001600	045363	EM307
1449	001602	050606	EM4003
1450	001604	041160	DT003
1451	001606	041542	DF025

1452	:	ERROR 32: ATTEMPTING TO CHECK HEADER GENERATION
1453	:	WITH VARIOUS TRACK VALUES
1454	:	SECOND WORD OF HEADER INCORRECT
1455	001610 045363	EM307
1456	001612 050636	EM4004
1457	001614 041160	DT003
1458	001616 041542	DF025
1459	:	ERROR 33: ATTEMPTING TO CHECK HEADER GENERATION
1460	:	WITH VARIOUS SECTOR VALUES
1461	:	CS1 INCORRECT
1462	001620 045462	EM308
1463	001622 050526	EM4000
1464	001624 041160	DT003
1465	001626 041542	DF025
1466	:	ERROR 34: ATTEMPTING TO CHECK HEADER GENERATION
1467	:	WITH VARIOUS SECTOR VALUES
1468	:	FIRST WORD OF HEADER INCORRECT
1469	001630 045462	EM308
1470	001632 050606	EM4003
1471	001634 041160	DT003
1472	001636 041542	DF025
1473	:	ERROR 35: ATTEMPTING TO CHECK HEADER GENERATION
1474	:	WITH VARIOUS SECTOR VALUES
1475	:	SECOND WORD OF HEADER INCORRECT
1476	001640 045462	EM308
1477	001642 050636	EM4004
1478	001644 041160	DT003
1479	001646 041542	DF025
1480	:	ERROR 36: ATTEMPTING TO CHECK HEADER GENERATION
1481	:	WITH VARIOUS FORMAT VALUES
1482	:	CS1 INCORRECT
1483	001650 045562	EM309
1484	001652 050526	EM4000
1485	001654 041160	DT003
1486	001656 041542	DF025
1487	:	ERROR 37: ATTEMPTING TO CHECK HEADER GENERATION
1488	:	WITH VARIOUS FORMAT VALUES
1489	:	FIRST WORD OF HEADER INCORRECT
1490	001660 045562	EM309
1491	001662 050606	EM4003
1492	001664 041160	DT003
1493	001666 041542	DF025
1494	:	ERROR 40: ATTEMPTING TO CHECK HEADER GENERATION
1495	:	WITH VARIOUS FORMAT VALUES
1496	:	SECOND WORD OF HEADER INCORRECT
1497	001670 045562	EM309
1498	001672 050636	EM4004
1499	001674 041160	DT003
1500	001676 041542	DF025
1501	:	ERROR 41: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1502	:	CS1 INCORRECT
1503	001700 045662	EM310
1504	001702 050526	EM4000
1505	001704 041200	DT041
1506	001706 041566	DF041
1507	:	ERROR 42: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA

1508			:	CS2 INCORRECT
1509	001710	045662	:	EM310
1510	001712	050666	:	EM4005
1511	001714	041200	:	DT041
1512	001716	041566	:	DF041
1513			:	ERROR 43: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1514			:	ERROR REGISTER INCORRECT
1515	001720	045662	:	EM310
1516	001722	050704	:	EM4006
1517	001724	041200	:	DT041
1518	001726	041566	:	DF041
1519			:	ERROR 44: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1520			:	BUS ADD INCORRECT
1521	001730	045662	:	EM310
1522	001732	050730	:	EM4007
1523	001734	041200	:	DT041
1524	001736	041566	:	DF041
1525			:	ERROR 45: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1526			:	WORD COUNT INCORRECT
1527	001740	045662	:	EM310
1528	001742	050756	:	EM4008
1529	001744	041200	:	DT041
1530	001746	041566	:	DF041
1531			:	ERROR 46: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1532			:	CS1 INCORRECT AFTER READING DATA BUFFER
1533	001750	045662	:	EM310
1534	001752	051003	:	EM4009
1535	001754	041230	:	DT046
1536	001756	041622	:	DF046
1537			:	ERROR 47: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1538			:	CS2 INCORRECT AFTER READING DATA BUFFER
1539	001760	045662	:	EM310
1540	001762	051053	:	EM4010
1541	001764	041230	:	DT046
1542	001766	041622	:	DF046

1543	:	ERROR 50: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1544	:	ERROR REG INCORRECT AFTER READING DATA BUFFER
1545	001770 045662	EM310
1546	001772 051123	EM4011
1547	001774 041230	DT046
1548	001776 041622	DF046
1549	:	ERROR 51: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1550	:	DATA READ FROM MEMORY INCORRECT
1551	002000 045662	EM310
1552	002002 051201	EM4012
1553	002004 041250	DT051
1554	002006 041646	DF051
1555	:	ERROR 52: ATTEMPTING TO CHECK HEADER RECOGNITION
1556	:	CS1 INCORRECT
1557	002010 045747	EM311
1558	002012 050526	EM4000
1559	002014 041262	DT052
1560	002016 041672	DF052
1561	:	ERROR 53: ATTEMPTING TO CHECK HEADER RECOGNITION
1562	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1563	002020 045747	EM311
1564	002022 051241	EM4013
1565	002024 041276	DT053
1566	002026 041716	DF053
1567	:	ERROR 54: ATTEMPTING TO CHECK SECTOR INCREMENT
1568	:	DISK ADDRESS REG INCORRECT
1569	002030 046016	EM312
1570	002032 051307	EM4014
1571	002034 041306	DT054
1572	002036 041742	DF054
1573	:	ERROR 55: ATTEMPTING TO CHECK SECTOR INCREMENT
1574	:	CYLINDER ADDRESS INCORRECT
1575	002040 046016	EM312
1576	002042 051343	EM4015
1577	002044 041306	DT054
1578	002046 041742	DF054
1579	:	ERROR 56: ATTEMPTING TO CHECK TRACK INCREMENT
1580	:	DISK ADDRESS REG INCORRECT
1581	002050 046063	EM313
1582	002052 051307	EM4014
1583	002054 041306	DT054
1584	002056 041742	DF054
1585	:	ERROR 57: ATTEMPTING TO CHECK TRACK INCREMENT
1586	:	CYLINDER ADDRESS INCORRECT
1587	002060 046063	EM313
1588	002062 051343	EM4015
1589	002064 041306	DT054
1590	002066 041742	DF054
1591	:	ERROR 60: ATTEMPTING TO CHECK CYLINDER INCREMENT
1592	:	DISK ADDRESS INCORRECT
1593	002070 046127	EM314
1594	002072 051307	EM4014
1595	002074 041306	DT054
1596	002076 041742	DF054
1597	:	ERROR 61: ATTEMPTING TO CHECK CYLINDER INCREMENT
1598	:	CYLINDER ADDRESS INCORRECT

1599	002100	046127	EM314
1600	002102	051343	EM4015
1601	002104	041306	DT054
1602	002106	041742	DF054
1603	:	:	ERROR 62: ATTEMPTING TO CHECK SECTOR PULSE DETECTION WITH
1604	:	:	WRITE DATA
1605	:	:	MAINT. REG. 1 INCORRECT
1606	002110	046176	EM315
1607	002112	051403	EM4016
1608	002114	041276	DT053
1609	002116	041716	DF053
1610	:	:	ERROR 63: ATTEMPTING TO FORCE BAD SECTOR ERROR
1611	:	:	CS1 INCORRECT
1612	002120	046272	EM316
1613	002122	050526	EM4000
1614	002124	041230	DT046
1615	002126	041622	DF046
1616	:	:	ERROR 64: ATTEMPTING TO FORCE BAD SECTOR ERROR
1617	:	:	CS2 INCORRECT
1618	002130	046272	EM316
1619	002132	050666	EM4005
1620	002134	041230	DT046
1621	002136	041622	DF046
1622	:	:	ERROR 65: ATTEMPTING TO FORCE BAD SECTOR ERROR
1623	:	:	ERROR REG INCORRECT
1624	002140	046272	EM316
1625	002142	050704	EM4006
1626	002144	041230	DT046
1627	002146	041622	DF046
1628	:	:	ERROR 66: ATTEMPTING TO FORCE HEADER VRC ERROR
1629	:	:	WHEN BAD SECTOR PRESENT
1630	:	:	CS1 INCORRECT
1631	002150	046337	EM317
1632	002152	050526	EM4000
1633	002154	041230	DT046
1634	002156	041622	DF046
1635	:	:	ERROR 67: ATTEMPTING TO FORCE HEADER VRC ERROR
1636	:	:	WHEN BAD SECTOR PRESENT
1637	:	:	CS2 INCORRECT
1638	002160	046337	EM317
1639	002162	050666	EM4005
1640	002164	041230	DT046
1641	002166	041622	DF046
1642	:	:	ERROR 70: ATTEMPTING TO FORCE HEADER VRC ERROR
1643	:	:	WHEN BAD SECTOR PRESENT
1644	:	:	ERROR REG INCORRECT
1645	002170	046337	EM317
1646	002172	050704	EM4006
1647	002174	041230	DT046
1648	002176	041622	DF046
1649	:	:	ERROR 71: ATTEMPTING TO FORCE HVRC ERROR
1650	:	:	CS1 INCORRECT
1651	002200	046435	EM318
1652	002202	050526	EM4000
1653	002204	041322	DT071
1654	002206	041766	DF071

1655	:	ERROR 72: ATTEMPTING TO FORCE HVRC ERROR	
1656	:	CS2 INCORRECT	
1657	002210	046435	EM318
1658	002212	050666	EM4005
1659	002214	041322	DT071
1660	002216	041766	DF071
1661	:	ERROR 73: ATTEMPTING TO FORCE HVRC ERROR	
1662	:	ERROR REG INCORRECT	
1663	002220	046435	EM318
1664	002222	050704	EM4006
1665	002224	041322	DT071
1666	002226	041766	DF071
1667	:	ERROR 74: ATTEMPTING TO FORCE OPERATION INCOMPLETE	
1668	:	CS1 INCORRECT	
1669	002230	046474	EM319
1670	002232	050526	EM4000
1671	002234	041350	DT074
1672	002236	042022	DF074
1673	:	ERROR 75: ATTEMPTING TO FORCE OPERATION INCOMPLETE	
1674	:	CS2 INCORRECT	
1675	002240	046474	EM319
1676	002242	050666	EM4005
1677	002244	041350	DT074
1678	002246	042022	DF074
1679	:	ERROR 76: ATTEMPTING TO FORCE OPERATION INCOMPLETE	
1680	:	ERROR REG INCORRECT	
1681	002250	046474	EM319
1682	002252	050704	EM4006
1683	002254	041350	DT074
1684	002256	042022	DF074
1685	:	ERROR 77: ATTEMPTING TO FORCE OPERATION INCOMPLETE	
1686	:	MR1 INCORRECT AFTER GAP IN WRITE DATA	
1687	002260	046474	EM319
1688	002262	051241	EM4013
1689	002264	041372	DT077
1690	002266	042046	DF077
1691	:	ERROR 100: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37	
1692	:	CS1 INCORRECT	
1693	002270	046545	EM320
1694	002272	050526	EM4000
1695	002274	041350	DT074
1696	002276	042022	DF074
1697	:	ERROR 101: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37	
1698	:	CS2 INCORRECT	
1699	002300	046545	EM320
1700	002302	050666	EM4005
1701	002304	041350	DT074
1702	002306	042022	DF074
1703	:	ERROR 102: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37	
1704	:	ERROR REG INCORRECT	
1705	002310	046545	EM320
1706	002312	050704	EM4006
1707	002314	041350	DT074
1708	002316	042022	DF074
1709	:	ERROR 103: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37	
1710	:	MR1 INCORRECT AFTER GAP IN WRITE DATA	

1711	002320	046545	EM320
1712	002322	051241	EM4013
1713	002324	041372	DT077
1714	002326	042046	DF077
1715	:	:	ERROR 104: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1716	:	:	CS1 INCORRECT
1717	002330	046631	EM321
1718	002332	050526	EM4000
1719	002334	041350	DT074
1720	002336	042022	DF074
1721	:	:	ERROR 105: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1722	:	:	CS2 INCORRECT
1723	002340	046631	EM321
1724	002342	050666	EM4005
1725	002344	041350	DT074
1726	002346	042022	DF074
1727	:	:	ERROR 106: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1728	:	:	ERROR REG INCORRECT
1729	002350	046631	EM321
1730	002352	050704	EM4006
1731	002354	041350	DT074
1732	002356	042022	DF074
1733	:	:	ERROR 107: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1734	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1735	002360	046631	EM321
1736	002362	051241	EM4013
1737	002364	041372	DT077
1738	002366	042046	DF077
1739	:	:	ERROR 110: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1740	:	:	CS1 INCORRECT
1741	002370	046715	EM322
1742	002372	050526	EM4000
1743	002374	041350	DT074
1744	002376	042022	DF074
1745	:	:	ERROR 111: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1746	:	:	CS2 INCORRECT
1747	002400	046715	EM322
1748	002402	050666	EM4005
1749	002404	041350	DT074
1750	002406	042022	DF074
1751	:	:	ERROR 112: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1752	:	:	ERROR REG INCORRECT
1753	002410	046715	EM322
1754	002412	050704	EM4006
1755	002414	041350	DT074
1756	002416	042022	DF074
1757	:	:	ERROR 113: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1758	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1759	002420	046715	EM322
1760	002422	051241	EM4013
1761	002424	041372	DT077
1762	002426	042046	DF077
1763	:	:	ERROR 114: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1764	:	:	SECTOR ERROR CS1 INCORRECT
1765	002430	047000	EM323
1766	002432	050526	EM4000

1767	002434	041350	DT074
1768	002436	042022	DF074
1769			ERROR 115: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1770			SECTOR ERROR CS2 INCORRECT
1771	002440	047000	EM323
1772	002442	050666	EM4005
1773	002444	041350	DT074
1774	002446	042022	DF074
1775			ERROR 116: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1776			SECTOR ERROR ERROR INCORRECT
1777	002450	047000	EM323
1778	002452	050704	EM4006
1779	002454	041350	DT074
1780	002456	042022	DF074
1781			ERROR 117: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1782			SECTOR ERROR MR1 INCORRECT
1783	002460	047000	EM323
1784	002462	051241	EM4013
1785	002464	041372	DT077
1786	002466	042046	DF077
1787			ERROR 120: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1788			VRC ERROR CS1 INCORRECT
1789	002470	047073	EM324
1790	002472	050526	EM4000
1791	002474	041350	DT074
1792	002476	042022	DF074
1793			ERROR 121: CHECKING HEADER RECOGNITION WITH PPEVIOUS HEADER
1794			VRC ERROR CS2 INCORRECT
1795	002500	047073	EM324
1796	002502	050666	EM4005
1797	002504	041350	DT074
1798	002506	042022	DF074
1799			ERROR 122: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1800			VRC ERROR ERROR REG INCORRECT
1801	002510	047073	EM324
1802	002512	050704	EM4006
1803	002514	041350	DT074
1804	002516	042022	DF074
1805			ERROR 123: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1806			VRC ERROR MR1 INCORRECT
1807	002520	047073	EM324
1808	002522	051241	EM4013
1809	002524	041372	DT077
1810	002526	042046	DF077
1811			ERROR 124: FORCING BAD SECTOR ERROR WITH PREV HEADER
1812			VRC ERROR CS1 INCORRECT
1813	002530	047166	EM325
1814	002532	050526	EM4000
1815	002534	041350	DT074
1816	002536	042022	DF074
1817			ERROR 125: FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1818			CS2 INCORRECT
1819	002540	047166	EM325
1820	002542	050666	EM4005
1821	002544	041350	DT074
1822	002546	042022	DF074

1823	:	ERROR 126: FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1824	:	ERROR REG INCORRECT
1825	002550 047166	EM325
1826	002552 050704	EM4006
1827	002554 041350	DT074
1828	002556 042022	DF074
1829	:	ERROR 127: FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1830	:	MR1 INCORRECT
1831	002560 047166	EM325
1832	002562 051241	EM4013
1833	002564 041372	DT077
1834	002566 042046	DF077
1835	:	ERROR 130: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1836	:	CS1 INCORRECT
1837	002570 047256	EM326
1838	002572 050526	EM4000
1839	002574 041350	DT074
1840	002576 042022	DF074
1841	:	ERROR 131: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1842	:	CS2 INCORRECT
1843	002600 047256	EM326
1844	002602 050666	EM4005
1845	002604 041350	DT074
1846	002606 042022	DF074
1847	:	ERROR 132: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1848	:	ERROR REG INCORRECT
1849	002610 047256	EM326
1850	002612 050704	EM4006
1851	002614 041350	DT074
1852	002616 042022	DF074
1853	:	ERROR 133: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1854	:	MR1 INCORRECT
1855	002620 047256	EM326
1856	002622 051241	EM4013
1857	002624 041372	DT077
1858	002626 042046	DF077
1859	:	ERROR 134: ATTEMPTING TO CHECK ECC PATTERN CLEARING
1860	:	ECC PATTERN INCORRECT
1861	002630 047475	EM329
1862	002632 051421	EM4017
1863	002634 041404	DT134
1864	002636 042072	DF134
1865	:	ERROR 135: ATTEMPTING TO CHECK ECC GENERATION
1866	:	ECC PATTERN INCORRECT
1867	002640 047546	EM330
1868	002642 051421	EM4017
1869	002644 041404	DT134
1870	002646 042072	DF134
1871	:	ERROR 136: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1872	:	CS1 INCORRECT
1873	002650 047611	EM331
1874	002652 050526	EM4000
1875	002654 041230	DT046
1876	002656 041622	DF046
1877	:	ERROR 137: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1878	:	CS2 INCORRECT

1879	002660	047611	EM331
1880	002662	050666	EM4005
1881	002664	041230	DT046
1882	002666	041622	DF046
1883	:	:	ERROR 140: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1884	:	:	ERROR REG. INCORRECT
1885	002670	047611	EM331
1886	002672	050704	EM4006
1887	002674	041230	DT046
1888	002676	041622	DF046
1889	:	:	ERROR 141: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1890	:	:	CS1 INCORRECT
1891	002700	047676	EM332
1892	002702	050526	EM4000
1893	002704	041230	DT046
1894	002706	041622	DF046
1895	:	:	ERROR 142: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1896	:	:	CS2 INCORRECT
1897	002710	047676	EM332
1898	002712	050666	EM4005
1899	002714	041230	DT046
1900	002716	041622	DF046
1901	:	:	ERROR 143: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1902	:	:	ERROR REG. INCORRECT
1903	002720	047676	EM332
1904	002722	050704	EM4006
1905	002724	041230	DT046
1906	002726	041622	DF046
1907	:	:	ERROR 144: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1908	:	:	CS1 INCORRECT
1909	002730	050057	EM335
1910	002732	050526	EM4000
1911	002734	041416	DT144
1912	002736	042116	DF144
1913	:	:	ERROR 145: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1914	:	:	CS2 INCORRECT
1915	002740	050057	EM335
1916	002742	050666	EM4005
1917	002744	041416	DT144
1918	002746	042116	DF144
1919	:	:	ERROR 146: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1920	:	:	ERROR REG. INCORRECT
1921	002750	050057	EM335
1922	002752	050704	EM4006
1923	002754	041416	DT144
1924	002756	042116	DF144
1925	:	:	ERROR 147: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1926	:	:	BUS ADDRESS INCORRECT
1927	002760	050057	EM335
1928	002762	050730	EM4007
1929	002764	041416	DT144
1930	002766	042116	DF144
1931	:	:	ERROR 150: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1932	:	:	WORD COUNT INCORRECT
1933	002770	050057	EM335
1934	002772	050756	EM4008

1935	002774	041416	DT144
1936	002776	042116	DF144
1937			ERROR 151: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1938			CYLINDER ADD INCORRECT
1939	003000	050057	EM335
1940	003002	051343	EM4015
1941	003004	041416	DT144
1942	003006	042116	DF144
1943			ERROR 152: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1944			DISK ADDRESS INCORRECT
1945	003010	050057	EM335
1946	003012	051307	EM4014
1947	003014	041416	DT144
1948	003016	042116	DF144
1949			ERROR 153: ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR
1950			CS1 INCORRECT
1951	003020	050133	EM336
1952	003022	050526	EM4000
1953	003024	041262	DT052
1954	003026	041672	DF052
1955			ERROR 154: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1956			CS1 INCORRECT
1957	003030	050210	EM337
1958	003032	050526	EM4000
1959	003034	041416	DT144
1960	003036	042116	DF144
1961			ERROR 155: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1962			CS2 INCORRECT
1963	003040	050210	EM337
1964	003042	050666	EM4005
1965	003044	041416	DT144
1966	003046	042116	DF144
1967			ERROR 156: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1968			ERROR REGISTER INCORRECT
1969	003050	050210	EM337
1970	003052	050704	EM4006
1971	003054	041416	DT144
1972	003056	042116	DF144
1973			ERROR 157: ATTEMPTIN PARTIAL SECTOR WRITE WITH ZERO FILL
1974			BUS ADDRESS INCORRECT
1975	003060	050210	EM337
1976	003062	050730	EM4007
1977	003064	041416	DT144
1978	003066	042116	DF144
1979			ERROR 160: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1980			WORD COUNT INCORRECT
1981	003070	050210	EM337
1982	003072	050756	EM4008
1983	003074	041416	DT144
1984	003076	042116	DF144
1985			ERROR 161: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1986			CYLINDER ADDRESS INCORRECT
1987	003100	050210	EM337
1988	003102	051343	EM4015
1989	003104	041416	DT144
1990	003106	042116	DF144

1991	:	ERROR 162: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL	
1992	:	DISK ADDRESS INCORRECT	
1993	003110	050210	EM337
1994	003112	051307	EM4014
1995	003114	041416	DT144
1996	003116	042116	DF144
1997	:	ERROR 163: ATTEMPTING WRITE DATA IN 18 BIT MODE	
1998	:	RKECPT INCORRECT	
1999	003120	050461	EM340
2000	003122	051421	EM4017
2001	003124	041456	DT151
2002	003126	042152	DF151
2003	:	ERROR 164: ATTEMPTING WRITE DATA IN 18 BIT MODE	
2004	:	RKECPT INCORRECT	
2005	003130	050461	EM340
2006	003132	051421	EM4017
2007	003134	041404	DT134
2008	003136	042072	DF134
2009			
2010			.SBTTL TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2011			
2012	003140	000000	T.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
2013	003142	000000	T.WC: .WORD 0 ;WORD COUNT REGISTER
2014	003144	000000	T.BA: .WORD 0 ;BUS ADDRESS REGISTER
2015	003146	000000	T.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2016	003150	000000	T.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2017	003152	000000	T.DS: .WORD 0 ;DRIVE STATUS REGISTER
2018	003154	000000	T.ER: .WORD 0 ;ERROR REGISTER
2019	003156	000000	T.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2020	003160	000000	T.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2021	003162	000000	T.DB: .WORD 0 ;DATA BUFFER
2022	003164	000000	T.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2023	003166	000000	T.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2024	003170	000000	T.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2025	003172	000000	T.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2026	003174	000000	T.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2027	003176	000000	T.SPARE: .WORD 0 ;SPARE REGISTER
2028			
2029			.SBTTL EXPECTED RK611 CONTROLLER REGISTERS
2030			
2031	003200	000000	E.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
2032	003202	000000	E.WC: .WORD 0 ;WORD COUNT REGISTER
2033	003204	000000	E.BA: .WORD 0 ;BUS ADDRESS REGISTER
2034	003206	000000	E.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2035	003210	000000	E.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2036	003212	000000	E.DS: .WORD 0 ;DRIVE STATUS REGISTER
2037	003214	000000	E.ER: .WORD 0 ;ERROR REGISTER
2038	003216	000000	E.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2039	003220	000000	E.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2040	003222	000000	E.DB: .WORD 0 ;DATA BUFFER
2041	003224	000000	E.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2042	003226	000000	E.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2043	003230	000000	E.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2044	003232	000000	E.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2045	003234	000000	E.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2046	003236	000000	E.SPARE: .WORD 0 ;SPARE REGISTER

```
2047 .SBTTL PROGRAM DEFINED VARIABLES
2048
2049 003240 000214 RKVEC: .WORD 214 ;RK611 VECTOR ADDRESS
2050 003242 000240 RKPRI: .WORD PR5 ;RK611 PRIORITY
2051 003244 000000 TRAPPC: .WORD 0 ;TRAP PC FOR MEMORY PARITY OPTION
2052 003246 000000 SRTFLG: .WORD 0 ;START FLAG
2053 ; 0 = 200
2054 ; 1 = 214
2055 ; -1 = 204
2056 003250 000000 ERRCNT: .WORD 0 ;ERROR COUNT FOR ABORT ERROR THRESHOLD
2057 003252 000000 P1.BIT: .WORD 0 ;NEXT BIT
2058 003254 000000 PR.BIT: .WORD 0 ;PRESENT BIT
2059 003256 000000 M1.BIT: .WORD 0 ;BIT-1
2060 003260 000000 M2.BIT: .WORD 0 ;BIT-2
2061 003262 000000 BITCNT: .WORD 0 ;BIT COUNT
2062 003264 000000 WRDCNT: .WORD 0 ;WORD COUNT FOR NPR TRANSFERS
2063 003266 000000 ECCHI: .WORD 0 ;16 MOST SIGNIFICANT BIT OF ECC
2064 003270 000000 ECCLO: .WORD 0 ;16 LEAST SIGNIFICANT BIT OF ECC
2065 003272 000000 ECCXOR: .WORD 0 ;FLAG FOR ECC GENERATION
2066 003274 000000 MEMPAR: .WORD 0 ;FLAG FOR MEMORY PARITY OPTION
2067 003276 000 SECTOR: .BYTE 0 ;SECTOR FOR INCREMENT TEST
2068 003277 000 TRACK: .BYTE 0 ;TRACK FOR INCREMENT TEST
2069 003300 000000 CYLN: .WORD 0 ;CYLINDER FOR INCREMENT TEST
2070 003302 000000 000000 000000 HEADER: .WORD 0,0,0 ;EXPECTED FOR INCREMENT TEST
2071 003310 000000 HDRCNT: .WORD 0 ;NUMBER OF HEADERS FOR OPI'S
2072 003312 000000 SAVSWR: .WORD 0 ;STORAGE FOR SWITCH REGISTER
2073 003314 000000 SEC24: .WORD 0 ;FLAG FOR 18-BIT WRITE DATA TESTS
2074 .SBTTL PROGRAM SETUP
2075
2076 003316 012737 000001 003246 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2077 003324 000406 BR START1
2078
2079 003326 012737 177777 003246 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2080 003334 000402 BR START1
2081
2082 003336 005037 003246 START: CLR SRTFLG ;CLEAR START FLAG
2083 003342 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2084 003344 105037 001103 CLR B $ERFLG ;CLEAR ERROR FLAG
2085 003350 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2086 003354 004737 037216 JSR PC,$TKINT ;INIT KEYBOARD
2087 003360 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2088 003364 012746 003372 MOV #1$,-(SP) ;LOAD START OF PROGRAM
2089 003370 000002 RTI ;LOAD PSW
2090
2091 003372 1$:
2092 .SBTTL INITIALIZE THE COMMON TAGS
2093 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2094 003372 012706 001100 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2095 003376 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2096 003400 022706 001140 CMP #SWR,R6 ;;DONE?
2097 003404 001374 BNE -6 ;;LOOP BACK IF NO
2098 003406 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
2099
2100 003412 012737 033246 000020 ;;INITIALIZE A FEW VECTORS
2101 003420 012737 000340 000022 MOV #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2102 003426 012737 036000 000030 MOV #340,@IOTVEC+2 ;;LEVEL 7
MOV #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
```

```
2103 003434 012737 000340 000032      MOV      #340,@#EMTVEC+2 ;;LEVEL 7
2104 003442 012737 041042 000034      MOV      #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2105 003450 012737 000340 000036      MOV      #340,@#TRAPVEC+2;LEVEL 7
2106 003456 012737 040712 000024      MOV      #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2107 003464 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;LEVEL 7
2108 003472 013737 033014 033006      MOV      $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
2109 003500 005037 001200          CLR      $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
2110 003504 005037 001202          CLR      $ESCAPE        ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2111 003510 112737 000001 001115      MOV      #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
2112 003516 012737 003516 001106      MOV      #,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2113 003524 012737 003524 00111C      MOV      #,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
2114                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2115                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2116 003532 013746 000004          MOV      @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2117 003536 012737 003572 000004      MOV      #64$,@#ERRVEC  ;;SET UP ERROR VECTOR
2118 003544 012737 177570 001140      MOV      #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
2119 003552 012737 177570 001142      MOV      #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2120 003560 022777 177777 175352      CMP      #-1,@SWR       ;;TRY TO REFERENCE HARDWARE SWR
2121 003566 001012          BNE     66$            ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2122                                     ;;AND THE HARDWARE SWR IS NOT - -1
2123 003570 000403          BR      65$           ;;BRANCH IF NO TIMEOUT
2124 003572 012716 003600 64$:      MOV      #65$,(SP)     ;;SET UP FOR TRAP RETURN
2125 003576 000002          RTI
2126 003600 012737 000176 001140 65$:      MOV      #SWREG,SWR    ;;POINT TO SOFTWARE SWR
2127 003606 012737 000174 001142      MOV      #DISPREG,DISPLAY
2128 003614 012637 000004 66$:      MOV      (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2129
2130 003620 005037 001222          CLR      $PASS         ;;CLEAR PASS COUNT
2131 003624 132737 000200 001235      BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
2132 003632 001403          BEQ     67$           ;;YES,USE NON-APT SWITCH
2133 003634 012737 001236 001140      MOV      #SSWREG,SWR   ;;NO,USE APT SWITCH REGISTER
2134 003642
2135 003642 005037 003250 67$:      CLR      ERRCNT
2136 .SBTTL TYPE PROGRAM NAME
2137 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2138 003646 005227 177777          INC      #-1           ;;FIRST TIME?
2139 003652 001045          BNE     68$           ;;BRANCH IF NO
2140 003654 022737 033150 000042      CMP      #ENDAD,@#42   ;;ACT-11?
2141 003662 001441          BEQ     68$           ;;BRANCH IF YES
2142 003664 104401 003732          TYPE    ,69$         ;;TYPE ASCIZ STRING
2143 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2144 003670 005737 000042          TST     @#42         ;;ARE WE RUNNING UNDER XXDP/ACT?
2145 003674 001012          BNE     70$           ;;BRANCH IF YES
2146 003676 123727 001234 000001      CMPB    $ENV,#1       ;;ARE WE RUNNING UNDER APT?
2147 003704 001406          BEQ     70$           ;;BRANCH IF YES
2148 003706 023727 001140 000176      CMP      SWR,#SWREG    ;;SOFTWARE SWITCH REG SELECTED?
2149 003714 001005          BNE     71$           ;;BRANCH IF NO
2150 003716 104406          GTSWR                ;;GET SOFT-SWR SETTINGS
2151 003720 000403          BR      71$
2152 003722 112737 000001 001134 70$:      MOV      #1,$AUTOB    ;;SET AUTO-MODE INDICATOR
2153 003730
2154 003730 000416          BR      68$         ;;GET OVER THE ASCIZ
2155 ;;69$: .ASCII7 <CRLF>/CZR6DD0 RK611 DSKLS PRT4/<CRLF>
2156 003766 68$:
2157 003766 005227 177777          INC      #-1         ;;TYPE RUN TIME MESSAGE FIRST PASS ONLY
2158 003772 001002          BNE     2$
```

2159	003774	104401	042345			TYPE	,OPR006	
2160	004000	022737	000001	003246	2\$:	CMP	#1,SR*FLG	:CHECK IF PARAMETER START
2161	004006	001117				BNE	15\$:NO, CONTINUE SETUP
2162	004010	104401	042176		5\$:	TYPE	,OPR001	:TYPE 'RK611 BUS ADDRESS () ='
2163	004014	013746	001270			MOV	\$BASE,-(SP)	::SAVE \$BASE FOR TYPEOUT
2164	004020	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2165	004022	104401	042225			TYPE	,OPR002	
2166	004026	104412				RDOCT		:GET VALUE
2167	004030	012637	001160			MOV	(SP)+,\$TMP0	
2168	004034	001407				BEQ	7\$:CHECK IF <CR>
2169	004036	022737	160000	001160		CMP	#160000,\$TMP0	:CHECK IF IN I/O PAGE
2170	004044	101361				BHI	5\$	
2171	004046	013737	001160	001270		MOV	\$TMP0,\$BASE	:LOAD NEW BUS ADDRESS
2172	004054	104401	042233		7\$:	TYPE	,OPR003	:TYPE 'RK611 VECTOR ADDRESS () ='
2173	004060	013746	001264			MOV	\$VECT1,-(SP)	:GET VECTOR ADDRESS FOR TYPE OUT
2174	004064	042716	160000			BIC	#160000,(SP)	:CLEAR PRIORITY BITS
2175	004070	104402				TYPOC		
2176	004072	104401	042225			TYPE	,OPR002	
2177	004076	104412				RDOCT		:GET VALUE
2178	004100	012637	001160			MOV	(SP)+,\$TMP0	
2179	004104	001407				BEQ	10\$:CHECK IF <CR>
2180	004106	022737	001000	001160		CMP	#1000,\$TMP0	:CHECK IF LEGAL
2181	004114	101757				BLOS	7\$	
2182	004116	013737	001160	001264		MOV	\$TMP0,\$VECT1	:LOAD NEW VECTOR ADDRESS
2183	004124	104401	042263		10\$:	TYPE	,OPR004	:TYPE 'RK611 PRIORITY () -'
2184	004130	005046				CLR	-(SP)	:MAKE ROOM ON THE STACK
2185	004132	113716	001265			MOVB	\$VECT1+1,(SP)	:GET PRIORITY
2186	004136	006216				ASR	(SP)	:SHIF RIGH 5 BITS
2187	004140	006216				ASR	(SP)	
2188	004142	006216				ASR	(SP)	
2189	004144	006216				ASR	(SP)	
2190	004146	006216				ASR	(SP)	
2191	004150	104402				TYPOC		
2192	004152	104401	042225			TYPE	,OPR002	
2193	004156	104412				RDOCT		:GET VALUE
2194	004160	012637	001160			MOV	(SP)+,\$TMP0	
2195	004164	001430				BEQ	15\$:CHECK FOR DEFAULT
2196	004166	022737	000007	001160		CMP	#7,\$TMP0	:CHECK IF LEGAL
2197	004174	103753				BLO	10\$	
2198	004176	022737	000004	001160		CMP	#4,\$TMP0	
2199	004204	101347				BHI	10\$	
2200	004206	006337	001160			ASL	\$TMP0	:SHIFT 5 BITS LEFT
2201	004212	006337	001160			ASL	\$TMP0	
2202	004216	006337	001160			ASL	\$TMP0	
2203	004222	006337	001160			ASL	\$TMP0	
2204	004226	006337	001160			ASL	\$TMP0	
2205	004232	042737	160000	001264		BIC	#160000,\$VECT1	:GET PRIORITY BITS
2206	004240	153737	001160	001265		BISB	\$TMP0,\$VECT1+1	
2207	004246	013737	001264	003240	15\$:	MOV	\$VECT1,RKVEC	
2208	004254	042737	160000	003240		BIC	#160000,RKVEC	:GET VECTOR
2209	004262	113737	001265	003242		MOVB	\$VECT1+1,RKPRI	:GET PRIORITY
2210	004270	013702	001270			MOV	\$BASE,R2	:SET '611 BASE
2211	004274	005037	001202			CLR	\$ESCAPE	:CLEAR ESCAPE
2212	004300	012746	000000		NEWPAS:	MOV	#PRO,-(SP)	:LOCK OUT INTERRUPTS
2213	004304	012746	004312			MOV	#TST1,-(SP)	
2214	004310	000002				RTI		

2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227 004312 000004
2228 004314 012737 000062 001200
2229 004322 013702 001270
2230 004326 012762 100000 000000
2231 004334 012762 000040 000026
2232 004342 012762 177777 000002
2233 004350 012762 053672 000004
2234 004356 012762 001777 000020
2235 004364 012762 003400 000006
2236 004372 012762 000007 000010
2237 004400 012762 012021 000000
2238
2239 004406 012700 000016
2240 004412 012762 000440 000026
2241 004420 012762 000040 000026
2242 004426 005300
2243 004430 001370
2244 004432 016237 000000 003140
2245 004440 016237 000034 003166
2246 004446 016237 000036 003170
2247 004454 012737 012021 003200
2248 004462 012737 071027 003226
2249 004470 012737 037760 003230
2250 004476 023737 003200 003140
2251 004504 001405
2252 004506 104003
2253 004510 012762 100000 000000
2254 004516 000412
2255
2256 004520 023737 003226 003166
2257 004526 001401
2258 004530 104004
2259 004532 023737 003230 003170
2260 004540 001401
2261 004542 104005

.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 #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!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.
: *

2262
2263
2264
2265
2266
2267
2268
2269
2270

```
2271
2272
2273 004544 000004
2274 004546 012737 000062 001200
2275 004554 013702 001270
2276 004560 012762 100000 000000
2277 004566 012762 000040 000026
2278 004574 012762 177777 000002
2279 004602 012762 053672 000004
2280 004610 012762 001777 000020
2281 004616 012762 003400 000006
2282 004624 012762 000007 000000
2283 004632 012762 012023 000000
2284
2285 004640 012700 000016
2286 004644 012762 000440 000026 1$:
2287 004652 012762 000040 000026
2288 004660 005300
2289 004662 001370
2290 004664 016237 000000 003140
2291 004672 016237 000034 003166
2292 004700 016237 000036 003170
2293 004706 012737 012023 003200
2294 004714 012737 071227 003226
2295 004722 012737 037760 003230
2296 004730 023737 003200 003140
2297 004736 001405
2298 004740 104006
2299 004742 012762 100000 000000
2300 004750 000412
2301
2302 004752 023737 003226 003166 2$:
2303 004760 001401
2304 004762 104007
2305 004764 023737 003230 003170 3$:
2306 004772 001401
2307 004774 104010
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320 004776 000004
2321 005000 012737 000050 001200
2322 005006 013702 001270
2323 005012 012762 100000 000000
2324 005020 012762 000040 000026
2325 005026 012762 177777 000002
2326 005034 012762 053672 000004
```

TST2: 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
MOV #3*4+2,R0 ;CLOCK IN DRIVE MESSAGE
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 #S.SEEK!S.RTC!S.FMT.7000,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
CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
BEQ 3\$;YES, CHECK MESSAGE B
ERROR 7 ;MESS A INCORRECT
CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
BEQ TST3 ;;YES, GO ON TO NEXT TEST
ERROR 10 ;MESS B INCORRECT

TST3: 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

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.

```
2327 005042 012762 001777 000020      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2328 005050 012762 003400 000006      MOV      #3400,RKDA(R2)  ;LOAD TRACK
2329 005056 012762 000007 000010      MOV      #7,RKCS2(R2)   ;LOAD DRIVE NUM.
2330 005064 012762 012031 000000      MOV      #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE WRTCHK WITH CDT SET IN
2331                                     ; 24 SECTOR FORMAT
2332 005072 012700 000016      MOV      #3*4+2,R0      ;CLOCK IN DRIVE MESSAGE
2333 005076 012762 000440 000026 1$:      MOV      #DMD.MCLK,RKMR1(R2)
2334 005104 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2335 005112 005300      DEC      R0
2336 005114 001370      BNE      1$
2337 005116 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2338 005124 016237 000034 003160      MOV      RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2339 005132 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2340 005140 012737 012031 003200      MOV      #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2341 005146 012737 071027 003226      MOV      #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2342 005154 012737 037760 003230      MOV      #37760,E.MR3   ;LOAD EXPECTED MR3
2343 005162 023737 003200 003140      CMP      E.CS1,T.CS1   ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2344 005170 001405      BEQ      2$           ;YES, CHECK MESSAGE A
2345 005172 104011      ERROR   11           ;CS1 INCORRECT
2346 005174 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2347 005202 000412      BR       TST4        ;;GO TO NEXT TEST
2348
2349 005204 023737 003226 003166 2$:      CMP      E.MR2,T.MR2   ;CHECK MESS A CORRECT
2350 005212 001401      BEQ      3$           ;YES, CHECK MESSAGE B
2351 005214 104012      ERROR   12           ;MESS A INCORRECT
2352 005216 023737 003230 003170 3$:      CMP      E.MR3,T.MR3   ;CHECK MESS B CORRECT
2353 005224 001401      BEQ      TST4        ;;YES, GO ON TO NEXT TEST
2354 005226 104013      ERROR   13           ;MESS B INCORRECT
2355
2356                                     ;*****
2357                                     ;*TEST 4      READ DATA CLEAR MESSAGE
2358                                     ;*
2359                                     ;* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE
2360                                     ;* CONTROLLER ON DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
2361                                     ;* RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
2362                                     ;* 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND THE
2363                                     ;* CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT.
2364                                     ;*
2365                                     ;*****
2366 005230 000004      TST4:   SCOPE
2367 005232 012737 000062 001200      MOV      #50,$TIMES    ;;DO 50. ITERATIONS
2368 005240 013702 001270      MOV      $BASE,R2      ;LOAD RK611 BASE
2369 005244 012762 000000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2370 005252 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2371 005260 012762 177777 000002      MOV      #-1,RKWC(R2)  ;LOAD WORD COUNT
2372 005266 012762 053672 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2373 005274 012762 001777 000020      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2374 005302 012762 003400 000006      MOV      #3400,RKDA(R2) ;LOAD TRACK
2375 005310 012762 000007 000010      MOV      #7,RKCS2(R2)  ;LOAD DRIVE NUMBER
2376 005316 012762 012021 000000      MOV      #CDT!CFMT!RDDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2377                                     ; 24 SECTOR FORMAT
2378 005324 012700 000156      MOV      #27.*4+2,R0   ;LOAD COUNT TO LOAD DRIVE CLEAR
2379 005330 012762 000440 000026 1$:      MOV      #DMD.MCLK,RKMR1(R2)
2380 005336 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2381 005344 005300      DEC      R0
2382 005346 001370      BNE      1$
```

```

2383 005350 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2384 005356 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2385 005364 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2386 005372 012737 012021 003200      MOV      #CDT.CFMT!RDATA,E.CS1 ;LOAD EXPECTED CS1
2387 005400 012737 071407 003226      MOV      #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2388 005406 005037 003230      CLR      E.MR3 ;LOAD EXPECTED MAINT REG.
2389 005412 023737 003200 003140      CMP      E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2390 005420 001405      BEQ      2$ ;YES, CHECK MESSB
2391 005422 104014      ERROR    14 ;CS1 INCORRECT
2392 005424 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2393 005432 000412      BR       TST5 ;GO TO NEXT TEST
2394
2395 005434 023737 003226 003166 2$:      CMP      E.MR2,T.MR2 ;CHECK MESS A CORRECT
2396 005442 001401      BEQ      3$ ;YES, CHECK MESS B
2397 005444 104015      ERROR    15 ;MESS A INCORRECT
2398 005446 023737 003230 003170 3$:      CMP      E.MR3,T.MR3 ;CHECK MESS B CORRECT
2399 005454 001401      BEQ      TST5 ;YES, GO ON TO NEXT TEST
2400 005456 104016      ERROR    16 ;MESS B INCORRECT
2401
2402
2403 *****
2404 *TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE
2405 *
2406 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2407 * CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
2408 * ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777,
2409 * TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE AND DRIVE CLEAR.
2410 * CHECK IF PROPER DRIVE CLEAR MESSAGE IS ASSEMBLED.
2411 *****
2412 ST5: SCOPE
2413 MOV #50,$TIMES ;DO 50. ITERATIONS
2414 MOV $BASE,R2 ;LOAD RK611 BASE
2415 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2416 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2417 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2418 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2419 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2420 MOV #3400,RKDA(R2) ;LOAD TRACK
2421 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2422 MOV #CDT!CFMT!WRDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2423 ; 24 SECTOR FORMAT
2424 MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2425 1$: MOV #DMD!MCLK,RKMR1(R2)
2426 MOV #DMD,RKMR1(R2)
2427 DEC R0
2428 BNE 1$
2429 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2430 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2431 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2432 MOV #CDT!CFMT!WRDATA,E.CS1 ;LOAD EXPECTED CS1
2433 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2434 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2435 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2436 BEQ 2$ ;YES, CHECK MESSB
2437 ERROR 17 ;CS1 INCORRECT
2438 MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

```
2439 005662 000412 BR TST6 ;;GO TO NEXT TEST
2440
2441 005664 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2442 005672 001401 BEQ 3$ ;YES, CHECK MESS B
2443 005674 104020 ERROR 20 ;MESS A INCORRECT
2444 005676 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2445 005704 001401 BEQ TST6 ;YES, GO ON TO NEXT TEST
2446 005706 104021 ERROR 21 ;MESS B INCORRECT
2447
2448
2449 *****
2450 *TEST 6 DRIVE CLEAR MESSAGE FOR WRITE CHECK
2451 *
2452 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2453 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
2454 * CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
2455 * CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
2456 * THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR.
2457 * MAKE SURE CORRECT MESSAGE IS ASSEMBLED.
2458 *****
2459 TST6: SCOPE
2460 005710 000004 MOV #50,$TIMES ;;DO 50. ITERATIONS
2461 005712 012737 000062 001200 MOV $BASE,R2 ;LOAD RK611 BASE
2462 005720 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2463 005724 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2464 005732 012762 000040 000026 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2465 005740 012762 177777 000002 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2466 005746 012762 053672 000004 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2467 005754 012762 001777 000020 MOV #3400,RKDA(R2) ;LOAD TRACK
2468 005762 012762 003400 000006 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2469 005776 012762 012031 000000 MOV #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2470 ; 24 SECTOR FORMAT
2471 006004 012700 000156 MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2472 006010 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)
2473 006016 012762 000040 000026 MOV #DMD,RKMR1(R2)
2474 006024 005300 DEC R0
2475 006026 001370 BNE 1$
2476 006030 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2477 006036 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2478 006044 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2479 006052 012737 012031 003200 MOV #CDT!CFMT.WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2480 006060 012737 071407 003226 MOV #S.CLR!S.FMT.70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2481 006066 005037 003230 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2482 006072 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2483 006100 001405 BEQ 2$ ;YES, CHECK MESSB
2484 006102 104022 ERROR 22 ;CS1 INCORRECT
2485 006104 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2486 006112 000412 BR TST7 ;;GO TO NEXT TEST
2487
2488 006114 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2489 006122 001401 BEQ 3$ ;YES, CHECK MESS B
2490 006124 104023 ERROR 23 ;MESS A INCORRECT
2491 006126 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2492 006134 001401 BEQ TST7 ;YES, GO ON TO NEXT TEST
2493 006136 104024 ERROR 24 ;MESS B INCORRECT
2494
```

```
2495 .SBTTL **HEADER GENERATION
2496
2497
2498 .....
2499 *TEST 7          HEADER GENERATION (PART 1)
2500 *
2501 *          CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT THE
2502 *          CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA OF 1
2503 *          WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
2504 *          SECTOR 0.  CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
2505 *          CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
2506 *          TO MAKE THAT THE HEADER IS CORRECT.  REPEAT FOR CYLINDERS
2507 *          1-1777.
2508 .....
2509 TST7:  SCOPE
2510 006140 000004          MOV      #10, $TIMES          ;;DO 10. ITERATIONS
2511 006142 012737 000012 001200  MOV      $BASE, R2          ;LOAD RK611 BASE
2512 006150 013702 001270          CLR      E.DCYL           ;INITIALIZE CYLINDER ADD
2513 006160 005037 003220          CLR      E.DA            ;INITIALIZE TRACK AND SECTOR
2514 006164 012737 000021 003200  MOV      #RDATA, E.CS1     ;INITIALIZE FORMAT
2515 006172 012737 006200 001110  MOV      #1$, $LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
2516                                     ; SUBTEST LOJP
2517
2518 006200          1$:
2519 006200 004737 034402          JSR      PC, CALHDR        ;CALULATE EXPECTED HEADER
2520 006204 012762 100000 000000  MOV      #CCLR, RKCS1(R2)  ;CLEAR RK611
2521 006212 012762 000040 000026  MOV      #DMD, RKMR1(R2)   ;PUT RK611 IM MAINTENANCE MODE
2522 006220 012762 177777 000002  MOV      #-1, RKWC(R2)     ;LOAD WORD COUNT
2523 006226 012762 053672 000004  MOV      #BUFF, RKBA(R2)   ;LOAD DUMMY BUS ADDRESS
2524 006234 013762 003220 000020  MOV      E.DCYL, RKDCYL(R2);LOAD CYLINDER NUMBER
2525 006242 013762 003206 000006  MOV      E.DA, RKDA(R2)    ;LOAD TRACK AND SECTOR
2526 006250 013762 003200 000000  MOV      E.CS1, RKCS1(R2)  ;LOAD COMMAND AND FORMAT
2527 006256 012700 000426          MOV      #69.*4+2, R0     ;LOAD COUNT UNTIL HEADER GENERATION
2528 006262 012762 000440 000026  5$:  MOV      #DMD!MCLK, RKMR1(R2)
2529 006270 012762 000040 000026  MOV      #DMD, RKMR1(R2)
2530 006276 005300          DEC      R0
2531 006300 001370          BNE     5$
2532 006302 016237 000000 003140  MOV      RKCS1(R2), T.CS1   ;STORE COMMAND AND STATUS REG 1
2533 006310 016237 000034 003166  MOV      RKMR2(R2), T.MR2   ;STORE FIRST HEADER WORD
2534 006316 016237 000036 003170  MOV      RKMR3(R2), T.MR3   ;STORE SECOND WORD
2535 006324 023737 003200 003140  CMP      T.CS1, T.CS1      ;CHECK COMMAND AND STATUS REG 1 CORRECT
2536 006332 001405          BEQ     6$
2537 006334 104025          ERROR  25                ;YES, CHECK FIRST LINE OF HEADER
2538 006336 012762 100000 000000  MOV      #CCLR, RKCS1(R2)  ;CLEAR CONTROLLER
2539 006344 000412          BR     15$                ;YES, CHECK SECOND WORD OF HEADER
2540
2541 006346 023737 003226 003166  6$:  CMP      E.MR2, T.MR2      ;CHECK IF FIRST WORD OF HEADER CORRECT
2542 006354 001401          BEQ     7$
2543 006356 104026          ERROR  26                ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
2544 006360 023737 003230 003170  7$:  CMP      E.MR3, T.MR3      ;CHECK IF SECOND WORD OF HEADER CORRECT
2545 006366 001401          BEQ     15$
2546 006370 104027          ERROR  27                ;YES, CHECK IF LOOP ON ERROR
2547 006372 104415          ERROR  27                ;SECOND WORD INCORRECT
2548 006374 005237 003220          SCOPE  15$                ;LOOP ON ERROR
2549 006400 022737 000633 003220  INC      E.DCYL           ;INCREMENT EXPECTED CYLINDER
2550 006406 001002          CMP      #633, E.DCYL     ;CHECK IF VALUE OF CYLINDER OVERFLOW DETECTION
2550                                     BNE     16$                ;NO, CONTINUE
```

```
2551 006410 005237 003220      INC      E.DCYL      ;USE 634
2552 006414 022737 001777 003220 16$:  CMP      #1777,E.DCYL ;CHECK IF FINISHED
2553 006422 103266          BHIS     1$          ;NO, GO TO NEXT CONFIGURATION
2554
2555
2556          ;*****
2557          ;*TEST 10      HEADER GENERATION (PART 2)
2558          ;*
2559          ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2560          ;*      THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
2561          ;*      1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
2562          ;*      0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE.
2563          ;*      CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
2564          ;*      TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR HEADS 1-7.
2565          ;*****
2566 006424 000004          TST10:  SCOPE
2567 006426 012737 000012 001200      MOV      #10,$TIMES ;:DO 10. ITERATIONS
2568 006434 013702 001270          MOV      $BASE,R2   ;:LOAD RK611 BASE
2569 006440 005037 003220          CLR      E.DCYL    ;:INITIALIZE CYLINDER ADD
2570 006444 005037 003206          CLR      E.DA     ;:INITIALIZE TRACK AND SECTOR
2571 006450 012737 000021 003200      MOV      #RDDATA,E.CS1 ;:INITIALIZE FORMAT
2572 006456 012737 006464 001110      MOV      #1$,$LPERR ;:LOAD LOOP ON ERROR LOCATION FOR
2573                                     ; SUBTEST LOOP
2574
2575 006464          1$:
2576 006464 004737 034402          JSR      PC,CALHDR  ;:CALULATE EXPECTED HEADER
2577 006470 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;:CLEAR RK611
2578 006476 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;:PUT RK611 IM MAINTENANCE MODE
2579 006504 012762 177777 000002      MOV      #-1,RKWC(R2) ;:LOAD WORD COUNT
2580 006512 012762 053672 000004      MOV      #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
2581 006520 013762 003220 000020      MOV      E.DCYL,RKDCYL(R2) ;:LOAD CYLINDER NUMBER
2582 006526 013762 003206 000006      MOV      E.DA,RKDA(R2) ;:LOAD TRACK AND SECTOR
2583 006534 013762 003200 000000      MOV      E.CS1,RKCS1(R2) ;:LOAD COMMAND AND FORMAT
2584 006542 012700 000426          MOV      #69.*4+2,R0 ;:LOAD COUNT UNTIL HEADER GENERATION
2585 006546 012762 000440 000026 5$:  MOV      #DMD!MCLK,RKMR1(R2)
2586 006554 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2587 006562 005300          DEC      R0
2588 006564 001370          BNE     5$
2589 006566 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;:STORE COMMAND AND STATUS REG 1
2590 006574 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;:STORE FIRST HEADER WORD
2591 006602 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;:STORE SECOND WORD
2592 006610 023737 003200 003140      CMP      E.CS1,T.CS1 ;:CHECK COMMAND AND STATUS REG 1 CORRECT
2593 006616 001405          BEQ     6$          ;:YES, CHECK FIRST LINE OF HEADER
2594 006620 104030          ERROR  30          ;:CS1 INCORRECT
2595 006622 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;:CLEAR CONTROLLER
2596 006630 000412          BR      15$        ;:YES, CHECK SECOND WORD OF HEADER
2597
2598 006632 023737 003226 003166 6$:  CMP      E.MR2,T.MR2 ;:CHECK IF FIRST WORD OF HEADER CORRECT
2599 006640 001401          BEQ     7$          ;:YES, CHECK IF SECOND WORD OF HEADER CORRECT
2600 006642 104031          ERROR  31          ;:FIRST WORD OF HEADER INCORRECT
2601 006644 023737 003230 003170 7$:  CMP      E.MR3,T.MR3 ;:CHECK IF SECOND WORD OF HEADER CORRECT
2602 006652 001401          BEQ     15$        ;:YES,CHECK IF LOOP ON ERROR
2603 006654 104032          ERROR  32          ;:SECOND WORD INCORRECT
2604 006656 104415          15$:  SCOPE
2605 006660 105237 003207          INCB    E.DA+1     ;:INCREMENT TRACK
2606 006664 122737 000007 003207      CMPB    #7,E.DA+1 ;:CHECK IF FINISHED
```

```
2607 006672 103274      BHIS 1$      ;NO, GO TO NEXT CONFIGURATION
2608
2609
2610      ;*****
2611      ;*TEST 11      HEADER GENERATION (PART 3)
2612      ;*
2613      ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2614      ;*      THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
2615      ;*      ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
2616      ;*      0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
2617      ;*      CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
2618      ;*      TO MAKE SURE HEADER IS CORRECT. REPEAT FOR SECTORS 1-25.
2619      ;*
2620      ;*****
2620 006674 000004      TST11: SCOPE
2621 006676 012737 000012 001200      MOV #10,$TIMES      ;;DO 10. ITERATIONS
2622 006704 013702 001270      MOV $BASE,R2      ;LOAD RK611 BASE
2623 006710 005037 003220      CLR E.DCYL      ;INITIALIZE CYLINDER ADD
2624 006714 005037 003206      CLR E.DA      ;INITIALIZE TRACK AND SECTOR
2625 006720 012737 000021 003200      MOV #RDDATA,E.CS1 ;INITIALIZE FORMAT
2626 006726 012737 006734 001110      MOV #1$,$LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
2627      ; SUBTEST LOOP
2628
2629 006734      1$:
2630 006734 004737 034402      JSR PC,CALHDR      ;CALULATE EXPECTED HEADER
2631 006740 012762 100000 000000      MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2632 006746 012762 000040 000026      MOV #DMD,RKMR1(R2) ;PUT RK611 IM MAINTENANCE MODE
2633 006754 012762 177777 000002      MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2634 006762 012762 053672 000004      MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2635 006770 013762 003220 000020      MOV E.DCYL,RKDCYL(R2) ;LOAD CYLINDER NUMBER
2636 006776 013762 003206 000006      MOV E.DA,RKDA(R2) ;LOAD TRACK AND SECTOR
2637 007004 013762 003200 000000      MOV E.CS1,RKCS1(R2) ;LOAD COMMAND AND FORMAT
2638 007012 012700 000426      MOV #69.*4+2,R0 ;LOAD COUNT UNTIL HEADER GENERATION
2639 007016 012762 000440 000026 5$:
2640 007024 012762 000040 000026      MOV #DMD!MCLK,RKMR1(R2)
2641 007032 005300      DEC R0
2642 007034 001370      BNE 5$
2643 007036 016237 000000 003140      MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
2644 007044 016237 000034 003166      MOV RKMR2(R2),T.MR2 ;STORE FIRST HEADER WORD
2645 007052 016237 000036 003170      MOV RKMR3(R2),T.MR3 ;STORE SECOND WORD
2646 007060 023737 003200 003140      CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2647 007066 001405      BEQ 6$ ;YES, CHECK FIRST LINE OF HEADER
2648 007070 104033      ERROR 33 ;CS1 INCORRECT
2649 007072 012762 100000 000000      MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2650 007100 000412      BR 15$ ;YES, CHECK SECOND WORD OF HEADER
2651
2652 007102 023737 003226 003166 6$:
2653 007110 001401      BEQ 7$ ;CHECK IF FIRST WORD OF HEADER CORRECT
2654 007112 104034      ERROR 34 ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
2655 007114 023737 003230 003170 7$:
2656 007122 001401      CMP E.MR3,T.MR3 ;CHECK IF SECOND WORD OF HEADER CORRECT
2657 007124 104035      BEQ 15$ ;YES,CHECK IF LOOP ON ERROR
2658 007126 104415      ERROR 35 ;SECOND WORD INCORRECT
2659 007130 105237 003206 003206 15$:
2660 007134 122737 000025      INCB E.DA ;INCREMENT SECTOR
2661 007142 103274      CMPB #25,E.DA ;CHECK IF FINISHED
2662      BHIS 1$ ;NO, GO TO NEXT CONFIGURATION
```

```
2663 *****
2664 *TEST 12          HEADER GENERATION (PART 4)
2665 *
2666 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2667 * THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
2668 * ONE WORD FOR AND RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
2669 * HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
2670 * MESSAGES, CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE
2671 * REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR 24
2672 * SECTOR FORMAT.
2673 *
2674 *****
2675 007144 000004 TST12: SCOPE
2676 007146 012737 000062 001200 MOV #50,$TIMES ;:DO 50. ITERATIONS
2677 007154 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
2678 007160 005037 003220 CLR E.DCYL ;:INITIALIZE CYLINDER ADD
2679 007164 005037 003206 CLR E.DA ;:INITIALIZE TRACK AND SECTOR
2680 007170 012737 000021 003200 MOV #RDDATA,E.CS1 ;:INITIALIZE FORMAT
2681 007176 012737 007204 001110 MOV #1$,$LPERR ;:LOAD LOOP ON ERROR LOCATION FOR
2682 ; SUBTEST LOOP
2683
2684 007204 1$:
2685 007204 004737 034402 JSR PC,CALHDR ;:CALULATE EXPECTED HEADER
2686 007210 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
2687 007216 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN MAINTENANCE MODE
2688 007224 012762 177777 000002 MOV #-1,RKWC(R2) ;:LOAD WORD COUNT
2689 007232 012762 053672 000004 MOV #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
2690 007240 013762 003220 000020 MOV E.DCYL,RKDCYL(R2) ;:LOAD CYLINDER NUMBER
2691 007246 013762 003206 000006 MOV E.DA,RKDA(R2) ;:LOAD TRACK AND SECTOR
2692 007254 013762 003200 000000 MOV E.CS1,RKCS1(R2) ;:LOAD COMMAND AND FORMAT
2693 007262 012700 000426 MOV #69.*4+2,R0 ;:LOAD COUNT UNTIL HEADER GENERATION
2694 007266 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
2695 007274 012762 000040 000026 MOV #DMD,RKMR1(R2)
2696 007302 005300 DEC R0
2697 007304 001370 BNE 5$
2698 007306 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;:STORE COMMAND AND STATUS REG 1
2699 007314 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;:STORE FIRST HEADER WORD
2700 007322 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;:STORE SECOND WORD
2701 007330 023737 003200 003140 CMP E.CS1,T.CS1 ;:CHECK COMMAND AND STATUS REG 1 CORRECT
2702 007336 001405 BEQ 6$ ;:YES, CHECK FIRST LINE OF HEADER
2703 007340 104036 ERROR 36 ;:CS1 INCORRECT
2704 007342 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR CONTROLLER
2705 007350 000412 BR 15$ ;:YES, CHECK SECOND WORD OF HEADER
2706
2707 007352 023737 003226 003166 6$: CMP E.MR2,T.MR2 ;:CHECK IF FIRST WORD OF HEADER CORRECT
2708 007360 001401 BEQ 7$ ;:YES, CHECK IF SECOND WORD OF HEADER CORRECT
2709 007362 104037 ERROR 37 ;:FIRST WORD OF HEADER INCORRECT
2710 007364 023737 003230 003170 7$: CMP E.MR3,T.MR3 ;:CHECK IF SECOND WORD OF HEADER CORRECT
2711 007372 001401 BEQ 15$ ;:YES,CHECK IF LOOP ON ERROR
2712 007374 104040 ERROR 40 ;:SECOND WORD INCORRECT
2713 007376 104415 15$: SCOPE1 ;:LOOP ON ERROR
2714 007400 032737 010000 003200 BIT #CFMT,E.CS1 ;:CHECK IF FINISHED
2715 007406 001004 BNE TST13 ;:YES, GO TO NEXT TEST
2716 007410 052737 010000 003200 BIS #CFMT,E.CS1 ;:USE 24 SECTOR FORMAT
2717 007416 000672 BR 1$
2718
```


2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733 007420 000004
2734 007422 012737 000062 001200
2735 007430 013702 001270
2736 007434 012762 100000 000000
2737 007442 012762 000040 000026
2738 007450 012762 177675 000002
2739 007456 012762 051630 000004
2740 007464 012762 000023 000000
2741 007472 012700 004724
2742 007476 012762 000440 000026
2743 007504 012762 000040 000026
2744 007512 005300
2745 007514 001370
2746 007516 016237 000000 003140
2747 007524 016237 000010 003150
2748 007532 016237 000002 003142
2749 007540 016237 000004 003144
2750 007546 016237 000014 003154
2751 007554 012737 000023 003200
2752 007562 012737 000200 003210
2753 007570 012737 177777 003202
2754 007576 012737 052034 003204
2755 007604 005037 003214
2756 007610 023737 003200 003140
2757 007616 001405
2758 007620 104041
2759 007622 012762 100000 000000
2760 007630 000517
2761
2762 007632 023737 003210 003150
2763 007640 001401
2764 007642 104042
2765 007644 023737 003204 003144
2766 007652 001401
2767 007654 104044
2768 007656 023737 003202 003142
2769 007664 001401
2770 007666 104045
2771 007670 02373 003214 003154
2772 007676 001401
2773 007700 104043
2774 007702 012703 051630

```
.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 $BASE, R2 ;:LOAD RK611 BASE  
MOV #CCLR, RKCS1(R2) ;:CLEAR RK611  
MOV #DMD, RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE  
MOV #-67, RKWC(R2) ;:WORD COUNT=67  
MOV #NPRBUF, RKBA(R2) ;:LOAD BUFFER ADDRESS  
MOV #WRDATA, RKCS1(R2) ;:ISSUE WRDATA  
MOV #68.*37., R0 ;:ISSUE ENOUGH CLOCKS FOR 68  
1$: MOV #DMD!MCLK, RKMR1(R2) ; NPR TRANSFERS  
MOV #DMD, RKMR1(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 RKER(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 #CCLR, 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
```



```
2775 007706 012701 000101      MOV      #65.,R1      ;LOAD COUNT FOR SICO
2776 007712 005037 003264      CLR      WRDCNT      ;INITIALIZE WORD COUNT
2777 007716 012737 000300 003210      MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
2778 007724 012337 003222 15$:      MOV      (R3)+,E.DB   ;LOAD EXPECTED DATA
2779 007730 016237 000024 003162      MOV      RKDB(R2),T.DB ;GET DATA READ
2780 007736 012700 000020      MOV      #20,R0      ;SET COUNT TO WAIT FOR SILO
2781 007742 005300 16$:      DEC      R0          ;DEC COUNT
2782 007744 001376      BNE     16$         ;WAIT FOR 0
2783 007746 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE CS1
2784 007754 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;STORE CS2
2785 007762 016237 000014 003154      MOV      RKER(R2),T.ER  ;STORE ERROR REG.
2786 007770 023737 003200 003140      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
2787 007776 001405      BEQ     20$         ;YES, CONTINUE
2788 010000 104046      ERROR   46         ;CS1 INCORRECT
2789 010002 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2790 010010 000427      BR      TST14      ;GO ON TO NEXT TEST
```

```
2791
2792 010012 023737 003210 003150 20$:      CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
2793 010020 001401      BEQ     21$         ;YES, CONTINUE
2794 010022 104047      ERROR   47         ;CS2 INCORRECT
2795 010024 023737 003214 003154 21$:      CMP      E.ER,T.ER    ;CHECK ERROR REG CORRECT
2796 010032 001401      BEQ     22$         ;YES, CONTINUE
2797 010034 104050      ERROR   50         ;ERROR REG INCORRECT
2798 010036 023737 003222 003162 22$:      CMP      E.DB,T.DB    ;CHECK DATA CORRECT
2799 010044 001401      BEQ     25$         ;YES, CONTINUE
2800 010046 104051      ERROR   51         ;DATA INCORRECT
2801 010050 005237 003264 25$:      INC      WRDCNT      ;INCREMENT WORD COUNT
2802 010054 005301      DEC      R1          ;
2803 010056 001003      BNE     26$         ;CHECK IF LAST WORD
2804 010060 012737 000100 003210      MOV      #IR,E.CS2   ;UPDATE EXPECTED CS2
2805 010066 100316 26$:      BPL     15$         ;CHECK IF FINISHED
```

.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.

::*****

```
2825
2826 010070 000004      TST14: SCOPE
2827 010072 012737 000012 001200      MOV      #10.,$TIMES ;DO 10. ITERATIONS
2828 010100 013702 001270      MOV      $BASE,R2    ;LOAD RK611 BASE
2829 010104 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2830 010112 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
```

2831	010120	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT=1
2832	010126	012762	053672	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
2833	010134	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
2834	010142	012700	000426			MOV	#69.*4+2,R0	:ISSUE ENOUGH CLOCKS UNITL READY
2835								:FOR SECTOR PULSE
2836	010146	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2837	010154	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2838	010162	005300				DEC	R0	
2839	010164	001370				BNE	1\$	
2840	010166	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
2841	010174	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2842	010202	012737	000023	003200		MOV	#WRDATA,E.CS1	:STORE EXPECTED CS1
2843	010210	005037	003254			CLR	PR.BIT	:GENERATE SYNCH
2844	010214	005037	003256			CLR	M1.BIT	
2845	010220	005037	003262			CLR	BITCNT	
2846	010224	012700	000377			MOV	#255,R0	
2847	010230	004737	035422		5\$:	JSR	PC,RDBIT	
2848	010234	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2849	010242	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
2850	010250	001112				BNE	63\$:NO, REPORT ERROR
2851	010252	005237	003262			INC	BITCNT	:INCREMENT BIT COUNT
2852	010256	005300				DEC	R0	:CHECK IF SYNCH FINISHED
2853	010260	001363				BNE	5\$	
2854	010262	012737	000001	003254		MOV	#1,PR.BIT	:SIMULATE SYNCH BIT
2855	010270	004737	035422			JSR	PC,RDBIT	
2856	010274	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2857	010302	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
2858	010310	001072				BNE	63\$:NO, REPORT CS1
2859	010312	005237	003262			INC	BITCNT	:INCREMENT BIT COUNT
2860	010316	012703	052040			MOV	#HEAD1,R3	:SIMULATE GOOD HEADER
2861	010322	012701	000003			MOV	#3,R1	
2862	010326	012304			12\$:	MOV	(R3)+,R4	:GET NEXT HEADER WORD
2863	010330	012700	000020			MOV	#16,R0	:LOAD BITS PER WORD
2864	010334	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
2865	010342	006004				ROR	R4	:GET NEXT BIT
2866	010344	103403				BCS	17\$	
2867	010346	005037	003254			CLR	PR.BIT	
2868	010352	000403				BR	18\$	
2869								
2870	010354	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
2871	010362	004737	035422		18\$:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
2872	010366	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
2873	010374	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
2874	010402	001035				BNE	63\$:NO, REPORT ERROR
2875	010404	005237	003262			INC	BITCNT	:INCREMENT BIT COUNT
2876	010410	005300				DEC	R0	:CHECK IF READY FOR NEXT HEADER WORD
2877	010412	001350				BNE	15\$:NO, CONTINUE
2878	010414	005301				DEC	R1	:CHECK IF FINISHED WITH HEAD
2879	010416	001343				BNE	12\$:NO, CONTINUE
2880	010420	012700	000105			MOV	#69,R0	:LOAD COUNT FOR GAP
2881	010424	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	:SIMULATE GAP
2882	010432	005037	003254			CLR	PR.BIT	
2883	010436	004737	035422			JSR	PC,RDBIT	
2884	010442	005300				DEC	R0	
2885	010444	001367				BNE	25\$	
2886	010446	016237	000026	003164		MOV	RKMR1(R2),T.MR1	:GET MR1

```

2887 010454 012737 062040 003224      MOV      #DMD.ECCW.MEWD.WRTGAT,E.MR1 ;LOAD EXPECTED MR
2888 010462 023737 003224 003164      CMP      E.MR1,*.MR1 ;CHECK IF WRITE GATE SET
2889 010470 001411                      BEQ      TST15 ;:YES, GO ON TO NEXT TEST
2890 010472 104053                      ERROR   53
2891 010474 000407                      BR       TST15 ;:GO ON TO NEXT TEST
2892
2893 010476 016237 000010 003150 63$:  MOV      RKCS2(R2),T.CS2 ;STORE CS2 FOR PRINT OUT
2894 010504 016237 000014 003154      MOV      RKER(R2),*.ER ;STORE ERROR RFG FOR PRINT OUT
2895 010512 104052                      ERROR   52 ;REPORT ERROR
2896
2897
2898 *****
2899 *TEST 15 SECTOR PULSE DETECTION FOR WRITE DATA
2900 *
2901 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2902 * CONTROLLER IN MAINTENANCE MODE ISSUE A WRITE DATA OF ONE
2903 * WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
2904 * SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
2905 * SIMULATE AN INDEX PULSE AND A HEADER WITH THE FOLLOWING
2906 * DATA:
2907 *
2908 * 000000
2909 * 140000
2910 * 140000
2911 *
2912 * MAKE SURE WRITE GATE DOES NOT SET.
2913 *****
2914 TST15: SCOPE
2915 010514 000004                      MOV      #10,STIMES ;:DO 10. ITERATIONS
2916 010516 012737 000012 001200      MOV      $BASE,R2 ;LOAD RK611 BASE
2917 010524 013702 001270 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2918 010530 012762 100000 000000      MOV      #700,R0 ;SET STALL COUNT
2919 010542 005300                      4$: DEC   R0 ;DEC COUNT
2920 010544 001376                      BNE     4$ ;LOOP UNTIL ZERO
2921 010546 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2922 010554 012762 177777 000002      MOV      #-1,RKWC(R2) ;WORD COUNT = 1
2923 010562 012762 053672 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2924 010570 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
2925 010576 012700 000426                      MOV      #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTILL READY
2926 * ; FOR SECTOR PULSE
2927 010602 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
2928 010610 012762 000040 000026      MOV      #DMD!FKMR1(R2)
2929 010616 005300                      DEC     R0
2930 010620 001370                      BNE     1$
2931 010622 012700 000004                      MOV      #4,R0 ;SIMULATE INDEX PULSE
2932 010626 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
2933 010634 012762 000640 000026 2$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
2934 010642 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
2935 010650 005300                      DEC     R0
2936 010652 001370                      BNE     2$
2937 010654 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2938 010662 005037 003254                      CLR     PR.BIT ;GENERATE SYNCH
2939 010666 005037 003256                      CLR     M1.BIT
2940 010672 012700 000377                      MOV      #255,R0
2941 010676 004737 035422 5$:  JSR     PC,RDBIT
2942 010702 005300                      DEC     R0
  
```

```

2943 010704 001374          BNE      5$
2944 010706 012737 000001 003254  MOV     #1,PR.BIT      ;SIMULATE SYNCH BIT
2945 010714 004737 035422          JSR     PC,RDBIT
2946 010720 012703 052040          MOV     #HEAD1,R3     ;SIMULATE GOOD HEADER
2947 010724 012701 000003          MOV     #3,R1
2948 010730 012304          12$:   MOV     (R3)+,R4      ;GET NEXT HEADER WORD
2949 010732 012700 000020          MOV     #16,R0        ;LOAD BITS PER WORDS
2950 010736 013737 003254 003256 15$:   MOV     PR.BIT,M1.BIT ;STORE PREVIOUS BIT
2951 010744 006004          ROR     R4             ;GET NEXT BIT
2952 010746 103403          BCS     17$
2953 010750 005037 003254          CLR     PR.BIT
2954 010754 000403          BR      18$
2955
2956 010756 012737 000001 003254 17$:   MOV     #1,PR.BIT
2957 010764 004737 035422 18$:   JSR     PC,RDBIT      ;SIMULATE NEXT BIT
2958 010770 005300          DEC     R0            ;CHECK IF READY FOR NEXT HEADER WORD
2959 010772 001361          BNE     15$          ;NO,CONTINUE
2960 010774 005301          DEC     R1            ;CHECK IF FINISHED WITH HEADER
2961 010776 001354          BNE     12$          ;NO,CONTINUE
2962 011000 012700 000100          MOV     #64,R0        ;LOAD COUNT FOR GAP
2963 011004 013737 003254 003256 25$:   MOV     PR.BIT,M1.BIT ;SIMULATE GAP
2964 011012 005037 003254          CLR     PR.BIT
2965 011016 004737 035422          JSR     PC,RDBIT
2966 011022 005300          DEC     R0
2967 011024 001367          BNE     25$
2968 011026 016237 000026 003164  MOV     RKMRI(R2),T.MR1 ;GET MR1
2969 011034 012737 022040 003224  MOV     #DMD ECCW!MEWD,E.MR1 ;LOAD EXPECTDED MR1
2970 011042 023737 003164 003224  CMP     T.MR1,E.MR1   ;CHECK IF WRITE GATE DID NOT SET
2971 011050 001401          BEQ     TST16         ;;YES, GO ON TO NEXT TEST
2972 011052 104053          ERROR   53
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986

```

```

*****
: *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.
: *
*****

```

```

2987 011054 000004          TST16: SCOPE
2988 011056 012737 000012 001200  MOV     #10,$TIMFS    ;;DO 10. ITERATIONS
2989 011064 013702 001270          MOV     $BASE,R2     ;LOAD RK611 BASE
2990 011070 005037 003300          CLR     CYLN         ;INITIALIZE CYLINDER
2991 011074 005037 003276          CLR     SECTOR       ;INITIALIZE TRACK AND SECTOR
2992 011100 012737 011106 001110  MOV     #1,$SLPLRR   ;LOAD LOOP ON ERROR LOCATION FOR
2993                                     ; SUBTEST LOOP
2994
2995
2996 011106 004737 034220 1$:   JSR     PC,INLHDR    ;GENERATE HEADER WORDS
2997 011112 012762 100000 000000  MOV     #CLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
2998 011120 012762 000040 000026  MOV     #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE

```

```

2999 011126 013762 003300 000020      MOV      CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3000 011134 013762 003276 000006      MOV      SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3001 011142 012762 177777 000002      MOV      #-1,RKWC(R2) ;WORD COUNT=1
3002 011150 012762 053672 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3003 011156 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3004 011164 012700 000426      MOV      #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3005                                     ; FOR SECTOR PULSE
3006 011170 012762 000440 000026 5$:      MOV      #DMD!MCLK,RKMR1(R2)
3007 011176 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3008 011204 005300      DEC      R0
3009 011206 001370      BNE      5$
3010 011210 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3011 011216 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3012 011224 005037 003254      CLR      PR.BIT ;GENERATE SYNCH
3013 011230 005037 003256      CLR      M1.BIT
3014 011234 012700 000377      MOV      #255.,R0
3015 011240 004737 035422 10$:      JSR      PC,RDBIT
3016 011244 005300      DEC      R0
3017 011246 001374      BNE      10$
3018 011250 012737 000001 003254      MOV      #1,PR.BIT ;SIMULATE SYNCH BIT
3019 011256 004737 035422      JSR      PC,RDBIT
3020 011262 012703 003302      MOV      #HEADER,R3 ;SIMULATE HEADER
3021 011266 012701 000003      MOV      #3,R1
3022 011272 012304 12$:      MOV      (R3)+,R4 ;GET NEXT HEADER WORD
3023 011274 012700 000020      MOV      #16.,R0 ;LOAD BITS PER WORD
3024 011300 013737 003254 003256 15$:      MOV      PR.BIT,M1.BIT
3025 011306 006004      ROR      R4
3026 011310 103403      BCS      17$
3027 011312 005037 003254      CLR      PR.BIT
3028 011316 000403      BR       18$
3029
3030 011320 012737 000001 003254 17$:      MOV      #1,PR.BIT
3031 011326 004737 035422 18$:      JSR      PC,RDBIT ;SIMULATE NEXT BIT
3032 011332 005300      DEC      R0 ;CHECK IF READY FOR NEXT HEADER WORD
3033 011334 001361      BNE      15$ ;NO, CONTINUE
3034 011336 005301      DEC      R1 ;CHECK IF FINISHED WITH HEADER
3035 011340 001354      BNE      12$ ;NO, CONTINUE
3036 011342 012700 000100      MOV      #64.,R0 ;LOAD COUNT FOR GAP
3037 011346 013737 003254 003256 25$:      MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3038 011354 005037 003254      CLR      PR.BIT
3039 011360 004737 035422      JSR      PC,RDBIT
3040 011364 005300      DEC      R0 ;CHECK IF GAP FINISHED
3041 011366 001367      BNE      25$ ;NO, CONTINUE
3042 011370 016237 000006 003146      MOV      RKDA(R2),T.DA ;GET DISK ADDRESS REG
3043 011376 016237 000020 003160      MOV      RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3044 011404 023737 003206 003146      CMP      E.DA,T.DA ;CHECK DISK ADD CORRECT
3045 011412 001401      BEQ      30$ ;YES, CONTINUE
3046 011414 104054      ERROR   54 ;DISK ADDRESS INCORRECT
3047 011416 023737 003220 003160 30$:      CMP      E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3048 011424 001401      BEQ      32$ ;YES, CHECK IF LOOP ON ERROR
3049 011426 104002      ERROR   R2 ;CYLINDER INCORRECT
3050 011430 104415 32$:      SCOP1 ;CHECK IF LOOP ON ERROR
3051 011432 105237 003276      INCR    SECTOR ;INCREMENT SECTOR
3052 011436 122737 000026 003276      C*     #26,SECTOR ;CHECK IF ALL SECTOR TRIED
3053 011444 001220      BNE     1$ ;NO, TRY NEXT VALUE
3054

```

3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069 011446 000004
3070 011450 012737 000012 001200
3071 011456 013702 001270
3072 011462 005037 003300
3073 011466 012737 000025 003276
3074 011474 012737 011502 001110
3075
3076
3077 011502
3078 011502 004737 034220
3079 011506 012762 100000 000000
3080 011514 012762 000040 000026
3081 011522 013762 003300 000020
3082 011530 013762 003276 000006
3083 011536 012762 177777 000002
3084 011544 012762 053672 000004
3085 011552 012762 000023 000000
3086 011560 012700 000426
3087
3088 011564 012762 000440 000026
3089 011572 012762 000040 000026
3090 011600 005300
3091 011602 001370
3092 011604 012762 000140 000026
3093 011612 012762 000040 000026
3094 011620 005037 003254
3095 011624 005037 003256
3096 011630 012700 000377
3097 011634 004737 035422
3098 011640 005300
3099 011642 001374
3100 011644 012737 000001 003254
3101 011652 004737 035422
3102 011656 012703 003302
3103 011662 012701 000003
3104 011666 012304
3105 011670 012700 000020
3106 011674 013737 003254 003256
3107 011702 006004
3108 011704 103403
3109 011706 005037 003254
3110 011712 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.  
:*****  
:TEST17: SCOPE  
:MOV #10,$TIMES ;:DO 10. ITERATIONS  
:MOV $BASE,R2 ;:LOAD RK611 BASF  
:CLR CYLN ;:INITIALIZE CYLINDER  
:MOV #25,SECTOR ;:INITIALIZE TRACK AND SECTOR  
:MOV #1$,$LPERR ;:LOAD LOOP ON ERROR LOCATION FOR  
: ; SUBTEST LOOP  
:  
:1$:  
:JSR PC,INCHDR ;:GENERATE HEADER WORDS  
:MOV #CCLR,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$
```

```

3111
3112 011714 012737 000001 003254 17$: MOV #1,DR,BIT
3113 011722 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3114 011726 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3115 011730 001361 BNE 15$ ;NO, CONTINUE
3116 011732 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3117 011734 001354 BNE 12$ ;NO, CONTINUE
3118 011736 012700 000100 MOV #64,,R0 ;LOAD COUNT FOR GAP
3119 011742 013737 003254 003256 25$: MOV PR,BIT,M1,BIT ;SIMULATE GAP
3120 011750 005037 003254 CLR PR,BIT
3121 011754 004737 035422 JSR PC,RDBIT
3122 011760 005300 DEC R0 ;CHECK IF GAP FINISHED
3123 011762 001367 BNE 25$ ;NO, CONTINUE
3124 011764 016237 000006 003146 MOV RKDA(R2),T,DA ;GET DISK ADDRESS REG
3125 011772 016237 000020 003160 MOV RKDCYL(R2),T,DCYL ;GET CYLINDER ADD REG.
3126 012000 023737 003206 003146 CMP E,DA,T,DA ;CHECK DISK ADD CORRECT
3127 012006 001401 BEQ 30$ ;YES, CONTINUE
3128 012010 104056 ERROR 56 ;DISK ADDRESS INCORRECT
3129 012012 023737 003220 003160 30$: CMP E,DCYL,T,DCYL ;CHECK IF CYLINDER ADD CORRECT
3130 012020 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3131 012022 104002 ERROR R2 ;CYLINDER INCORRECT
3132 012024 104415 32$: SCOPI ;CHECK IF LOOP ON ERROR
3133 012026 105237 003277 INCB TRACK ;INCREMENT TRACK
3134 012032 122737 000003 003277 CMPB #3,TRACK ;CHECK IF ALL TRACKS TRIED
3135 012040 001220 BNE 1$ ;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.

```

```

3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150 012042 000004 TST20: SCOPE
3151 012044 012737 000012 001200 MOV #10,,$TIMES ;DO 10. ITERATIONS
3152 012052 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3153 012056 005037 003300 CLR CYLN ;INITIALIZE CYLINDER
3154 012062 012737 001025 003276 MOV #1025,SECTOR ;INITIALIZE TRACK AND SECTOR
3155 012070 012737 012076 001110 MOV #1$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
3156 ; SUBTEST LOOP
3157
3158 012076 1$: JSR PC,INCHDR ;GENERATE HEADER WORDS
3159 012076 004737 034220 MOV #CCLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3160 012102 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3161 012110 012762 000040 000026 MOV CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3162 012116 013762 003300 000020 MOV SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3163 012124 013762 003276 000006 MOV #-1,RKWC(R2) ;WORD COUNT=1
3164 012132 012762 177777 000002 MOV #BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3165 012140 012762 053672 000004 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3166 012146 012762 000023 000000

```

```
3167 012154 012700 000426      MOV      #69,*4+2,R0      ;ISSUE ENOUGH CLOCKS UNTIL READY
3168                                     ; FOR SECTOR PULSE
3169 012160 012762 000440 000026 5$:  MOV      #DMD.MCLK,RKMR1(R2)
3170 012166 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3171 012174 005300      DEC      R0
3172 012176 001370      BNE      5$
3173 012200 012762 000140 000026      MOV      #DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3174 012206 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3175 012214 005037 003254      CLR      PR.BIT          ;GENERATE SYNCH
3176 012220 005037 003256      CLR      M1.BIT
3177 012224 012700 000377      MOV      #255,R0
3178 012230 004737 035422      10$:  JSR      PC,RDBIT
3179 012234 005300      DEC      R0
3180 012236 001374      BNE      10$
3181 012240 012737 000001 003254      MOV      #1,PR.BIT        ;SIMULATE SYNCH BIT
3182 012246 004737 035422      JSR      PC,RDBIT
3183 012252 012703 003302      MOV      #HEADER,R3      ;SIMULATE HEADER
3184 012256 012701 000003      MOV      #3,R1
3185 012262 012304      12$:  MOV      (R3)+,R4          ;GET NEXT HEADER WORD
3186 012264 012700 000020      MOV      #16,R0          ;LOAD BITS PER WORD
3187 012270 013737 003254 003256 15$:  MOV      PR.BIT,M1.BIT
3188 012276 006004      ROR      R4
3189 012300 103403      BCS      17$
3190 012302 005037 003254      CLR      PR.BIT
3191 012306 000403      BR       18$
3192
3193 012310 012737 000001 003254 17$:  MOV      #1,PR.BIT
3194 012316 004737 035422      18$:  JSR      PC,RDBIT          ;SIMULATE NEXT BIT
3195 012322 005300      DEC      R0              ;CHECK IF READY FOR NEXT HEADER WORD
3196 012324 001361      BNE      15$            ;NO, CONTINUE
3197 012326 005301      DEC      R1              ;CHECK IF FINISHED WITH HEADER
3198 012330 001354      BNE      12$            ;NO, CONTINUE
3199 012332 012700 000100      MOV      #64,R0          ;LOAD COUNT FOR GAP
3200 012336 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT    ;SIMULATE GAP
3201 012344 005037 003254      CLR      PR.BIT
3202 012350 004737 035422      JSR      PC,RDBIT
3203 012354 005300      DEC      R0              ;CHECK IF GAP FINISHED
3204 012356 001367      BNE      25$            ;NO, CONTINUE
3205 012360 016237 000006 003146      MOV      RKDA(R2),T.DA    ;GET DISK ADDRESS REG
3206 012366 016237 000020 003160      MOV      RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3207 012374 023737 003206 003146      CMP      E.DA,T.DA        ;CHECK DISK ADD CORRECT
3208 012402 001401      BEQ      30$            ;YES, CONTINUE
3209 012404 104060      ERROR    60              ;DISK ADDRESS INCORRECT
3210 012406 023737 003220 003160 30$:  CMP      E.DCYL,T.DCYL    ;CHECK IF CYLINDER ADD CORRECT
3211 012414 001401      BEQ      32$            ;YES, CHECK IF LOOP ON ERROR
3212 012416 104002      ERROR    R2              ;CYLINDER INCORRECT
3213 012420 104415      32$:  SCOP1                    ;CHECK IF LOOP ON ERROR
3214 012422 005237 003300      INC      CYLN            ;INCREMENT CYLINDER
3215 012426 022737 000633 003300      CMP      #633,CYLN        ;CHECK IF ALL CYLINDER TRIED
3216 012434 001220      BNE      1$              ;NO, TRY NEXT VALUE
3217
3218
3219 .....
3220 *TEST 21      BAD SECTOR ERROR (PART 1)
3221
3222 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
3223 * CONTROLLER IN MAINTENANCE MODE.  ISSUE A WRITE DATA OF
```


3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278

012436 000004
012440 012737 000012 001200
012446 013702 001270
012452 012762 100000 000000
012460 012762 000040 000026
012466 012762 000000 000020
012474 012762 000000 000006
012502 012762 177777 000002
012510 012762 053672 000004
012516 012762 000023 000000
012524 012700 000426
012530 012762 000440 000026
012536 012762 000040 000026
012544 005300
012546 001370
012550 012762 000140 000026
012556 012762 000040 000026
012564 005037 003254
012570 005037 003256
012574 012700 000377
012600 004737 035422
012604 005300
012606 001374
012610 012737 000001 003254
012616 004737 035422
012622 012703 052046
012626 012701 000003
012632 012304
012634 012700 000020
012640 013737 003254 003256
012646 006004
012650 103403
012652 005037 003254
012656 000403
012660 012737 000001 003254
012666 004737 035422
012672 005300
012674 001361
012676 005301
012700 001354
012702 012700 000100

: * 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.
: *
: * *****

```
TST21: SCOPE
MOV #10.,$TIMES ;:DO 10. ITERATIONS
MOV $BASE,R2 ;:LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
MOV #0,RKDCYL(R2) ;:LOAD CYLINDER
MOV #0,RKDA(R2) ;:LOAD TRACK AND SECTOR
MOV #-1,RKWC(R2) ;:WORD COUNT=1
MOV #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
MOV #WRDATA,RKCS1(R2) ;:ISSUE WRITE DATA
MOV #69.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL READY
;: FOR SECTOR PULSE
1$: 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$: JSR PC,RDBIT
DEC R0 ;:CHECK IF SYNCH FINISHED
BNE 5$
MOV #1,PR.BIT ;:SIMULATE SYNCH BIT
JSR PC,RDBIT
MOV #HEAD2,R3 ;:SIMULATE HEADER ERROR
MOV #3,R1
12$: MOV (R3)+,R4 ;:GET NEXT HEADER WORD
MOV #16.,R0 ;:LOAD BITS PER WORD
15$: MOV PR.BIT,M1.BIT ;:STORE PREVIOUS BIT
ROR R4 ;:GET NEXT BIT
BCS 17$
CLR PR.BIT
BR 18$
17$: MOV #1,PR.BIT
18$: JSR PC,RDBIT ;:SIMULATE NEXT BIT
DEC R0 ;:CHECK IF READY FOR NEXT HEADER WORD
BNE 15$ ;:NO, CONTINUE
DEC R1 ;:CHECK IF FINISHED WITH HEAD2
BNE 12$ ;:NO, CONTINUE
MOV #64.,R0 ;:LOAD COUNT FOR GAP
```

```
3279 012706 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3280 012714 005037 003254 CLR PR.BIT
3281 012720 004737 035422 JSR PC,RDBIT
3282 012724 005300 DEC R0 ;CHECK IF GAP FINISHED
3283 012726 001367 BNE 25$ ;NO, CONTINUE
3284 012730 016237 000600 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3285 012736 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3286 012744 016237 000014 003154 MOV RKR(R2),T.ER ;GET ERROR REG
3287 012752 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3288 012760 012737 000300 003210 MOV #IR!GR,E.CS2 ;LOAD EXPECTED CS2
3289 012766 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG.
3290 012774 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3291 013002 001401 BEQ 30$ ;YES, CHECK CS2
3292 013004 104063 ERROR 63 ;CS1 INCORRECT
3293 013006 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3294 013014 001401 BEQ 32$ ;YES, CHECK ERROR REG.
3295 013016 104064 ERROR 64 ;CS2 INCORRECT
3296 013020 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3297 013026 001401 BEQ TST22 ;:YES, GO ON TO NEXT TEST
3298 013030 104065 ERROR 65 ;ERROR REG INCORRECT
3299
```

```
3300 *****
3301 *TEST 22 BAD SECTOR ERROR (PART 2)
3302 *
3303 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3304 * CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3305 * ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
3306 * HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR
3307 * MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH
3308 * THE FOLLOWING DATA:
3309 *
3310 * 000000
3311 * 1000C0
3312 * 100000
3313 *
3314 * MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS
3315 * IS NOT INCREMENTED.
3316 *
3317 *****
```

```
3318 013032 000004 TST22: SCOPE
3319 013034 012737 000012 001200 MOV #10,$TIMES ;:DO 10. ITERATIONS
3320 013042 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3321 013046 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
3322 013054 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3323 013062 012762 000000 000020 MOV #0,RKDCYL(R2) ;LOAD CYLINDER
3324 013070 012762 000000 000006 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
3325 013076 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
3326 013104 012762 053672 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3327 013112 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3328 013120 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3329 ; FOR SECTOR PULSE
3330 013124 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3331 013132 012762 000040 000026 MOV #DMD,RKMR1(R2)
3332 013140 005300 DEC R0
3333 013142 001370 BNE 1$
3334 013144 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
```

```

3335 013152 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3336 013160 005037 003254      CLR      PR.BIT          ;GENERATE SYNCH
3337 013164 005037 003256      CLR      M1.BIT
3338 013170 012700 000377      MOV      #255.,R0
3339 013174 004737 035422      5$:     JSR      PC,RDBIT
3340 013200 005300      DEC      R0              ;CHECK IF SYNCH FINISHED
3341 013202 001374      BNE      5$
3342 013204 012737 000001 003254      MOV      #1,PR.BIT      ;SIMULATE SYNCH BIT
3343 013212 004737 035422      JSR      PC,RDBIT
3344 013216 012703 052054      MOV      #HEAD3,R3      ;SIMULATE HEADER ERROR
3345 013222 012701 000003      MOV      #3,R1
3346 013226 012304      12$:     MOV      (R3)+,R4        ;GET NEXT HEADER WORD
3347 013230 012700 000020      MOV      #16.,R0        ;LOAD BITS PER WORD
3348 013234 013737 003254 003256 15$:     MOV      PR.BIT,M1.BIT   ;STORE PREVIOUS BIT
3349 013242 006004      ROR      R4              ;GET NEXT BIT
3350 013244 103403      BCS      17$
3351 013246 005037 003254      CLR      PR.BIT
3352 013252 000403      BR       18$
3353
3354 013254 012737 000001 003254 17$:     MOV      #1,PR.BIT
3355 013262 004737 035422 18$:     JSR      PC,RDBIT      ;SIMULATE NEXT BIT
3356 013266 005300      DEC      R0              ;CHECK IF READY FOR NEXT HEADER WORD
3357 013270 001361      BNE      15$
3358 013272 005301      DEC      R1              ;CHECK IF FINISHED WITH HEAD3
3359 013274 001354      BNE      12$
3360 013276 012700 000100      MOV      #64.,R0        ;LOAD COUNT FOR CAP
3361 013302 013737 003254 003256 25$:     MOV      PR.BIT,M1.BIT   ;SIMULATE GAP
3362 013310 005037 003254      CLR      PR.BIT
3363 013314 004737 035422      JSR      PC,RDBIT
3364 013320 005300      DEC      R0              ;CHECK IF GAP FINISHED
3365 013322 001367      BNE      25$
3366 013324 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
3367 013332 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
3368 013340 016237 000014 003154      MOV      RKER(R2),T.ER   ;GET ERROR REG
3369 013346 012737 000023 003200      MOV      #WRDATA,E.CS1  ;LOAD EXPECTED CS1
3370 013354 012737 000300 003210      MOV      #IR.OR,E.CS2   ;LOAD EXPECTED CS2
3371 013362 012737 000200 003214      MOV      #BSE,E.ER      ;LOAD EXPECTED ERROR REG.
3372 013370 023737 003200 003140      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
3373 013376 001401      BEQ      30$            ;YES, CHECK CS2
3374 013400 104063      ERROR    63            ;CS1 INCORRECT
3375 013402 023737 003210 003150 30$:     CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
3376 013410 001401      BEQ      32$            ;YES, CHECK ERROR REG.
3377 013412 104064      ERROR    64            ;CS2 INCORRECT
3378 013414 023737 003214 003154 32$:     CMP      E.ER,T.ER      ;CHECK ERROR REG CORRECT
3379 013422 001401      BEQ      TST23          ;YES, GO ON TO NEXT TEST
3380 013424 104065      ERROR    65            ;ERROR REG INCORRECT

```

```

3381
3382 :*****
3383 :*TEST 23      OPERATION INCOMPLETE
3384 :*
3385 :*
3386 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3387 :* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3388 :* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253,
3389 :* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
3390 :* MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH
3390 :* 1 BIT DIFFERENT IN 30 BITS OF OPI DETERMINATION. ALL

```

```
3391      :* SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY
3392      :* OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS
3393      :* THAT SET.
3394      :*
3395      :*****
3396 013426 000004 TST23: SCOPE
3397 013430 012737 000012 001200 MOV #10.,$TIMES ;DO 10. ITERATIONS
3398 013436 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3399 013442 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
3400 013450 012700 002500 MOV #2500,R0 ;SET COUNT FOR STALL
3401 013454 005300 2$: DEC R0 ;DEC COUNT
3402 013456 001376 BNE 2$ ;LOOP UNTIL 0
3403 013460 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3404 013466 012762 053672 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
3405 013474 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT = 1
3406 013502 012762 001253 000020 MOV #1253,RKDCYL(R2) ;LOAD CYLINDER
3407 013510 012762 001023 000006 MOV #1023,RKDA(R2) ;LOAD TRACK AND SECTOR
3408 013516 012762 010023 000000 MOV #WRDATA.CFMT,RKCS1(R2) ;ISSUE WRITE DATA
3409 013524 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3410 ; FOR SECTOR PULSE
3411 013530 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)
3412 013536 012762 000040 000026 MOV #DMD,RKMR1(R2)
3413 013544 005300 DEC R0
3414 013546 001370 BNF 1$
3415 013550 012705 000040 MOV #32.,R5 ;LOAD HEADER COUNT
3416 013554 012703 052250 MOV #OPI1,R3 ;LOAD ADDRESS OF HEADERS
3417 013560 012737 010023 003200 MOV #WRDATA!CFMT,E.CS1 ;LOAD EXPECTED CS1
3418 013566 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3419 013574 005037 003214 CLR E.ER ;LOAD EXPECTED ERROR REG
3420 013600 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
3421 013606 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT
3422 013612 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3423 013620 012762 000040 000026 MOV #DMD,RKMR1(R2)
3424 013626 022737 000037 003310 CMP #31.,HDRCNT ;CHECK IF ALL HEADERS DONE
3425 013634 001460 BEQ 26$ ;YES - SKIP TO ERROR TEST
3426 013636 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3427 013642 005037 003256 CLR M1.BIT
3428 013646 012700 000377 MOV #255.,R0
3429 013652 004737 035422 10$: JSR PC,RDBIT
3430 013656 005300 DEC R0 ;CHECK IF SYNCH FINISHED
3431 013660 001374 BNE 10$
3432 013662 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3433 013670 004737 035422 JSR PC,RDBIT
3434 013674 012701 000003 MOV #3,R1 ;SIMULATE OPI
3435 013700 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3436 013702 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
3437 013706 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3438 013714 006004 ROR R4 ;GET NEXT BIT
3439 013716 103403 BCS 17$
3440 013720 005037 003254 CLR PR.BIT
3441 013724 000403 BR 18$
3442
3443 013726 012737 000001 003254 17$: MOV #1,PR.BIT
3444 013734 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3445 013740 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3446 013742 001361 BNE 15$ ;NO, CONTINUE
```

```

3447 013744 005301      DEC      R1      ;CHECK IF FINISHED WITH HEADER
3448 013746 001354      BNE      12$     ;NO,CONTINUE
3449 013750 012700 000100  MOV      #64.,R0 ;LOAD COUNT FOR GAP
3450 013754 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3451 013762 005037 003254      CLR      PR.BIT
3452 013766 004737 035422      JSR      PC,RDBIT
3453 013772 005300      DEC      R0      ;CHECK IF GAP IS FINISHED
3454 013774 001367      BNE      25$     ;NO, CONTINUE
3455 013776 016237 000000 003140 26$:  MOV      RKCS1(R2),T.CS1 ;GET CS1
3456 014004 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
3457 014012 016237 000014 003154      MOV      RKER(R2),T.ER  ;GET ERROR REG.
3458 014020 023737 003200 003140      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
3459 014026 001401      BEQ      30$     ;YES, CONTINUE
3460 014030 104074      ERROR    74      ;CS1 INCORRECT
3461 014032 023737 003210 003150 30$:  CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
3462 014040 001401      BEQ      31$     ;YES, CONTINUE
3463 014042 104075      ERROR    75      ;CS2 INCORRECT
3464 014044 023737 003214 003154 31$:  CMP      E.ER,T.ER    ;CHECK ERROR REG CORRECT
3465 014052 001401      BEQ      32$     ;YES, CONTINUE
3466 014054 104076      ERROR    76      ;ERROR REG INCORRECT
3467 014056 016237 000026 003164 32$:  MOV      RKMRI(R2),T.MR1 ;GET MR1
3468 014064 023737 003224 003164      CMP      E.MR1,T.MR1   ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
3469 014072 001401      BEQ      34$     ;YES, CONTINUE
3470 014074 104077      ERROR    77      ;MR1 INCORRECT
3471 014076 005237 003310 34$:  INC      HDRCNT      ;INCREMENT HEADER COUNT
3472 014102 022737 000037 003310      CMP      #31.,HDRCNT  ;CHECK IF LAST HEADER
3473 014110 001011      BNE      35$     ;NO, CHECK IF FINSHED
3474 014112 012737 020000 003214      MOV      #OPI,E.ER    ;LOAD ERROR BIT
3475 014120 042737 000001 003200      BIC      #GO,E.CS1   ;ADJUST E.CS1 FOR END OF OP CONTENTS
3476 014126 052737 100200 003200      BIS      #RDY!CERR,E.CS1
3477 014134 005305      35$:  DEC      R5      ;CHECK IF FINISHED
3478 014136 001402      BEQ      37$     ;YES - SKIP
3479 014140 000137 013612      JMP      5$      ;DO NEXT SECTOR
3480 014144 012762 100000 000000 37$:  MOV      #CLR,RKCS1(R2) ;CLEAR CONTROLLER
3481 014152 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
3482 014160 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;CS2
3483 014166 016237 000014 003154      MOV      RKER(R2),T.ER  ;ER
3484 014174 012737 000200 003200      MOV      #RDY,E.CS1   ;SET EXPECTED CS1
3485 014202 023737 003140 003200      CMP      T.CS1,E.CS1  ;CHECK IF CORRECT
3486 014210 001401      BEQ      TST24    ;GO TO NEXT TEST
3487 014212 104153      ERROR    153

```

```

3488
3489
3490 .....
3491 *TEST 24      HEADER VRC
3492
3493 *
3494 *      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
3495 *      CONTROLLER IN MAINTENANCE MODE.  ISSUE A WRITE DATA OF
3496 *      ONE WORD WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1253,
3497 *      HEAD 2, SECTOR 23.  CLOCK IN BOTH SEEK AND DRIVE CLEAR
3498 *      MESSAGES SIMULATE A SECTOR PULSE AND HEADER WITH BIT 0
3499 *      OF THE VRC INCORRECT.  MAKE SURE ONLY HEADER VRC AND
3500 *      CONTROLLER ERROR ARE THE ONLY ERRORS SET.  REPEAT FOR
3501 *      BITS 1-15 OF VRC.
3502 .....
TST24: SCOPE

```



```

3559 014556 005301          DEC      R1          :CHECK IF HEADER FINISHED
3560 014560 001354          BNE      17$         :NO, CONTINUE
3561 014562 012700 000100    MOV      #64.,R0     :SIMULATE GAP
3562 014566 013737 003254 003256 30$:  MOV      PR.BIT,M1.BIT
3563 014574 005037 003254    CLR      PR.BIT
3564 014600 004737 035422    JSR      PC,RDBIT
3565 014604 005300          DEC      R0
3566 014606 001367          BNE      30$
3567 014610 016237 000000 003140  MOV      RKCS1(R2),T.CS1 :STORE CS1
3568 014616 016237 000010 003150  MOV      RKCS2(R2),T.CS2 :STORE CS2
3569 014624 016237 000014 003154  MOV      RKER(R2),T.ER   :STORE ERROR REG
3570 014632 023737 003200 003140  CMP      E.CS1,T.CS1    :CHECK CS1 CORRECT
3571 014640 001401          BEQ      35$         :YES, CHECK CS2
3572 014642 104071          ERROR    71          :CS1 INCORRECT
3573 014644 023737 003210 003150 35$:  CMP      E.CS2,T.CS2    :CHECK CS2 CORRECT
3574 014652 001401          BEQ      37$         :YES, CHECK ERROR REG
3575 014654 104072          ERROR    72          :CS2 INCORRECT
3576 014656 023737 003214 003154 37$:  CMP      E.ER,T.ER     :CHECK ERROR REG CORRECT
3577 014664 001401          BEQ      40$         :YES, CHECK IF LOOP ON ERROR
3578 014666 104073          ERROR    73          :ERROR REG INCORRECT
3579 014670 104415          SCOP1          :CHECK IF LOOP ON ERROR
3580 014672 006337 001170    ASL      $TMP4        :GET NEXT PATTERN
3581 014676 103402          BCS      TST25       :CHECK IF FINISHED
3582 014700 000137 014266    JMP      1$          :NO, CONTINUE
3583
3584 :*****
3585 :*TEST 25          BAD SECTOR ERROR AND HEADER VRC
3586 :*
3587 :*
3588 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3589 :* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3590 :* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300.
3591 :* HEAD 1, SECTOR 17, CLOCK IN BOTH SEEK AND DRIVE CLEAR
3592 :* MESSAGES. SIMULATE THE FOLLOWING HEADER:
3593 :*
3594 :*          000300
3595 :*          040057
3596 :*          040356
3597 :*
3598 :* MAKE SURE ONLY HEADER VRC ERROR SETS.
3599 :*****
3600 014704 000004          TST25:  SCOPE
3601 014706 012737 000012 001200  MOV      #10.,$TIMES   ;;DO 10. ITERATIONS
3602 014714 013702 001270    MOV      $BASE,R2     :LOAD RK611 BASE
3603 014720 012762 100000 000000  MOV      #CCLR,RKCS1(R2) :CLEAR RK611
3604 014726 012762 000040 000026  MOV      #DMD,RKMR1(R2) :PUT RK611 IN DIAGNOSTIC MODE
3605 014734 012762 000300 000020  MOV      #300,RKDCYL(R2) :LOAD CYLINDER
3606 014742 012762 000417 000006  MOV      #417,RKDA(R2)  :LOAD TRACK AND SECTOR
3607 014750 012762 177777 000002  MOV      #-1,RKWC(R2)   :WORD COUNT=1
3608 014756 012762 053672 000004  MOV      #BUFF,RKBA(R2) :LOAD DUMMY BUS ADDRESS
3609 014764 012762 000023 000000  MOV      #WRDATA,RKCS1(R2) :ISSUE WRITE DATA
3610 014772 012700 000426    MOV      #69.*4+2,R0   :ISSUE ENOUGH CLOCKS UNTIL READY
3611 :*          : FOR SECTOR PULSE
3612 014776 012762 000440 000026 1$:  MOV      #DMD.MCLK,RKMR1(R2)
3613 015004 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3614 015012 005300          DEC      R0

```

```
3615 015014 001370 BNE 1$
3616 015016 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3617 015024 012762 000040 000026 MOV #DMD,RKMR1(R2)
3618 015032 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3619 015036 005037 003256 CLR M1.BIT
3620 015042 012700 000377 MOV #255,R0
3621 015046 004737 035422 5$: JSR PC,RDBIT
3622 015052 005300 DEC R0 ;CHECK IF SYNCH FINISHED
3623 015054 001374 BNE 5$
3624 015056 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3625 015064 004737 035422 JSR PC,RDBIT
3626 015070 012703 052062 MOV #HEAD4,R3 ;SIMULATE HEADER ERROR
3627 015074 012701 000003 MOV #3,R1
3628 015100 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3629 015102 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
3630 015106 013737 003254 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3631 015114 006004 ROR R4 ;GET NEXT BIT
3632 015116 103403 BCS 17$
3633 015120 005037 003254 CLR PR.BIT
3634 015124 000403 BR 18$
3635
3636 015126 012737 000001 003254 17$: MOV #1,PR.BIT
3637 015134 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3638 015140 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3639 015142 001361 BNE 15$ ;NO, CONTINUE
3640 015144 005301 DEC R1 ;CHECK IF FINISHED WITH HEAD4
3641 015146 001354 BNE 12$ ;NO, CONTINUE
3642 015150 012700 000100 MOV #64,R0 ;LOAD COUNT FOR GAP
3643 015154 013737 003254 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3644 015162 005037 003254 CLR PR.BIT
3645 015166 004737 035422 JSR PC,RDBIT
3646 015172 005300 DEC R0 ;CHECK IF GAP FINISHED
3647 015174 001367 BNE 25$ ;NO, CONTINUE
3648 015176 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3649 015204 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3650 015212 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
3651 015220 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3652 015226 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3653 015234 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG.
3654 015242 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3655 015250 001401 BEQ 30$ ;YES, CHECK CS2
3656 015252 104066 ERROR 66 ;CS1 INCORRECT
3657 015254 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3658 015262 001401 BEQ 32$ ;YES, CHECK ERROR REG.
3659 015264 104067 ERROR 67 ;CS2 INCORRECT
3660 015266 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3661 015274 001401 BEQ TST26 ;YES, GO ON TO NEXT TEST
3662 015276 104070 ERROR 70 ;ERROR REG INCORRECT
```

```
3663
3664
3665 *****
3666 :TEST 26 GOOD HEADER AND PREVIOUS BSE
3667 :
3668 : CLEAR RK611 CONTROLLER WITH CONTROLLER CLEAR. PUT
3669 : CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF
3670 : ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 100,
: HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
```



```
3727 015546 006004 ROR R4 ;GET NEXT BIT
3728 015550 103403 BCS 17$
3729 015552 005037 003254 CLR PR.BIT
3730 015556 000403 BR 18$
3731
3732 015560 012737 000001 003254 17$: MOV #1,PR.BIT
3733 015566 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3734 015572 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3735 015574 001361 BNE 15$ ;NO, CONTINUE
3736 015576 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3737 015600 001354 BNE 12$ ;NO, CONTINUE
3738 015602 012700 000102 MOV #66.,R0 ;LOAD COUNT FOR GAP
3739 ; PLUS 2 COUNTS FOR WRTGAT TO SET
3740 015606 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3741 015614 005037 003254 CLR PR.BIT
3742 015620 004737 035422 JSR PC,RDBIT
3743 015624 005300 DEC R0 ;CHECK IF GAP IS FINISHED
3744 015626 001367 BNE 25$ ;NO, CONTINUE
3745 015630 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3746 015636 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3747 015644 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
3748 015652 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3749 015660 001401 BEQ 30$ ;YES, CONTINUE
3750 015662 104114 ERROR 114 ;CS1 INCORRECT
3751 015664 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3752 015672 001401 BEQ 31$ ;YES, CONTINUE
3753 015674 104115 ERROR 115 ;CS2 INCORRECT
3754 015676 023737 003214 003154 31$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3755 015704 001401 BEQ 32$ ;YES, CONTINUE
3756 015706 104116 ERROR 116 ;ERROR REG INCORRECT
3757 015710 016237 000026 003164 32$: MOV RKMR1(R2),T.MR1 ;GET MR1
3758 015716 023737 003224 003164 CMP E.MR1,T.MR1 ;CHECK WRITE GATE CORRECT
3759 015724 001401 BEQ 34$ ;YES, CONTINUE
3760 015726 104117 ERROR 117 ;MR1 INCORRECT
3761 015730 005237 003310 INC HDRCNT ;INCREMENT HEADER COUNT
3762 015734 012737 062040 003224 34$: MOV #WRTGAT.DMD.MEWD,ECCW,E.MR1 ;LOAD EXPECTED MR1
3763 015742 005305 DEC R5 ;CHECK IF FINISHED
3764 015744 001402 BEQ TST27 ;:YES, GO ON TO NEXT TEST
3765 015746 000137 015454 JMP 5$
```

```
3766
3767
3768 :*****
3769 :*TEST 27 GOOD HEADER AND PREVIOUS HVRC
3770 :*
3771 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3772 :* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF
3773 :* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 200,
3774 :* HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
3775 :* MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE
3776 :* FOLLOWING 3 WORDS WITH A BAD HEADER VRC:
3777 :*
3778 :* 000200
3779 :* 140000
3780 :* 140000
3781 :*
3782 :* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
: SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE
```

3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838

015752 000004
015754 012737 000012 001200
015762 013702 001270
015766 012762 100000 000000
015774 012762 000040 000026
016002 012762 053672 000004
016010 012762 177777 000002
016016 012762 000001 000006
016024 012762 000200 000020
016032 012762 000023 000000
016040 012700 000426
016044 012762 000440 000026 1\$:
016052 012762 000040 000026
016060 005300
016062 001370
016064 012705 000002
016070 012703 052104
016074 012737 000023 003200
016102 012737 000300 003210
016110 005037 003214
016114 012737 022040 003224
016122 005037 003310
016126 012762 000140 000026 5\$:
016134 012762 000040 000026
016142 005037 003254
016146 005037 003256
016152 012700 000377
016156 004737 035422 10\$:
016162 005300
016164 001374
016166 012737 000001 003254
016174 004737 035422
016200 012701 000003
016204 012304 12\$:
016206 012700 000020
016212 013737 003254 003256 15\$:
016220 006004
016222 103403
016224 005037 003254
016230 000403
016232 012737 000001 003254 17\$:
016240 004737 035422 18\$:
016244 005300
016246 001361

FOLLOWING 3 WORDS:
000200
140001
140201
MAKE SURE WRITE GATE SEES INDICATING THAT HEADER HAS BEEN RECOGNIZED.

TST27: 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 #200, RKDCYL(R2) ;LOAD CYLINDER
MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
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 #2, R5 ;LOAD HEADER COUNT
MOV #HEAD6, 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!ECCW, E.MR1 ;LOAD EXPECTED MR1
CLR HDRCNT ;INITIALIZE HEADER COUNT
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 ;CHECK IF SYNCH FINISHED
BNE 10\$
MOV #1, PR.BIT ;SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #3, R1 ;SIMULATE HEADER
12\$: MOV (R3)+, R4 ;GET NEXT HEADER WORD
MOV #16, R0 ;LOAD BITS PER WORD
15\$: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT
BCS 17\$
CLR PR.BIT
BR 18\$
17\$: MOV #1, PR.BIT
18\$: JSR PC, RDBIT ;SIMULATE NEXT BIT
DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
BNE 15\$;NO. CONTINUE

```

3839 016250 005301      DEC      R1      ;CHECK IF FINISHED WITH HEADER
3840 016252 001354      BNE      12$     ;NO, CONTINUE
3841 016254 012700 000102  MOV      #66.,R0 ;LOAD COUNT FOR GAP
3842                                ;PLUS 2 COUNTS FOR WRTGAT TO SET
3843 016260 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3844 016266 005037 003254      CLR      PR.BIT
3845 016272 004737 035422      JSR      PC,RDBIT
3846 016276 005300      DEC      R0      ;CHECK IF GAP IS FINISHED
3847 016300 001367      BNE      25$     ;NO, CONTINUE
3848 016302 016237 000000 003140  MOV      RKCS1(R2),T.CS1 ;GET CS1
3849 016310 016237 000010 003150  MOV      RKCS2(R2),T.CS2 ;GET CS2
3850 016316 016237 000014 003154  MOV      RKER(R2),T.ER   ;GET ERROR REG
3851 016324 023737 003200 003140  CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
3852 016332 001401      BEQ      30$     ;YES, CONTINUE
3853 016334 104120      ERROR   120      ;CS1 INCORRECT
3854 016336 023737 003210 003150 30$:  CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
3855 016344 001401      BEQ      31$     ;YES, CONTINUE
3856 016346 104121      ERROR   121      ;CS2 INCORRECT
3857 016350 023737 003214 003154 31$:  CMP      E.ER,T.ER     ;CHECK ERROR REG CORRECT
3858 016356 001401      BEQ      32$     ;YES, CONTINUE
3859 016360 104122      ERROR   122      ;ERROR REG INCORRECT
3860 016362 016237 000026 003164 32$:  MOV      RKMR1(R2),T.MR1 ;GET MR1
3861 016370 023737 003224 003164  CMP      E.MR1,T.MR1    ;CHECK WRITE GATE CORRECT
3862 016376 001401      BEQ      34$     ;YES, CONTINUE
3863 016400 104123      ERROR   123      ;MR1 INCORRECT
3864 016402 005237 003310 34$:  INC      HDRCNT      ;INCREMENT HEADER COUNT
3865 016406 012737 062040 003224  MOV      #WRTGAT.DMD.MEWD.ECCW,E.MR1 ;LOAD EXPECTED MR1
3866 016414 005305      DEC      R5      ;CHECK IF FINISHED
3867 016416 001402      BEQ      TST30    ;:YES, GO ON TO NEXT TEST
3868 016420 000137 016126      JMP      5$

```

```

3869
3870 .....
3871 *TEST 30      BAD SECTOR ERROR AND PREVIOUS HVRC
3872
3873
3874 *      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3875 *      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A
3876 *      WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR
3877 *      FORMAT, CYLINDER 400, HEAD 0, SECTOR 1.  CLOCK THROUGH
3878 *      SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
3879 *      AND A HEADER OF THE FOLLOWING 3 WORDS WITH A
3880 *      BAD HEADER VRC:
3881 *
3882 *      000400
3883 *      140000
3884 *      140000
3885 *
3886 *      MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
3887 *      DOES NOT SET.  SIMULATE A SECTOR PULSE AND A
3888 *      HEADER CONSISTING OF THE FOLLOWING 3 WORDS:
3889 *
3890 *      000400
3891 *      040001
3892 *      040401
3893 *
3894 *      MAKE SURE BAD SECTOR ERROR SETS AND HEADER VRC
3895 *      ERROR DOES NOT SET.

```

```
3895  
3896  
3897 016424 000004  
3898 016426 012737 000012 001200  
3899 016434 013702 001270  
3900 016440 012762 100000 000000  
3901 016446 012762 000040 000026  
3902 016454 012762 053672 000004  
3903 016462 012762 177777 000002  
3904 016470 012762 000001 000006  
3905 016476 012762 000400 000020  
3906 016504 012762 000023 000000  
3907 016512 012700 000426  
3908  
3909 016516 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)  
3910 016524 012762 000040 000026 MOV #DMD,RKMR1(R2)  
3911 016532 005300 DEC R0  
3912 016534 001370 BNE 1$  
3913 016536 012705 000002 MOV #2,R5 ;LOAD HEADER COUNT  
3914 016542 012703 052120 MOV #HEAD7,R3 ;LOAD ADDRESS OF HEADERS  
3915 016546 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1  
3916 016554 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2  
3917 016562 005037 003214 CLR E.ER ;LOAD EXPECTED MR1  
3918 016566 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1  
3919 016574 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT  
3920 016600 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE  
3921 016606 012762 000040 000026 MOV #DMD,RKMR1(R2)  
3922 016614 005037 003254 CLR PR.BIT ;GENERATE SYNCH  
3923 016620 005037 003256 CLR M1.BIT  
3924 016624 012700 000377 MOV #255.,R0  
3925 016630 004737 035422 10$: JSR PC,RDBIT  
3926 016634 005300 DEC R0 ;CHECK IF SYNCH FINISHED  
3927 016636 001374 BNE 10$  
3928 016640 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT  
3929 016646 004737 035422 JSR PC,RDBIT  
3930 016652 012701 000003 MOV #3,R1 ;SIMULATE HEADER  
3931 016656 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD  
3932 016660 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD  
3933 016664 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT  
3934 016672 006004 ROR R4 ;GET NEXT BIT  
3935 016674 103403 BCS 17$  
3936 016676 005037 003254 CLR PR.BIT  
3937 016702 000403 BR 18$  
3938  
3939 016704 012737 000001 003254 17$: MOV #1,PR.BIT  
3940 016712 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT  
3941 016716 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD  
3942 016720 001361 BNE 15$ ;NO, CONTINUE  
3943 016722 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER  
3944 016724 001354 BNE 12$ ;NO, CONTINUE  
3945 016726 012700 000102 MOV #66.,R0 ;LOAD COUNT FOR GAP  
3946 ; PLUS 2 COUNTS FOR WRTGAT TO SET  
3947 016732 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP  
3948 016740 005037 003254 CLR PR.BIT  
3949 016744 004737 035422 JSR PC,RDBIT  
3950 016750 005300 DEC R0 ;CHECK IF GAP IS FINISHED
```

```

3951 016752 001367      BNE      25$      :NO, CONTINUE
3952 016754 016237 000000 003140      MOV      RKCS1(R2),T.CS1 :GET CS1
3953 016762 016237 000010 003150      MOV      RKCS2(R2),T.CS2 :GET CS2
3954 016770 016237 000014 003154      MOV      RKER(R2),T.ER   :GET ERROR REG
3955 016776 023737 003200 003140      CMP      E.CS1,T.CS1    :CHECK CS1 CORRECT
3956 017004 001401      BEQ      30$      :YES, CONTINUE
3957 017006 104124      ERROR   124      :CS1 INCORRECT
3958 017010 023737 003210 003150 30$:      CMP      E.CS2,T.CS2    :CHECK CS2 CORRECT
3959 017016 001401      BEQ      31$      :YES, CONTINUE
3960 017020 104125      ERROR   125      :CS2 INCORRECT
3961 017022 023737 003214 003154 31$:      CMP      E.ER,T.ER      :CHECK ERROR REG CORRECT
3962 017030 001401      BEQ      32$      :YES, CONTINUE
3963 017032 104126      ERROR   126      :ERROR REG INCORRECT
3964 017034 016237 000026 003164 32$:      MOV      RKMR1(R2),T.MR1 :GET MR1
3965 017042 023737 003224 003164      CMP      E.MR1,T.MR1    :CHECK WRITE GATE CORRECT
3966 017050 001401      BEQ      34$      :YES, CONTINUE
3967 017052 104127      ERROR   127      :MR1 INCORRECT
3968 017054 005237 003310      INC      HDRCNT        :INCREMENT HEADER COUNT
3969 017060 012737 000200 003214 34$:      MOV      #BSE,E.ER      :LOAD EXPECTER ERROR REG
3970 017066 005305      DEC      R5            :CHECK IF FINISHED
3971 017070 001402      BFO     TST31         :;YES, GO ON TO NEXT TEST
3972 017072 000137 016600      JMP      5$

```

3973

3974 :*****

3975 :*TEST 31 HEADER VRC AND PREVIOUS BSE

3976 :*

3977 :* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.

3978 :* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A

3979 :* WRITE DATA OR ONE WORD TO AN RK06 IN 26 SECTOR

3980 :* FORMAT, CYLINDER 140, HEAD 0, SECTOR 1. CLOCK THROUGH

3981 :* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE

3982 :* AND A HEADER OF THE FOLLOWING 3 WORDS WITH A HEADER

3983 :* VRC ERROR:

3984 :*

3985 :* 000140

3986 :* 040000

3987 :* 040140

3988 :*

3989 :* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE

3990 :* DOES NOT SET. STIMULATE A SECTOR PULSE AND A

3991 :* HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

3992 :*

3993 :* 000140

3994 :* 140001

3995 :* 140101

3996 :*

3997 :* MAKE SURE HEADER VRC ERROR SETS AND BAD SECTOR

3998 :* ERROR DOES NOT SET.

3999 :*

4000 :*****

4001 :TST31: SCOPE

4002 :MOV #10,\$TIMES ;DO 10. ITERATIONS

4003 :MOV \$BASE,R2 ;LOAD RK611 BASE

4004 :MOV #CLR,RKCS1(R2) ;CLEAR RK611

4005 :MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE

4006 :MOV #BJFF,RKBA(R2) ;LOAD DUMMY BJS ADDRESS

```
4007 017134 012762 177777 000002      MOV      #-1,RKWC(R2)      ;WORD COUNT -1
4008 017142 012762 000001 000006      MOV      #1,RKDA(R2)      ;LOAD TRACK AND SECTOR
4009 017150 012762 000140 000020      MOV      #140,RKDCYL(R2)  ;LOAD CYLINDER
4010 017156 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
4011 017164 012700 000426      MOV      #69.*4+2,R0      ;ISSUE ENOUGH CLOCKS UNTIL READY
4012                                     ;      FOR SECTOR PULSE
4013 017170 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
4014 017176 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4015 017204 005300      DEC      R0
4016 017206 001370      BNE     1$
4017 017210 012705 000002      MOV      #2,R5            ;LOAD HEADER COUNT
4018 017214 012703 052134      MOV      #HEAD8,R3        ;LOAD ADDRESS OF HEADERS
4019 017220 012737 000023 003200      MOV      #WRDATA,E.CS1    ;LOAD EXPECTED CS1
4020 017226 012737 000300 003210      MOV      #IR.0R,E.CS2    ;LOAD EXPECTED CS2
4021 017234 005037 003214      CLR      E.ER            ;LOAD EXPECTED MR1
4022 017240 012737 022040 003224      MOV      #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
4023 017246 005037 003310      CLR      HDRCNT          ;INITIALIZE HEADER COUNT
4024 017252 012762 000140 000026 5$:      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4025 017260 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4026 017266 005037 003254      CLR      PR.BIT          ;GENERATE SYNCH
4027 017272 005037 003256      CLR      M1.BIT
4028 017276 012700 000377      MOV      #255.,R0
4029 017302 004737 035422      JSR     PC,RDBIT        10$:
4030 017306 005300      DEC      R0              ;CHECK IF SYNCH FINISHED
4031 017310 001374      BNE     10$
4032 017312 012737 000001 003254      MOV      #1,PR.BIT        ;SIMULATE SYNCH BIT
4033 017320 004737 035422      JSR     PC,RDBIT
4034 017324 012701 000003      MOV      #3,R1            ;SIMULATE HEADER
4035 017330 012304 000000 12$:      MOV      (R3)+,R4        ;GET NEXT HEADER WORD
4036 017332 012700 000020      MOV      #16.,R0         ;LOAD BITS PER WORD
4037 017336 013737 003254 003256 15$:      MOV      PR.BIT,M1.BIT    ;STORE PREVIOUS BIT
4038 017344 006004      ROR     R4              ;GET NEXT BIT
4039 017346 103403      BCS     17$
4040 017350 005037 003254      CLR      PR.BIT
4041 017354 000403      BR     18$
4042
4043 017356 012737 000001 003254 17$:      MOV      #1,PR.BIT
4044 017364 004737 035422 18$:      JSR     PC,RDBIT        ;SIMULATE NEXT BIT
4045 017370 005300      DEC      R0              ;CHECK IF READY FOR NEXT HEADER WORD
4046 017372 001361      BNE     15$              ;NO, CONTINUE
4047 017374 005301      DEC      R1              ;CHECK IF FINISHED WITH HEADER
4048 017376 001354      BNE     12$              ;NO, CONTINUE
4049 017400 012700 000102      MOV      #66.,R0         ;LOAD COUNT FOR GAP
4050                                     ;      PLUS 2 COUNTS FOR WRTGAT TO SET
4051 017404 013737 003254 003256 25$:      MOV      PR.BIT,M1.BIT    ;SIMULATE GAP
4052 017412 005037 003254      CLR      PR.BIT
4053 017416 004737 035422      JSR     PC,RDBIT
4054 017422 005300      DEC      R0              ;CHECK IF GAP IS FINISHED
4055 017424 001367      BNE     25$              ;NO, CONTINUE
4056 017426 016237 000000 003140      MOV      RKCS1(R2),T.CS1  ;GET CS1
4057 017434 016237 000010 003150      MOV      RKCS2(R2),T.CS2  ;GET CS2
4058 017442 016237 000014 003154      MOV      RKER(R2),T.ER    ;GET ERROR REG
4059 017450 023737 003200 003140      CMP      E.CS1,T.CS1      ;CHECK CS1 CORRECT
4060 017456 001401      BEQ     30$              ;YES, CONTINUE
4061 017460 104130      ERROR   130              ;CS1 INCORRECT
4062 017462 023737 003210 003150 30$:      CMP      E.CS2,T.CS2      ;CHECK CS2 CORRECT
```

4063	017470	001401				BEQ	31\$:YES, CONTINUE
4064	017472	104131				ERROR	131	:CS2 INCORRECT
4065	017474	023737	003214	003154	31\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
4066	017502	001401				BEQ	32\$:YES, CONTINUE
4067	017504	104132				ERROR	132	:ERROR REG INCORRECT
4068	017506	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	:GET MR1
4069	017514	023737	003224	003164		CMP	E.MR1,T.MR1	:CHECK WRITE GATE CORRECT
4070	017522	001401				BEQ	34\$:YES, CONTINUE
4071	017524	104133				ERROR	133	:MR1 INCORRECT
4072	017526	005237	003310		34\$:	INC	HDRCNT	:INCREMENT HEADER COUNT
4073	017532	012737	000400	003214		MOV	#HVRC,E.ER	:LOAD EXPECTER ERROR REG
4074	017540	005305				DEC	R5	:CHECK IF FINISHED
4075	017542	001402				BEQ	TST32	:YES, GO ON TO NEXT TEST
4076	017544	000137	017252			JMP	5\$	

4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102

```

:*****
: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.
:*****

```

4103	017550	000004				TST32:	SCOPE	
4104	017552	012737	000012	001200		MOV	#10,\$TIMES	:DO 10. ITERATIONS
4105	017560	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
4106	017564	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4107	017572	012700	002500			MOV	#2500,R0	:SET COUNT FOR STALL
4108	017576	005300			2\$:	DEC	R0	:DEC COUNT
4109	017600	001376				BNE	2\$:LOOP UNTIL 0
4110	017602	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4111	017610	012762	053672	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUFFER ADDRESS
4112	017616	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT = 1
4113	017624	012762	000240	000020		MOV	#240,RKDCYL(R2)	:LOAD CYLINDER
4114	017632	012762	000001	000006		MOV	#1,RKDA(R2)	:LOAD TRACK AND SECTOR
4115	017640	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4116	017646	012700	000426			MOV	#69.*4+2,R0	:ISSUE ENOUGH CLODKS UNTIL READY
4117								:FOR SECTOR PULSE
4118	017652	012762	000440	000026	1\$:	MOV	#DMD.MCLK,RKMR1(R2)	

4119	017660	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4120	017666	005300				DEC	R0	
4121	017670	001370				BNE	1\$	
4122	017672	012705	000040			MOV	#32.,R5	;LOAD HEADER COUNT
4123	017676	012703	052550			MOV	#OPI2,R3	;LOAD ADDRESS OF HEADERS
4124	017702	012737	000023	003200		MOV	#WRDATA,E.CS1	;LOAD EXPECTED CS1
4125	017710	012737	000300	003210		MOV	#IR.0R,E.CS2	;LOAD EXPECTED CS2
4126	017716	005037	003214			CLR	E.ER	;LOAD EXPECTED ERROR PEG
4127	017722	012737	022040	003224		MOV	#DMD!MEWD.ECCW,E.MR1	;LOAD EXPECTED MR1
4128	017730	005037	003310			CLR	HDRCNT	;INITIALIZE HEADER COUNT
4129	017734	012762	000140	000026	5\$:	MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
4130	017742	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4131	017750	022737	000037	003310		CMP	#31.,HDRCNT	;CHECK IF ALL HEADERS DONE
4132	017756	001460				BEQ	26\$;YES - SKIP TO ERROR TEST
4133	017760	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
4134	017764	005037	003256			CLR	M1.BIT	
4135	017770	012700	000377			MOV	#255.,R0	
4136	017774	004737	035422		10\$:	JSR	PC,RDBIT	
4137	020000	005300				DEC	R0	;CHECK IF SYNCH FINISHED
4138	020002	001374				BNE	10\$	
4139	020004	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
4140	020012	004737	035422			JSR	PC,RDBIT	
4141	020016	012701	000003			MOV	#3,R1	;SIMULATE OPI
4142	020022	012304			12\$:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
4143	020024	012700	000020			MOV	#16.,R0	;LOAD BITS PER WORD
4144	020030	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
4145	020036	006004				ROR	R4	;GET NEXT BIT
4146	020040	103403				BCS	17\$	
4147	020042	005037	003254			CLR	PR.BIT	
4148	020046	000403				BR	18\$	
4149								
4150	020050	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
4151	020056	004737	035422		18\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
4152	020062	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
4153	020064	001361				BNE	15\$;NO, CONTINUE
4154	020066	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
4155	020070	001354				BNE	12\$;NO,CONTINUE
4156	020072	012700	000100			MOV	#64.,R0	;LOAD COUNT FOR GAP
4157	020076	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
4158	020104	005037	003254			CLR	PR.BIT	
4159	020110	004737	035422			JSR	PC,RDBIT	
4160	020114	005300				DEC	R0	;CHECK IF GAP IS FINISHED
4161	020116	001367				BNE	25\$;NO, CONTINUE
4162	020120	016237	000000	003140	26\$:	MOV	RKCS1(R2),T.CS1	;GET CS1
4163	020126	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
4164	020134	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ERROR REG.
4165	020142	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4166	020150	001401				BEQ	30\$;YES, CONTINUE
4167	020152	104100				ERROR	100	;CS1 INCORRECT
4168	020154	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4169	020162	001401				BEQ	31\$;YES, CONTINUE
4170	020164	104101				ERROR	101	;CS2 INCORRECT
4171	020166	023737	003214	003154	31\$:	CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
4172	020174	001401				BEQ	32\$;YES, CONTINUE
4173	020176	104102				ERROR	102	;ERROR REG INCORRECT
4174	020200	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	;GET MR1

```

4175 020206 023737 003224 003164      CMP      E.MR1,T.MR1      ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4176 020214 001401                      BEQ      34$              ;YES, CONTINUE
4177 020216 104103                      ERROR   103              ;MR1 INCORRECT
4178 020220 005237 003310 34$:      INC      HDRCNT          ;INCREMENT HEADER COUNT
4179 020224 022737 000037 003310      CMP      #31.,HDRCNT     ;CHECK IF LAST HEADER
4180 020232 001011                      BNE     35$              ;NO, CHECK IF FINSHED
4181 020234 012737 020400 003214      MOV     #OPI.HVRC,E.ER   ;LOAD ERROR BIT
4182 020242 042737 000001 003200      BIC     #GO,E.CS1        ;ADJUST E.CS1 FOR END OF OP CONTENTS
4183 020250 052737 100200 003200      BIS     #RDY.CERR,E.CS1
4184 020256 005305 35$:      DEC     R5                ;CHECK IF FINISHED
4185 020260 001402                      BEQ     37$              ;YES - SKIP
4186 020262 000137 017734                      JMP     5$                ;DO NEXT SECTOR
4187 020266 012762 100000 000000 37$:      MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4188 020274 016237 000000 003140      MOV     RKCS1(R2),T.CS1 ;GET CS1
4189 020302 016237 000010 003150      MOV     RKCS2(R2),T.CS2 ;CS2
4190 020310 016237 000014 003154      MOV     RKER(R2),T.ER    ;ER
4191 020316 012737 000200 003200      MOV     #RDY,E.CS1      ;SET EXPECTED CS1
4192 020324 023737 003140 003200      CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
4193 020332 001401                      BEQ     TST33            ;GO TO NEXT TEST
4194 020334 104153                      ERROR   153

```

4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228

```

:*****
:*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.
:*****

```

4229 020336 000004
4230 020340 012737 000012 001200

```

TST33: SCOPE
MOV     #10.,$TIMES      ;DO 10. ITERATIONS

```

4231	020346	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
4232	020352	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4233	020360	012700	002500			MOV	#2500,RC	:SET COUNT FOR STALL
4234	020364	005300			2\$:	DEC	R0	:DEC COUNT
4235	020366	001376				BNE	2\$:LOOP UNTIL 0
4236	020370	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
4237	020376	012762	053672	000004		MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUFFER ADDRESS
4238	020404	012762	177777	000002		MOV	#-1,RKWC(R2)	:WORD COUNT = 1
4239	020412	012762	000300	000020		MOV	#300,RKDCYL(R2)	:LOAD CYLINDER
4240	020420	012762	000001	000006		MOV	#1,RKDA(R2)	:LOAD TRACK AND SECTOR
4241	020426	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
4242	020434	012700	000426			MOV	#69.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL READY
4243								:FOR SECTOR PULSE
4244	020440	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4245	020446	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4246	020454	005300				DEC	R0	
4247	020456	001370				BNE	1\$	
4248	020460	012705	000040			MOV	#32.,R5	:LOAD HEADER COUNT
4249	020464	012703	053050			MOV	#OPI3,R3	:LOAD ADDRESS OF HEADERS
4250	020470	012737	000023	003200		MOV	#WRDATA,E.CS1	:LOAD EXPECTED CS1
4251	020476	012737	000300	003210		MOV	#IR.0R,E.CS2	:LOAD EXPECTED CS2
4252	020504	005037	003214			CLR	E.ER	:LOAD EXPECTED ERROR REG
4253	020510	012737	022040	003224		MOV	#DMD!MEWD!ECCW,E.MR1	:LOAD EXPECTED MR1
4254	020516	005037	003310			CLR	HDRCNT	:INITIALIZE HEADER COUNT
4255	020522	012762	000140	000026	5\$:	MOV	#DMD.MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
4256	020530	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4257	020536	022737	000037	003310		CMP	#31.,HDRCNT	:CHECK IF ALL HEADERS DONE
4258	020544	001460				BEQ	26\$:YES - SKIP TO ERROR TEST
4259	020546	005037	003254			CLR	PR.BIT	:GENERATE SYNCH
4260	020552	005037	003256			CLR	M1.BIT	
4261	020556	012700	000377			MOV	#255.,R0	
4262	020562	004737	035422		10\$:	JSR	PC,RDBIT	
4263	020566	005300				DEC	R0	:CHECK IF SYNCH FINISHED
4264	020570	001374				BNE	10\$	
4265	020572	012737	000001	003254		MOV	#1,PR.BIT	:SIMULATE SYNCH BIT
4266	020600	004737	035422			JSR	PC,RDBIT	
4267	020604	012701	000003			MOV	#3,R1	:SIMULATE OPI
4268	020610	012304			12\$:	MOV	(R3)+,R4	:GET NEXT HEADER WORD
4269	020612	012700	000020			MOV	#16.,R0	:LOAD BITS PER WORD
4270	020616	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4271	020624	006004				ROR	R4	:GET NEXT BIT
4272	020626	103403				BCS	17\$	
4273	020630	005037	003254			CLR	PR.BIT	
4274	020634	000403				BR	18\$	
4275								
4276	020636	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
4277	020644	004737	035422		18\$:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
4278	020650	005300				DEC	R0	:CHECK IF READY FOR NEXT HEADER WORD
4279	020652	001361				BNE	15\$:NO, CONTINUE
4280	020654	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER
4281	020656	001354				BNE	12\$:NO, CONTINUE
4282	020660	012700	000100			MOV	#64.,R0	:LOAD COUNT FOR GAP
4283	020664	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	:SIMULATE GAP
4284	020672	005037	003254			CLR	PR.BIT	
4285	020676	004737	035422			JSR	PC,RDBIT	
4286	020702	005300				DEC	R0	:CHECK IF GAP IS FINISHED

4287	020704	001367				BNE	25\$:NO, CONTINUE
4288	020706	016237	000000	003140	26\$:	MOV	RKCS1(R2),T.CS1	:GET CS1
4289	020714	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:GET CS2
4290	020722	016237	000014	003154		MOV	RKER(R2),T.ER	:GET ERROR REG.
4291	020730	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4292	020736	001401				BEQ	30\$:YES, CONTINUE
4293	020740	104104				ERROR	104	:CS1 INCORRECT
4294	020742	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4295	020750	001401				BEQ	31\$:YES, CONTINUE
4296	020752	104105				ERROR	105	:CS2 INCORRECT
4297	020754	023737	003214	003154	31\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
4298	020762	001401				BEQ	32\$:YES, CONTINUE
4299	020764	104106				ERROR	106	:ERROR REG INCORRECT
4300	020766	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	:GET MR1
4301	020774	023737	003224	003164		CMP	E.MR1,T.MR1	:CHECK TO MAKE SURE WRITE GATE DID NOT SET
4302	021002	001401				BEQ	34\$:YES, CONTINUE
4303	021004	104107				ERROR	107	:MR1 INCORRECT
4304	021006	005237	003310		34:	INC	HDRCNT	:INCREMENT HEADER COUNT
4305	021012	022737	000037	003310		CMP	#31.,HDRCNT	:CHECK IF LAST HEADER
4306	021020	001011				BNE		:NO, CHECK IF FINSHED
4307	021022	012737	020400	003214		MOV	#OPI!HVRC,E.ER	:LOAD ERROR BIT
4308	021030	042737	000001	003200		BIC	#GO,E.CS1	:ADJUST E.CS1 FOR END OF OP CONTENTS
4309	021036	052737	100200	003200		BIS	#RDY!CERR,E.CS1	
4310	021044	005305			35\$:	DEC	R5	:CHECK IF FINISHED
4311	021046	001402				BEQ	37\$:YES - SKIP
4312	021050	000137	020522			JMP	5\$:DO NEXT SECTOR
4313	021054	012762	100000	000000	37\$:	MOV	#CLR,RKCS1(R2)	:CLEAR CONTROLLER
4314	021062	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
4315	021070	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:CS2
4316	021076	016237	000014	003154		MOV	RKER(R2),T.ER	:ER
4317	021104	012737	070200	003200		MOV	#RDY,E.CS1	:SET EXPECTED CS1
4318	021112	023737	003140	003200		CMP	T.CS1,E.CS1	:CHECK IF CORRECT
4319	021120	001401				BEQ	TST34	:GO TO NEXT TEST
4320	021122	104153				ERROR	153	

4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342

```

:*****
:*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

```

```
4343          140040
.344          :
4345          :
4346          :
4347          :
4348 021124 000004          :
4349 021126 012737 000012 001200          :
4350 021134 013702 001270          :
4351 021140 012762 100000 000000          :
4352 021146 012700 002500          :
4353 021152 005300          2$: DEC R0          ;DEC COUNT
4354 021154 001376          BNE 2$          ;LOOP UNTIL 0
4355 021156 012762 000040 000026          MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4356 021164 012762 053672 000004          MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
4357 021172 012762 177777 000002          MOV #-1,RKWC(R2) ;WORD COUNT = 1
4358 021200 012762 000040 000020          MOV #40,RKDCYL(R2) ;LOAD CYLINDER
4359 021206 012762 000001 000006          MOV #1,RKDA(R2) ;LOAD TRACK AND SECTOR
4360 021214 012762 000023 000000          MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
4361 021222 012700 000426          MOV #69.*4+2,R0 ;ISSUE ENOUGH CLODKS UNTIL READY
4362          ; FOR SECTOR PULSE
4363 021226 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4364 021234 012762 000040 000026          MOV #DMD,RKMR1(R2)
4365 021242 005300          DEC R0
4366 021244 001370          BNF 1$
4367 021246 012705 000040          MOV #32.,R5 ;LOAD HEADER COUNT
4368 021252 012703 053350          MOV #OPI4,R3 ;LOAD ADDRESS OF HEADERS
4369 021256 012737 000023 003200          MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
4370 021264 012737 000300 003210          MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4371 021272 005037 003214          CLR E.ER ;LOAD EXPECTED ERROR REG
4372 021276 012737 022040 003224          MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
4373 021304 005037 003310          CLR HDRCNT ;INITIALIZE HEADER COUNT
4374 021310 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4375 021316 012762 000040 000026          MOV #DMD,RKMR1(R2)
4376 021324 022737 000037 003310          CMP #31.,HDRCNT ;CHECK IF ALL HEADERS DONE
4377 021332 001460          BEQ 26$ ;YES - SKIP TO ERROR TEST
4378 021334 005037 003254          CLR PR.BIT ;GENERATE SYNCH
4379 021340 005037 003256          CLR M1.BIT
4380 021344 012700 000377          MOV #255.,R0
4381 021350 004737 035422          10$: JSR PC,RDBIT
4382 021354 005300          DEC R0 ;CHECK IF SYNCH FINISHED
4383 021356 001374          BNE 10$
4384 021360 012737 000001 003254          MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4385 021366 004737 035422          JSR PC,RDBIT
4386 021372 012701 000003          MOV #3,R1 ;SIMULATE OPI
4387 021376 012304          12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4388 021400 012700 000020          MOV #16.,R0 ;LOAD BITS PER WORD
4389 021404 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4390 021412 006004          ROR R4 ;GET NEXT BIT
4391 021414 103403          BCS 17$
4392 021416 005037 003254          CLR PR.BIT
4393 021422 000403          BR 18$
4394
4395 021424 012737 000001 003254 17$: MOV #1,PR.BIT
4396 021432 004737 035422 18$: JSR PC,RDBIT
4397 021436 005300          DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
4398 021440 001361          BNE 15$ ;NO, CONTINUE
```

```
4399 021442 005301          DEC      R1          ;CHECK IF FINISHED WITH HEADER
4400 021444 001354          BNE      12$         ;NO, CONTINUE
4401 021446 012700 000100    MOV      #64.,R0     ;LOAD COUNT FOR GAP
4402 021452 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT ;SIMULATE GAP
4403 021460 005037 003254    CLR      PR.BIT
4404 021464 004737 035422    JSR      PC,RDBIT
4405 021470 005300          DEC      R0          ;CHECK IF GAP IS FINISHED
4406 021472 001367          BNE      25$         ;NO, CONTINUE
4407 021474 016237 000000 003140 26$:  MOV      RKCS1(R2),T.CS1 ;GET CS1
4408 021502 016237 000010 003150    MOV      RKCS2(R2),T.CS2 ;GET CS2
4409 021510 016237 000014 003154    MOV      RKER(R2),T.ER  ;GET ERROR REG.
4410 021516 023737 003200 003140    CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4411 021524 001401          BEQ      30$         ;YES, CONTINUE
4412 021526 104110          ERROR    110         ;CS1 INCORRECT
4413 021530 023737 003210 003150 30$:  CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4414 021536 001401          BEQ      31$         ;YES, CONTINUE
4415 021540 104111          ERROR    111         ;CS2 INCORRECT
4416 021542 023737 003214 003154 31$:  CMP      E.ER,T.ER     ;CHECK ERROR REG CORRECT
4417 021550 001401          BEQ      32$         ;YES, CONTINUE
4418 021552 104112          ERROR    112         ;ERROR REG INCORRECT
4419 021554 016237 000026 003164 32$:  MOV      RKMRI(R2),T.MRI ;GET MRI
4420 021562 023737 003224 003164    CMP      E.MRI,T.MRI   ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4421 021570 001401          BEQ      34$         ;YES, CONTINUE
4422 021572 104113          ERROR    113         ;MRI INCORRECT
4423 021574 005237 003310 34$:  INC      HDRCNT       ;INCREMENT HEADER COUNT
4424 021600 022737 000037 003310    CMP      #31.,HDRCNT   ;CHECK IF LAST HEADER
4425 021606 001011          BNE      35$         ;NO, CHECK IF FINSHED
4426 021610 012737 020400 003214    MOV      #OPI!HVRC,E.ER ;LOAD ERROR BIT
4427 021616 042737 000001 003200    BIC      #GO,E.CS1     ;ADJUST E.CS1 FOR END OF OP CONTENTS
4428 021624 052737 100200 003200    BIS      #RDY!CERR,E.CS1
4429 021632 005305          DEC      R5          ;CHECK IF FINISHED
4430 021634 001402          BEQ      37$         ;YES - SKIP
4431 021636 000137 021310    JMP      5$          ;DO NEXT SECTOR
4432 021642 012762 100000 000000 37$:  MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4433 021650 016237 000000 003140    MOV      RKCS1(R2),T.CS1 ;GET CS1
4434 021656 016237 000010 003150    MOV      RKCS2(R2),T.CS2 ;CS2
4435 021664 016237 000014 003154    MOV      RKER(R2),T.ER  ;ER
4436 021672 012737 000200 003200    MOV      #RDY,E.CS1    ;SET EXPECTED CS1
4437 021700 023737 003140 003200    CMP      T.CS1,E.CS1   ;CHECK IF CORRECT
4438 021706 001401          BEQ      TST35       ;GO TO NEXT TEST
4439 021710 104153          ERROR    153
```

4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454

```
*****  
*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.  
*****
```

```
4455 021712 000004          TST35: SCOPE
4456 021714 012737 000010 001200 MOV #10,$*TIMES ;:DO 10 ITERATIONS
4457 021722 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
4458 021726 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
4459 021734 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
4460 021742 012762 053672 000004 MOV #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
4461 021750 012762 177777 000002 MOV #-1,RKWC(R2) ;:WORD COUNT=1
4462 021756 012762 000000 000020 MOV #0,RKDCYL(R2) ;:LOAD CYLINDER 11DMS
4463 021764 012762 000000 000006 MOV #0,RKDA(R2) ;:LOAD TRACK AND SECTOR
4464 021772 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;:ISSUE COMMAND
4465 022000 012700 000426 MOV #69.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL
4466 ;: READY FOR SECTOR PULSE.
4467 022004 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4468 022012 012762 000040 000026 MOV #DMD,RKMR1(R2)
4469 022020 005300 DEC R0
4470 022022 001370 BNE 1$
4471 022024 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
4472 022032 012762 000040 000026 MOV #DMD,RKMR1(R2)
4473 022040 005037 003254 CLR PR.BIT ;:GENERATE SYNCH
4474 022044 005037 003256 CLR M1.BIT
4475 022050 012700 000377 MOV #255.,R0
4476 022054 004737 035422 5$: JSR PC,RDBIT
4477 022060 005300 DEC R0 ;:CHECK IF SYNCH FINISHED
4478 022062 001374 BNF 5$
4479 022064 012737 000001 003254 MOV #1,PR.BIT ;:SIMULATE SYNCH
4480 022072 004737 035422 JSR PC,RDBIT
4481 022076 012703 052046 MOV #HEAD2,R3 ;:LOAD HEADER
4482 022102 012701 000003 MOV #3,R1 ;:LOAD WORDS PER HEADER
4483 022106 012304 10$: MOV (R3)+,R4 ;:GET NEXT WORD
4484 022110 012700 000020 MOV #16.,R0 ;:LOAD BITS PER WORD
4485 022114 013737 003254 12$: MOV PR.BIT,M1.BIT ;:STORE PREVIOUS BIT
4486 022122 006004 ROR R4 ;:GET NEXT BIT
4487 022124 103403 BCS 15$
4488 022126 005037 003254 CLR PR.BIT
4489 022132 000403 BR 16$
4490
4491 022134 012737 000001 003254 15$: MOV #1,PR.BIT
4492 022142 004737 035422 16$: JSR PC,RDBIT ;:SIMULATE NEXT BIT
4493 022146 005300 DEC R0 ;:CHECK IF READY FOR NEXT WORD
4494 022150 001361 BNE 12$ ;:NO, CONTINUE
4495 022152 005301 DEC R1 ;:CHECK IF FINISHED WITH HEADER
4496 022154 001354 BNE 10$ ;:NO, CONTINUE
4497 022156 012700 000101 MOV #65.,R0
4498 022162 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;:SIMULATE GAP
4499 022170 005037 003254 CLR PR.BIT
4500 022174 004737 035422 JSR PC,RDBIT
4501 022200 005300 DEC R0 ;:CHECK IF GAP FINISHED
4502 022202 001367 BNE 20$ ;:NO, CONTINUE
4503 022204 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;:LOAD COUNT UNTIL POSTAMBLE
4504 022210 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4505 022216 012762 000040 000026 MOV #DMD,RKMR1(R2)
4506 022224 005300 DEC R0
4507 022226 001370 BNE 25$
4508 022230 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;:GET CS1
4509 022236 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;:GET CS2
4510 022244 016237 000014 003154 MOV RKER(R2),T.ER ;:GET ERROR REG
```

```

4511 022252 012737 100222 003200 MOV #CERR.RDY!WRDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4512 022260 012737 000300 003210 MOV #IR!OR,F.CS2 ;LOAD EXPECTED CS2
4513 022266 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG
4514 022274 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4515 022302 001401 BEQ 30$ ;YES, CONTINUE
4516 022304 104136 ERROR 136 ;CS1 INCORRECT
4517 022306 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4518 022314 001401 BEQ 32$ ;YES, CONTINUE
4519 022316 104137 ERROR 137 ;CS2 INCORRECT
4520 022320 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4521 022326 001401 BEQ 35$ ;YES - SKIP
4522 022330 104140 ERROR 140 ;ERROR REG INCORRECT
4523 022332 012762 100000 000000 35$: MOV #CLR,RKCS1(R2) ;CLEAR CONTROLLER
4524 022340 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4525 022346 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4526 022354 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4527 022362 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4528 022370 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4529 022376 001401 BEQ TST36 ;GO TO NEXT TEST
4530 022400 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.
*****

```

```

4545 022402 000004 TST36: SCOPE
4546 022404 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
4547 022412 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4548 022416 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
4549 022424 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4550 022432 012762 053672 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
4551 022440 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
4552 022446 012762 000300 000020 MOV #300,RKDCYL(R2) ;LOAD CYLINDER 11DMS
4553 022454 012762 000000 000006 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
4554 022462 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE COMMAND
4555 022470 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
4556 ; READY FOR SECTOR PULSE.
4557 022474 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)
4558 022502 012762 000040 000026 MOV #DMD,RKMR1(R2)
4559 022510 005300 DEC R0
4560 022512 001370 BNE 1$
4561 022514 012762 000140 000026 MOV #DMD.MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4562 022522 012762 000040 000026 MOV #DMD,RKMR1(R2)
4563 022530 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4564 022534 005037 003256 CLR M1.BIT
4565 022540 012700 000377 MOV #255,R0
4566 022544 004737 035422 5$: JSR PC,RDBIT

```



```
4567 022550 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4568 022552 001374 BNE 5$
4569 022554 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4570 022562 004737 035422 JSR PC,RDBIT
4571 022566 012703 052150 MOV #HEAD10,R3 ;LOAD HEADER
4572 022572 012701 000C03 MOV #3,R1 ;LOAD WORDS PER HEADER
4573 022576 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4574 022600 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4575 022604 013737 003254 003256 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4576 022612 006004 ROR R4 ;GET NEXT BIT
4577 022614 103403 BCS 15$
4578 022616 005037 003254 CLR PR.BIT
4579 022622 000403 BR 16$
4580
4581 022624 012737 000001 003254 15$: MOV #1,PR.BIT
4582 022632 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4583 022636 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4584 022640 001361 BNE 12$ ;NO, CONTINUE
4585 022642 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4586 022644 001354 BNE 10$ ;NO, CONTINUE
4587 022646 012700 000101 MOV #65,R0
4588 022652 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4589 022660 005037 003254 CLR PR.BIT
4590 022664 004737 035422 JSR PC,RDBIT
4591 022670 005300 DEC R0 ;CHECK IF GAP FINISHED
4592 022672 001367 BNE 20$ ;NO, CONTINUE
4593 022674 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4594 022700 012762 000440 000026 25$: MOV #DMD.MCLK,RKMR1(R2)
4595 022706 012762 000040 000026 MOV #DMD,RKMR1(R2)
4596 022714 005300 DEC R0
4597 022716 001370 BNE 25$
4598 022720 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4599 022726 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4600 022734 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4601 022742 012737 100222 003200 MOV #CERR!RDY.WRDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4602 022750 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4603 022756 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4604 022764 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4605 022772 001401 BEQ 30$ ;YES, CONTINUE
4606 022774 104141 ERROR 141 ;CS1 INCORRECT
4607 022776 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4608 023004 001401 BEQ 32$ ;YES, CONTINUE
4609 023006 104142 ERROR 142 ;CS2 INCORRECT
4610 023010 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4611 023016 001401 BEQ 35$ ;YES - SKIP
4612 023020 104143 ERROR 143 ;ERROR REG INCORRECT
4613 023022 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4614 023030 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4615 023036 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4616 023044 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4617 023052 012737 000200 003200 MOV #2C0,E.CS1 ;SET EXPECTED CS1
4618 023060 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4619 023066 001401 BEQ TS37 ;GO TO NEXT TEST
4620 023070 104153 ERROR 153 ;ERROR DID NOT CLEAR
4621
4622
```

4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635 023072 000004
4636 023074 012737 000012 001200
4637 023102 013702 001270
4638 023106 012762 100000 000000
4639 023114 012762 000040 000026
4640 023122 012762 053672 000004
4641 023130 012762 177777 000002
4642 023136 012762 000300 000020
4643 023144 012762 000000 000006
4644 023152 012762 000021 000000
4645 023160 012700 000426
4646
4647 023164 012762 000440 000026 1\$:
4648 023172 012762 000040 000026
4649 023200 005300
4650 023202 001370
4651 023204 012762 000140 000026
4652 023212 012762 000040 000026
4653 023220 005037 003254
4654 023224 005037 003256
4655 023230 012700 000377
4656 023234 004737 035422 5\$:
4657 023240 005300
4658 023242 001374
4659 023244 012737 000001 003254
4660 023252 004737 035422
4661 023256 012703 05250
4662 023262 012701 000003
4663 023266 012304 10\$:
4664 023270 012700 000020
4665 023274 013737 003254 003256 12\$:
4666 023302 006004
4667 023304 103403
4668 023306 005037 003254
4669 023312 000403
4670
4671 023314 012737 000001 003254 15\$:
4672 023322 004737 035422 16\$:
4673 023326 005300
4674 023330 001361
4675 023332 005301
4676 023334 001354
4677 023336 012700 000101
4678 023342 013737 003254 003256 20\$:

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 SURF
CONTROLLER ERROR RESETS.

TST37: 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 11DMS
MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
MOV #RDDATA,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,RDBIT
DEL R0 ;CHECK IF SYNCH FINISHED
BNE 5\$
MOV #1,PR.BIT ;SIMULATE SYNCH
JSR PC,RDBIT
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,RDBIT ;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

```
4679 023350 005037 003254 CLR PR.BIT
4680 023354 004737 035422 JSR PC,RDB:1
4681 023360 005300 DEC R0 ;CHECK IF GAP FINISHED
4682 023362 001367 BNE 20$ ;NO, CONTINUE
4683 023364 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4684 023370 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4685 023376 012762 000040 000026 MOV #DMD,RKMR1(R2)
4686 023404 005300 DEC R0
4687 023406 001370 BNE 25$
4688 023410 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4689 023416 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4690 023424 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4691 023432 012737 100220 003200 MOV #CERR!RDY!RDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4692 023440 012737 000100 003210 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4693 023446 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4694 023454 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4695 023462 001401 BEQ 30$ ;YES, CONTINUE
4696 023464 104141 ERROR 141 ;CS1 INCORRECT
4697 023466 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4698 023474 001401 BEQ 32$ ;YES, CONTINUE
4699 023476 104142 ERROR 142 ;CS2 INCORRECT
4700 023500 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4701 023506 001401 BEQ 35$ ;YES - SKIP
4702 023510 104143 ERROR 143 ;ERROR REG INCORRECT
4703 023512 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4704 023520 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4705 023526 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4706 023534 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4707 023542 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4708 023550 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4709 023556 001401 BEQ TST40 ;GO TO NEXT TEST
4710 023560 104153 ERROR 153 ;ERROR DID NOT CLEAR
```

```
4711
4712
4713 :*****
4714 :*TEST 40 WRITE CHECK AND HVRC
4715 :*
4716 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4717 :* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4718 :* CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4719 :* CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4720 :* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
4721 :* A HEADER WITH A HEADER VRC ERROR. MAKE SURE
4722 :* HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
4723 :* AND MAKE SURE CONTROLLER ERROR RESETS.
4724 :*****
```

```
4725 023562 000004 TST40: SCOPE
4726 023564 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
4727 023572 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4728 023576 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4729 023604 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4730 023612 012762 053672 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
4731 023620 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
4732 023626 012762 000300 000020 MOV #300,RKDCYL(R2) ;LOAD CYLINDER 11DMS
4733 023634 012762 000000 000006 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
4734 023642 012762 000031 000000 MOV #WRTCHK,RKCS1(R2) ;ISSUE COMMAND
```

```
4735 023650 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
4736 ; READY FOR SECTOR PULSF.
4737 023654 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4738 023662 012762 000040 000026 MOV #DMD,RKMR1(R2)
4739 023670 005300 DEC R0
4740 023672 001370 BNE 1$
4741 023674 012762 000140 000026 MOV #DMD.MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4742 023702 012762 000040 000026 MOV #DMD,RKMR1(R2)
4743 023710 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4744 023714 005037 003256 CLR M1.BIT
4745 023720 012700 000377 MOV #255.,R0
4746 023724 004737 035422 5$: JSR PC,RDBIT
4747 023730 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4748 023732 001374 BNE 5$
4749 023734 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4750 023742 004737 035422 JSR PC,RDBIT
4751 023746 012703 052150 MOV #HEAD10,R3 ;LOAD HEADER
4752 023752 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4753 023756 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4754 023760 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4755 023764 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4756 023772 006004 ROR R4 ;GET NEXT BIT
4757 023774 103403 BCS 15$
4758 023776 005037 003254 CLR PR.BIT
4759 024002 000403 BR 16$
4760
4761 024004 012737 000001 003254 15$: MOV #1,PR.BIT
4762 024012 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4763 024016 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4764 024020 001361 BNE 12$ ;NO, CONTINUE
4765 024022 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4766 024024 001354 BNE 10$ ;NO, CONTINUE
4767 024026 012700 000101 MOV #65.,R0
4768 024032 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4769 024040 005037 003254 CLR PR.BIT
4770 024044 004737 035422 JSR PC,RDBIT
4771 024050 005300 DEC R0 ;CHECK IF GAP FINISHED
4772 024052 001367 BNE 20$ ;NO, CONTINUE
4773 024054 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4774 024060 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4775 024066 012762 000040 000026 MOV #DMD,RKMR1(R2)
4776 024074 005300 DEC R0
4777 024076 001370 BNE 25$
4778 024100 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4779 024106 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4780 024114 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4781 024122 012737 100230 003200 MOV #CERR!RDY!WRTCHK<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4782 024130 012737 000100 003210 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4783 024136 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4784 024144 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4785 024152 001401 BEQ 30$ ;YES, CONTINUE
4786 024154 104141 ERROR 141 ;CS1 INCORRECT
4787 024156 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4788 024164 001401 BEQ 32$ ;YES, CONTINUE
4789 024166 104142 ERROR 142 ;CS2 INCORRECT
4790 024170 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
```

```
4791 024176 001401 BEQ 35$ ;YES - SKIP
4792 024200 104143 ERROR 143 ;ERROR REG INCORRECT
4793 024202 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4794 024210 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4795 024216 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4796 024224 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4797 024232 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4798 024240 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4799 024246 001401 BEQ TST41 ;GO TO NEXT TEST
4800 024250 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 0. 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.
*****
```

```
4818 024252 000004 TST41: SCOPE
4819 024254 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
4820 024262 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4821 024266 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4822 024274 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4823 024302 012762 052164 000004 MOV #ECC1,RKBA(R2) ;LOAD ADDRESS OF ECC DATA
4824 024310 012762 177770 000002 MOV #-10,RKWC(R2) ;WORD COUNT =10
4825 024316 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
4826 024324 012700 000450 MOV #8.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL
4827 ; READY FOR SECTOR PULSE
4828 024330 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4829 024336 012762 000040 000026 MOV #DMD,RKMR1(R2)
4830 024344 005300 DEC R0
4831 024346 001370 BNE 1$
4832 024350 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4833 024356 012762 000040 000026 MOV #DMD,RKMR1(R2)
4834 024364 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4835 024370 005037 003256 CLR M1.BIT
4836 024374 012700 000377 MOV #255.,R0
4837 024400 004737 035422 5$: JSR PC,RDBIT
4838 024404 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4839 024406 001374 BNE 5$
4840 024410 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4841 024416 004737 035422 JSR PC,RDBIT
4842 024422 012701 000003 MOV #3,R1 ;SIMULATE HEADER
4843 024426 012703 052040 MOV #HEAD1,R3 ; CYL 0, TRK 0, SECTOR 0
4844 024432 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4845 024434 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4846 024440 013737 003254 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
```

```
4847 024446 006004 ROR R4 ;GET NEXT BIT
4848 024450 103403 BCS 17$
4849 024452 005037 003254 CLR PR.BIT
4850 024456 000403 BR 18$
4851
4852 024460 012737 000001 003254 17$: MOV #1,PR.BIT
4853 024466 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4854 024472 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
4855 024474 001361 BNE 15$ ;NO, CONTINUE
4856 024476 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4857 024500 001354 BNE 12$ ;NO, CONTINUE
4858 024502 012700 000101 MOV #65,R0 ;SIMULATE GAP +1 FOR SWITCH FROM READ TO WRITE
4859 024506 013737 003254 003256 20$: MOV PR.BIT,M1.BIT
4860 024514 005037 003254 CLR PR.BIT
4861 024520 004737 035422 JSR PC,RDBIT
4862 024524 005300 DEC R0 ;CHECK IF GAP FINISHED
4863 024526 001367 BNE 20$ ;NO, CONTINUE
4864 024530 012700 000400 MOV #256,R0 ;LOAD COUNT FOR SYNCH FIELD
4865 024534 012737 047344 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
4866 024542 005037 003252 CLR P1.BIT ;INITIALIZE BITS
4867 024546 005037 003254 CLR PR.BIT
4868 024552 005037 003256 CLR M1.BIT
4869 024556 005037 003260 CLR M2.BIT
4870 024562 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
4871 024566 012737 062040 003224 MOV #DMD!ECCW!MEWD.WRTGAT,E.MR1 ;LOAD EXPECTED MR1
4872 024574 004737 034674 25$: JSR PC,WRTBIT ;WRITE BIT
4873 024600 104002 ERROR 2 ;DATA INCORRECT
4874 024602 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
4875 024606 005300 DEC R0 ;CHECK IF FINISHED
4876 024610 001371 BNE 25$ ;NO, CONTINUE
4877 024612 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
4878 024620 004737 034674 JSR PC,WRTBIT
4879 024624 104002 ERROR 2
4880 024626 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
4881 024632 012737 047416 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
4882 024640 005037 003234 CLR E.ECPT ;CLEAR EXPECTED ECC PATTERN
4883 024644 012703 052164 MOV #ECC1,R3 ;LOAD START OF DATA
4884 024650 012701 000004 MOV #4,R1 ;LOAD COUNT
4885 024654 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
4886 024656 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4887 024662 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
4888 024670 013737 003254 003256 MOV PR.BIT,M1.BIT
4889 024676 013737 003252 003254 MOV P1.BIT,PR.BIT
4890 024704 006004 ROR R4 ;GET NEXT BIT
4891 024706 103403 BCS 34$
4892 024710 005037 003252 CLR P1.BIT
4893 024714 000403 BR 35$
4894
4895 024716 012737 000001 003252 34$: MOV #1,P1.BIT
4896 024724 004737 034674 35$: JSR PC,WRTBIT ;SIMULATE BIT
4897 024730 104002 ERROR 2
4898 024732 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
4899 024740 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
4900 024746 001401 BEQ 37$ ;YES, CONTINUE
4901 024750 104134 ERROR 134 ;ECC PATTERN NOT ZERO
4902 024752 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
```

4903	024756	005300	DEC	R0	:CHECK IF FINISHED WITH WORD
4904	024760	001340	BNE	32\$:NO, CONTINUE
4905	024762	005301	DEC	R1	:CHECK IF FINISHED WITH TEST
4906	024764	001333	BNE	30\$:NO, CONTINUE

4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929

```
*****  
*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.  
*****
```

4930	024766	000004				TST42: SCOPE	
4931	024770	012737	000012	001200		MOV	#10., \$TIMES ;:DO 10. ITERATIONS
4932	024776	013702	001270			MOV	\$BASE, R2 ;:LOAD RK611 BASE
4933	025002	012762	100000	000000		MOV	#CCLR, RKCS1(R2) ;:CLEAR RK611
4934	025010	012762	000040	000026		MOV	#DMD, RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
4935	025016	012762	052204	000004		MOV	#ECC2, RKBA(R2) ;:LOAD ECC DATA
4936	025024	012762	177766	000002		MOV	#-12, RKWC(R2) ;:WORD COUNT=12
4937	025032	012762	000023	000000		MOV	#WRDATA, RKCS1(R2) ;:ISSUE WRITE DATA
4938	025040	012700	000562			MOV	#10.*37., R0 ;:ISSUE ENOUGH CLOCKS UNTIL ;: READY FOR SECTOR PULSE
4939							
4940	025044	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)
4941	025052	012762	000040	000026		MOV	#DMD, RKMR1(R2)
4942	025060	005300				DEC	R0
4943	025062	001370				BNE	1\$
4944	025064	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2) ;:SIMULATE SECTOR PULSE
4945	025072	012762	000040	000026		MOV	#DMD, RKMR1(R2)
4946	025100	005037	003254			CLR	PR.BIT ;:GENERATE SYNCH
4947	025104	005037	003256			CLR	M1.BIT
4948	025110	012700	000377			MOV	#255., R0
4949	025114	004737	035422		5\$:	JSR	PC, RDBIT
4950	025120	005300				DEC	R0 ;:CHECK IF SYNCH FINISHED
4951	025122	001374				BNE	5\$
4952	025124	012737	000001	003254		MOV	#1, PR.BIT ;:SIMULATE SYNCH BIT
4953	025132	004737	035422			JSR	PC, RDBIT
4954	025136	012701	000003			MOV	#3, R1 ;:SIMULATE HEADER
4955	025142	012703	052040			MOV	#HEAD1, R3 ;: CYL 0, TRK 0, SECTOR 0
4956	025146	012304			12\$:	MOV	(R3)+, R4 ;:GET NEXT HEADER WORD
4957	025150	012700	000020			MOV	#16., R0 ;:LOAD BITS PER WORD
4958	025154	013737	003254	003256	15\$:	MOV	PR.BIT, M1.BIT ;:STORE PREVIOUS BIT

```
4959 025162 006004 ROR R4 ;GET NEXT BIT
4960 025164 103403 BCS 17$
4961 025166 005037 003254 CLR PR.BIT
4962 025172 000403 BR 18$
4963
4964 025174 012737 000001 003254 17$: MOV #1,PR.BIT
4965 025202 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4966 025206 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
4967 025210 001361 BNE 15$ ;NO, CONTINUE
4968 025212 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4969 025214 001354 BNE 12$ ;NO, CONTINUE
4970 025216 012700 000101 MOV #65,R0 ;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
4971 025222 013737 003254 003256 20$: MOV PR.BIT,M1.BIT
4972 025230 005037 003254 CLR PR.BIT
4973 025234 004737 035422 JSR PC,RDBIT
4974 025240 005300 DEC R0 ;CHECK IF GAP FINISHED
4975 025242 001367 BNE 20$ ;NO, CONTINUE
4976 025244 012700 000400 MOV #256,R0 ;LOAD COUNT FOR SYNCH FIELD
4977 025250 012737 047344 001310 MOV #EM327,EMW ;LOAD ERROR
4978 025256 005037 003252 CLR P1.BIT ;INITIALIZE BITS
4979 025262 005037 003254 CLR PR.BIT
4980 025266 005037 003256 CLR M1.BIT
4981 025272 005037 003260 CLR M2.BIT
4982 025276 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
4983 025302 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
4984 025310 004737 034674 25$: JSR PC,WRTBIT ;WRITE BIT
4985 025314 104002 ERROR 2 ;DATA INCORRECT
4986 025316 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
4987 025322 005300 DEC R0 ;CHECK IF FINISHED
4988 025324 001371 BNE 25$ ;NO, CONTINUE
4989 025326 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
4990 025334 004737 034674 JSR PC,WRTBIT
4991 025340 104002 ERROR 2
4992 025342 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
4993 025346 012737 047416 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
4994 025354 005037 003266 CLR ECCHI ;INITIALIZE ECC
4995 025360 005037 003270 CLR ECCLO
4996 025364 012703 052204 MOV #ECC2,R3 ;LOAD START OF DATA
4997 025370 012701 000006 MOV #6,R1 ;LOAD COUNT
4998 025374 012304 30$: MOV (R3)+,R4 ;GET NEXT BIT
4999 025376 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
5000 025402 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5001 025410 013737 003254 003256 MOV PR.BIT,M1.BIT
5002 025416 013737 003252 003254 MOV P1.BIT,PR.BIT
5003 025424 006004 ROR R4 ;GET NEXT BIT
5004 025426 103403 BCS 34$
5005 025430 005037 003252 CLR P1.BIT
5006 025434 000403 BR 35$
5007
5008 025436 012737 000001 003252 34$: MOV #1,P1.BIT
5009 025444 004737 034674 35$: JSR PC,WRTBIT ;SIMULATE BIT
5010 025450 104002 ERROR 2
5011 025452 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
5012 025460 004737 034526 JSR PC,ECCGEN ;GENERATE EXPECTED ECC PAT
5013 025464 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5014 025472 001401 BE 37$ ;YES, CONTINUE
```


5015 025474 104135
5016 025476 005237 003262
5017 025502 005300
5018 025504 001336
5019 025506 005301
5020 025510 001331

ERROR 135 ;ECC PATTERN INCORRECT
37\$: INC BITCN ;INCREMENT BIT COUNT
DEC R0 ;CHECK IF FINISHED WITH WORD
BNE 32\$;NO, CONTINUE
DEC R1 ;CHECK IF FINISHED WITH TEST
BNE 30\$;NO, CONTINUE

5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043

*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.

5044 025512 000004
5045 025514 012737 000012 001200
5046 025522 013702 001270
5047 025526 012762 100000 000C00
5048 025534 012762 000040 000026
5049 025542 012762 052224 000004
5050 025550 012762 177766 000002
5051 025556 012762 000023 000000
5052 025564 012700 000562
5053
5054 025570 012762 000440 000026 1\$:
5055 025576 012762 000040 000026
5056 025604 005300
5057 025606 001370
5058 025610 012762 000140 000026
5059 025616 012762 000040 000026
5060 025624 005037 003254
5061 025630 005037 003256
5062 025634 012700 000377
5063 025640 004737 035422 5\$:
5064 025644 005300
5065 025646 001374
5066 025650 012737 000001 003254
5067 025656 004737 035422
5068 025662 012701 000003
5069 025666 012703 052040
5070 025672 012304 12\$:

*ST43: 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 #ECC3, 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 UNTI
* ; 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, RDBIT
* DEC R0 ;CHECK IF SYNCH FINISHED
* BNE 5\$
* MOV #1, PR.BIT ;SIMULATE SYNCH BIT
* JSR PC, RDBIT
* MOV #3, R1 ;SIMULATE HEADER
* MOV #HEAD1, R3 ;CYL 0, TRK 0, SECTOR 0
* 12\$:
* MOV (R3)+, R4 ;GET NEXT HEADER WORD

5071	025674	012700	000020			MOV	#16.,R0	;LOAD BITS PER WORD
5072	025700	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
5073	025706	006004				ROR	R4	;GET NEXT BIT
5074	025710	103403				BCS	17\$	
5075	025712	005037	003254			CLR	PR.BIT	
5076	025716	000403				BR	18\$	
5077								
5078	025720	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
5079	025726	004737	035422		18\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
5080	025732	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
5081	025734	001361				BNE	15\$;NO, CONTINUE
5082	025736	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
5083	025740	001354				BNE	12\$;NO, CONTINUE
5084	025742	012700	000101			MOV	#65.,R0	;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5085	025746	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT	
5086	025754	005037	003254			CLR	PR.BIT	
5087	025760	004737	035422			JSR	PC,RDBIT	
5088	025764	005300				DEC	R0	;CHECK IF GAP FINISHED
5089	025766	001367				BNE	20\$;NO, CONTINUE
5090	025770	012700	000400			MOV	#256.,R0	;LOAD COUNT FOR SYNCH FIELD
5091	025774	012737	047344	001310		MOV	#EM327,EMW	;LOAD ERROR
5092	026002	005037	003252			CLR	P1.BIT	;INITIALIZE BITS
5093	026006	005037	003254			CLR	PR.BIT	
5094	026012	005037	003256			CLR	M1.BIT	
5095	026016	005037	003260			CLR	M2.BIT	
5096	026022	005037	003262			CLR	BITCNT	;INITIALIZE BIT COUNT
5097	026026	012737	062040	003224		MOV	#DMD!ECCW.MEWD.WRTGAT,E.MR1	;SET EXPECTED MR1
5098	026034	004737	034674		25\$:	JSR	PC,WRTBIT	;WRITE BIT
5099	026040	104002				ERROR	2	;DATA INCORRECT
5100	026042	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
5101	026046	005300				DEC	R0	;CHECK IF FINISHED
5102	026050	001371				BNE	25\$;NO, CONTINUE
5103	026052	012737	000001	003252		MOV	#1,P1.BIT	;SIMULATE SYNCH BIT
5104	026060	004737	034674			JSR	PC,WRTBIT	
5105	026064	104002				ERROR	2	
5106	026066	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
5107	026072	012737	047416	001310		MOV	#EM328,EMW	;LOAD ERROR MESSAGE
5108	026100	005037	003266			CLR	ECCHI	;INITIALIZE ECC
5109	026104	005037	003270			CLR	ECCL0	
5110	026110	012703	052224			MOV	#ECC3,R3	;LOAD START OF DATA
5111	026114	012701	000006			MOV	#6,R1	;LOAD COUNT
5112	026120	012304			30\$:	MOV	(R3)+,R4	;GET NEXT BIT
5113	026122	012700	000020			MOV	#16.,R0	;LOAD BITS PER WORD
5114	026126	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5115	026134	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5116	026142	013737	003252	003254		MOV	P1.BIT,PR BIT	
5117	026150	006004				ROR	R4	;GET NEXT BIT
5118	026152	103403				BCS	34\$	
5119	026154	005037	003252			CLR	P1.BIT	
5120	026160	000403				BR	35\$	
5121								
5122	026162	012737	000001	003252	34\$:	MOV	#1,P1.BIT	
5123	026170	004737	034674		35\$:	JSR	PC,WRTBIT	;SIMULATE BIT
5124	026174	104002				ERROR	2	
5125	026176	016237	000032	003174		MOV	RKE(P1(R2),T.ECPT	;STORE ECC WORD
5126	026204	004737	034526			JSR	PC,ECCGEN	;GENERATE EXPECTED ECC PAT

```

5127 026210 023737 003234 003174      CMP      E.ECPT,T.ECPT      ;CHECK ECC PATTERN CORRECT
5128 026216 001401                      BEQ      37$                ;YES, CONTINUE
5129 026220 104135                      ERROR    135                ;ECC PATTERN INCORRECT
5130 026222 005237 003262      37$:    INC      BITCNT        ;INCREMENT BIT COUNT
5131 026226 005300                      DEC      R0                  ;CHECK IF FINISHED WITH WORD
5132 026230 001336                      BNE     32$                ;NO, CONTINUE
5133 026232 005301                      DEC     R1                  ;CHECK IF FINISHED WITH TEST
5134 026234 001331                      BNE     30$                ;NO, CONTINUE
5135
5136
5137      ;*****
5138      ;*TEST 44      ECC WRITING
5139      ;*
5140      ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5141      ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
5142      ;*      DATA OF 400 WORDS.  TO AN RK06 IN 26 SECTOR FORMAT,
5143      ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
5144      ;*      AND DRIVE CLEAR MESSAGE.  SIMULATE A SECTOR PULSE
5145      ;*      AND A GOOD HEADER.  CLOCK THROUGH ALL 400 WORDS
5146      ;*      AND THE TWO ECC WORDS.  MAKE SURE THE TWO ECC WORDS
5147      ;*      ARE CORRECT AND WRITTEN PROPERLY.  CHECK BUS ADDRESS,
5148      ;*      WORD COUNT, CYLINDER, TRACK, AND SECTOR.
5149      ;*
5150      ;*****
5150 026236 000004      TST44:  SCOPE
5151 026240 012737 000012 001200      MOV     #10.,$TIMES      ;;DO 10. ITERATIONS
5152
5153 026246 013702 001270      MOV     $BASE,R2        ;LOAD RK611 BASE
5154 026252 012762 100000 000000      MOV     #CLR,RKCS1(R2)  ;CLEAR RK611
5155 026260 012762 000040 000026      MOV     #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5156 026266 012762 054672 000004      MOV     #ECCBUF,RKBA(R2);LOAD ADDRESS
5157 026274 012762 177400 000002      MOV     #-400,RKWC(R2) ;WORD COUNT=400
5158 026302 012762 000023 000000      MOV     #WRDATA,RKCS1(R2);ISSUE WRITE DATA
5159 026310 012700 004612      MOV     #66.*37.,R0     ;ISSUE ENOUGH CLOCKS UNTIL
5160                                ;   READY FOR SECTOR PULSE
5161 026314 012762 000440 000026 1$:    MOV     #DMD!MCLK,RKMR1(R2)
5162 026322 012762 000040 000026      MOV     #DMD,RKMR1(R2)
5163 026330 005300                      DEC     R0
5164 026332 001370                      BNE     1$
5165 026334 012762 000140 000026      MOV     #DMD!MSP,RKMR1(R2);SIMULATE SECTOR PULSE
5166 026342 012762 000040 000026      MOV     #DMD,RKMR1(R2)
5167 026350 005037 003254                      CLR     PR.BIT
5168 026354 005037 003256                      CLR     M1.BIT
5169 026360 012700 000377                      MOV     #255.,R0
5170 026364 004737 035422      5$:    JSR     PC,RDBIT
5171 026370 005300                      DEC     R0                  ;CHECK IF SYNCH FINISHED
5172 026372 001374                      BNE     5$
5173 026374 012737 000001 003254      MOV     #1,PR.BIT        ;SIMULATE SYNCH
5174 026402 004737 035422                      JSR     PC,RDBIT
5175 026406 012703 052040                      MOV     #HEAD1,R3        ;LOAD HEADER
5176 026412 012701 000003                      MOV     #3,R1            ;LOAD WORDS PER HEADER
5177 026416 012304      10$:   MOV     (R3)+,R4          ;GET NEXT WORD
5178 026420 012700 000020                      MOV     #16.,R0         ;LOAD BITS PER
5179 026424 013737 003254 003256 12$:   MOV     PR.BIT,M1.BIT    ;STORE PREVIOUS BIT
5180 026432 006004                      ROR     R4                ;GET NEXT BIT
5181 026434 103403                      BCS     15$
5182 026436 005037 003254                      CLR     PR.BIT

```

```
5183 026442 000403 BR 16$
5184
5185 026444 012737 000001 003254 15$: MOV #1,PR.BIT
5186 026452 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5187 026456 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
5188 026460 001361 BNE 12$ ;NO, CONTINUE
5189 026462 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5190 026464 001354 BNE 10$ ;NO, CONTINUE
5191 026466 012700 000101 MOV #65.,R0
5192 026472 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5193 026500 005037 003254 CLR PR.BIT
5194 026504 004737 035422 JSR PC,RDBIT
5195 026510 005300 DEC R0 ;CHECK IF GAP FINISHED
5196 026512 001367 BNE 20$ ;NO, CONTINUE
5197 026514 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH
5198 026520 005037 003254 CLR PR.BIT
5199 026524 005037 003256 CLR M1.BIT
5200 026530 005037 003260 CLR M2.BIT
5201 026534 012737 047344 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5202 026542 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5203 026546 012700 000400 MOV #256.,R0 ;LOAD SYNCH COUNT AND WRITE SYNCH
5204 026552 012737 062040 003224 MOV #DMD!ECCW.MEWD.WRTGAT,F.MR1 ;SET EXPECTED MR1
5205 026560 004737 034674 25$: JSR PC,WRTBIT
5206 026564 104002 ERROR 2
5207 026566 005237 003262 INC BITCNT ;CLEAR BIT COUNT
5208 026572 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5209 026574 001371 BNE 25$ ;NO, CONTINUE
5210 026576 012737 000001 003252 MOV #1,P1.BIT ;WRITE SYNCH BIT
5211 026604 004737 034674 JSR PC,WRTBIT
5212 026610 104002 ERROR 2
5213 026612 005037 003266 CLR ECCHI ;INITIALIZE ECC GENERATOR
5214 026616 005037 003270 CLR ECCLO
5215 026622 012737 047416 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5216 026630 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5217 026634 012701 000400 MOV #256.,R1 ;LOAD WORDS PER SECTOR
5218 026640 012703 054672 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5219 026644 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5220 026646 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
5221 026652 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5222 026660 013737 003254 003256 MOV PR.BIT,M1.BIT
5223 026666 013737 003252 003254 MOV P1.BIT,PR.BIT
5224 026674 006004 ROR R4 ;DETERMINE NEXT BIT
5225 026676 103403 BCS 34$
5226 026700 005037 003252 CLR P1.BIT
5227 026704 000403 BR 35$
5228
5229 026706 012737 000001 003252 34$: MOV #1,P1.BIT
5230 026714 004737 034674 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5231 026720 104002 ERROR 2
5232 026722 004737 034526 JSR PC,ECCGEN ;GENERATE ECC
5233 026726 016237 000032 003174 MOV RKECPT(R2),I.ECPT ;GET PATTERN
5234 026734 023737 003234 003174 CMP E.ECPT,I.ECPT ;CHECK ECC PATTERN CORRECT
5235 026742 001401 BEQ 37$ ;YES, CONTINUE
5236 026744 104135 ERROR 135 ;ECC PATTERN INCORRECT
5237 026746 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5238 026752 022700 000002 CMP #2,R0 ;IF THE NEXT BIT IS THE LAST BIT OF
```

5239	026756	001006				BNE	28\$:THE LAST DATA WORD, ECCW MUST BE RESET
5240	026760	022701	000001			CMP	#1,R1		:IN THE EXPECTED MR1 TO INDICATE ECC
5241	026764	001003				BNE	28\$:BEING WRITTEN
5242	026766	042737	020000	003224		BIC	#ECCW,E.MR1		
5243	026774	005300			28\$:	DEC	R0		:CHECK IF THROUGH WITH WORD
5244	026776	001325				BNE	32\$:NO, CONTINUE
5245	027000	005301				DEC	R1		:CHECK IF AT END OF SFCTOR
5246	027002	001320				BNE	30\$:NO, CONTINUE
5247	027004	012737	047763	001310		MOV	#EM333,EMW		:LOAD ERROR MESSAGE
5248	027012	005037	003262			CLR	BITCNT		:INITIALIZE BIT COUNT
5249	027016	012701	000002			MOV	#2,R1		:LOAD NUMBER OF ECO WORDS
5250	027022	012703	003266			MOV	#ECCHI,R3		:LOAD ADDRESS OF ECC
5251	027026	012304			40\$:	MOV	(R3)+,R4		:GET NEXT ECC WORD
5252	027030	012700	000020			MOV	#16,R0		:LOAD BITS PER WORD
5253	027034	013737	003256	003260	42\$:	MOV	M1.BIT,M2.BIT		:SHIFT BITS
5254	027042	013737	003254	003256		MOV	PR.BIT,M1.BIT		
5255	027050	013737	003252	003254		MOV	P1.BIT,PR.BIT		
5256	027056	006004				ROR	R4		:DETERMINE NEXT BIT
5257	027060	103403				BCS	44\$		
5258	027062	005037	003252			CLR	P1.BIT		
5259	027066	000403				BR	45\$		
5260									
5261	027070	012737	000001	003252	44\$:	MOV	#1,P1.BIT		
5262	027076	004737	034674		45\$:	JSR	PC,WRTBIT		:WRITE NEXT BIT
5263	027102	104002				ERROR	2		
5264	027104	005237	003262			INC	BITCNT		:INCREMENT BIT COUNT
5265	027110	022700	000002			CMP	#2,R0		:IF THE LAST BIT OF THE LAST ECC WORD
5266	027114	001006				BNE	46\$:IS BEING WRITTEN, ECCW MUST BE SET
5267	027116	022701	000001			CMP	#1,R1		:IN EXPECTED MR1 TO INDICATE ECC
5268	027122	001003				BNE	46\$:WRITING IS DONE
5269	027124	052737	020000	003224		BIS	#ECCW,E.MR1		
5270	027132	005300			46\$:	DEC	R0		:CHECK IF THROUGH WITH WORD
5271	027134	001337				BNE	42\$:NO, CONTINUE
5272	027136	005301				DEC	R1		:CHECK IF FINISH WITH ECC
5273	027140	001332				BNE	40\$:NO, CONTINUE
5274	027142	012737	050021	001310		MOV	#EM334,EMW		:LOAD ERROR MESSAGE
5275	027150	005037	003262			CLR	BITCNT		:CLEAR BIT COUNT
5276	027154	012700	000017			MOV	#15,R0		:LOAD POSTAMBLE BIT COUNT
5277	027160	013737	003256	003260	47\$:	MOV	M1.BIT,M2.BIT		:SHIFT BIT
5278	027166	013737	003254	003256		MOV	PR.BIT,M1.BIT		
5279	027174	013737	003252	003254		MOV	P1.BIT,PR.BIT		
5280	027202	005037	003252			CLR	P1.BIT		
5281	027206	004737	034674			JSR	PC,WRTBIT		:WRITE NEXT BIT
5282	027212	104002				ERROR	2		
5283	027214	005237	003262			INC	BITCNT		:INCREMENT BIT COUNT
5284	027220	005300				DEC	R0		:CHECK IF THROUGH WITH POSTAMBLE
5285	027222	001356				BNE	47\$:NO, CONTINUE
5286	027224	012700	000014			MOV	#3*4,R0		:ISSUE CLOCKS TO COMPLETE COMMAND
5287	027230	012762	000440	000026	50\$:	MOV	#DMD.MCLK,RKMR1(R2)		:ISSUE CLOCK PULSES
5288	027236	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
5289	027244	005300				DEC	R0		
5290	027246	001370				BNE	50\$		
5291	027250	016237	000000	003140		MOV	RKCS1(R2),T.CS1		:SAVE CS1
5292	027256	016237	000010	003150		MOV	RKCS2(R2),T.CS2		:SAVE CS2
5293	027264	016237	000014	003154		MOV	RKER(R2),T.ER		:SAVE ERROR REG
5294	027272	012737	000222	003200		MOV	#RDY!WRDATA<^C<GO>>,E.CS1		:LOAD EXPECTED CS1

```

5295 027300 012737 000100 003210 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
5296 027306 005037 003214 CLR E.ER ;LOAD EXPECTED ERROR REG.
5297 027312 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;SAVE CYLINDER ADDR REG
5298 027320 016237 000006 003146 MOV RKDA(R2),T.DA ;SAVE DISK AND REG
5299 027326 016237 000004 003144 MOV RKBA(R2),T.BA ;SAVE BUS ADD REG
5300 027334 016237 000002 003142 MOV RKWC(R2),T.WC ;SAVE WORD COUNT
5301 027342 005037 003220 CLR E.DCYL ;LOAD EXPECTED CYLINDER AND REG.
5302 027346 012737 000001 003206 MOV #1,E.DA ;LOAD EXPECTED DISK
5303 027354 012737 055672 003204 MOV #ECCBLF+<400*2>,E.BA ;LOAD EXPECTED
5304 027362 005037 003202 CLR E.WC ;LOAD EXPECTED ECC WORD
5305 027366 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1
5306 027374 001401 BEQ 55$
5307 027376 104144 ERROR 144
5308 027400 023737 003210 003150 55$: CMP E.CS2,T.CS2 ;CHECK CS2
5309 027406 001401 BEQ 56$
5310 027410 104145 ERROR 145
5311 027412 023737 003214 003154 56$: CMP E.ER,T.ER ;CHECK ERROR REG
5312 027420 001401 BEQ 57$
5313 027422 104146 ERROR 146
5314 027424 023737 003204 003144 57$: CMP E.BA,T.BA ;CHECK BUS ADD
5315 027432 001401 BFO 58$
5316 027434 104147 ERROR 147
5317 027436 023737 003202 003142 58$: CMP E.WC,T.WC ;CHECK WORD COUNT
5318 027444 001401 BEQ 59$
5319 027446 104150 ERROR 150
5320 027450 023737 003220 003160 59$: CMP E.DCYL,T.DCYL ;CHECK CYLINDER ADD
5321 027456 001401 BEQ 60$
5322 027460 104151 ERROR 151
5323 027462 023737 003206 003146 60$: CMP E.DA,T.DA ;CHECK DISK ADDR.
5324 027470 001401 BEQ TST45 ;:YES, GO ON TO NEXT TEST
5325 027472 104152 ERROR 152
    
```

.SBTTL **PARTIAL WRITE DATA

```

5326
5327
5328
5329
5330 :*****
5331 :*TEST 45 ZERO FILL ON WRITE DATA
5332 :*
5333 :* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5334 :* PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
5335 :* DATA OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
5336 :* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5337 :* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
5338 :* AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS AND
5339 :* THE TWO ECC WORDS. CHECK THE SECTOR FOR ZERO FILL
5340 :* AND MAKE SURE THE TWO ECC WORDS ARE WRITTEN PROPERLY.
5341 :* CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECTOR.
5342 :*
    
```

```

5343 027474 000004 TST45: SCOPE
5344 027476 012737 000012 001200 MOV #10, $TIMES ;:DO 10. ITERATIONS
5345
5346 027504 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5347 027510 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
5348 027516 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5349 027524 012762 054672 000004 MOV #ECCBUF,RKBA(R2) ;LOAD ADDRESS
5350 027532 012762 177675 000002 MOV #-103,RKWC(R2) ;WORD COUNT=400
    
```

```

5351 027540 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
5352 027546 012700 004612 MOV #65.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5353 ; READY FOR SECTOR PULSE
5354 027552 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5355 027560 012762 000040 000026 MOV #DMD,RKMR1(R2)
5356 027566 005300 DEC R0
5357 027570 001370 BNE 1$
5358 027572 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5359 027600 012762 000040 000026 MOV #DMD,RKMR1(R2)
5360 027606 005037 003254 CLR PR.BIT
5361 027612 005037 003256 CLR M1.BIT
5362 027616 012700 000377 MOV #255.,R0
5363 027622 004737 035422 5$: JSR PC,RDBIT
5364 027626 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5365 027630 001374 BNE 5$
5366 027632 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
5367 027640 004737 035422 JSR PC,RDBIT
5368 027644 012703 052040 MOV #HEAD1,R3 ;LOAD HEADER
5369 027650 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
5370 027654 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5371 027656 012700 000020 MOV #16.,R0 ;LOAD BITS PER
5372 027662 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5373 027670 006004 ROR R4 ;GET NEXT BIT
5374 027672 103403 BCS 15$
5375 027674 005037 003254 CLR PR.BIT
5376 027700 000463 BR 16$
5377
5378 027702 012737 000001 003254 15$: MOV #1,PR.BIT
5379 027710 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5380 027714 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
5381 027716 001361 BNE 12$ ;NO, CONTINUE
5382 027720 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5383 027722 001354 BNE 10$ ;NO, CONTINUE
5384 027724 012700 000101 MOV #65.,R0
5385 027730 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5386 027736 005037 003254 CLR PR.BIT
5387 027742 004737 035422 JSR PC,RDBIT
5388 027746 005300 DEC R0 ;CHECK IF GAP FINISHED
5389 027750 001367 BNE 20$ ;NO, CONTINUE
5390 027752 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH
5391 027756 005037 003254 CLR PR.BIT
5392 027762 005037 003256 CLR M1.BIT
5393 027766 005037 003260 CLR M2.BIT
5394 027772 012737 047344 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5395 030000 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5396 030004 012700 000400 MOV #256.,R0 ;LOAD SYNCH COUNT AND WRITE SYNCH
5397 030010 012737 062040 003224 MOV #DMD!ECCW.MEWD.WRTGAT,E.MR1 ;SET EXPECTED MR1
5398 030016 004737 034674 25$: JSR PC,WRTBIT
5399 030022 104002 ERROR 2
5400 030024 005237 003262 INC BITCNT ;CLEAR BIT COUNT
5401 030030 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5402 030032 001371 BNE 25$ ;NO, CONTINUE
5403 030034 012737 000001 003252 MOV #1,P1.BIT ;WRITE SYNCH BIT
5404 030042 004737 034674 JSR PC,WRTBIT
5405 030046 104002 ERROR 2
5406 030050 005037 003266 CLR ECCHI ;INITIALIZE ECC GENERATOR
  
```

```
5407 030054 005037 003270 CLR ECCLO
5408 030060 012737 047416 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5409 030066 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5410 030072 012701 000400 MOV #256,R1 ;LOAD WORDS PER SECTOR
5411 030076 012703 054672 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5412 030102 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5413 030104 012700 000020 31$: MOV #16,R0 ;LOAD BITS PER WORD
5414 030110 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5415 030116 013737 003254 003256 MOV PR.BIT,M1.BIT
5416 030124 013737 003252 003254 MOV P1.BIT,PR.BIT
5417 030132 006004 ROR R4 ;DETERMINE NEXT BIT
5418 030134 103403 BCS 34$
5419 030136 005037 003252 CLR P1.BIT
5420 030142 000403 BR 35$
5421
5422 030144 012737 000001 003252 34$: MOV #1,P1.BIT
5423 030152 004737 034674 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5424 030156 104002 ERROR 2
5425 030160 004737 034526 JSR PC,ECCGEN ;GENERATE ECC
5426 030164 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET PATTERN
5427 030172 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5428 030200 001401 BEQ 37$ ;YES, CONTINUE
5429 030202 104135 ERROR 135 ;ECC PATTERN INCORRECT
5430 030204 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5431 030210 022700 000002 CMP #2,R0 ;IF THE NEXT BIT IS THE LAST BIT OF
5432 030214 001006 BNE 28$ ;THE LAST DATA WORD, ECCW MUST BE RESET
5433 030216 022701 000001 CMP #1,R1 ;IN THE EXPECTED MR1 TO INDICATE ECC
5434 030222 001003 BNE 28$ ;BEING WRITTEN
5435 030224 042737 020000 003224 BIC #ECCW,E.MR1
5436 030232 005300 28$: DEC R0 ;CHECK IF THROUGH WITH WORD
5437 030234 001325 BNE 32$ ;NO, CONTINUE
5438 030236 005301 DEC R1 ;CHECK IF AT END OF SECTOR
5439 030240 001405 BEQ 39$ ;YES -SKIP
5440 030242 022703 055100 CMP #ECCBUF+<103*2>,R3 ;CHECK IF 103 WORDS TRANSFERED
5441 030246 003315 BGT 30$ ;NO - GO TO GET NEXT WORD
5442 030250 005004 CLR R4 ;ELSE CLEAR R4 FOR ZEROS
5443 030252 000714 BR 31$ ;GO SIMULATE REST OF SECTOR
5444 030254 012737 047763 001310 39$: MOV #EM333,EMW ;LOAD ERROR MESSAGE
5445 030262 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5446 030266 012701 000002 MOV #2,R1 ;LOAD NUMBER OF ECC WORDS
5447 030272 012703 003266 MOV #ECCHI,R3 ;LOAD ADDRESS OF ECC
5448 030276 012304 40$: MOV (R3)+,R4 ;GET NEXT ECC WORD
5449 030300 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
5450 030304 013737 003256 003260 42$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5451 030312 013737 003254 003256 MOV PR.BIT,M1.BIT
5452 030320 013737 003252 003254 MOV P1.BIT,PR.BIT
5453 030326 006004 ROR R4 ;DETERMINE NEXT BIT
5454 030330 103403 BCS 44$
5455 030332 005037 003252 CLR P1.BIT
5456 030336 000403 BR 45$
5457
5458 030340 012737 000001 003252 44$: MOV #1,P1.BIT
5459 030346 004737 034674 45$: JSR PC,WRTBIT ;WRITE NEXT BIT
5460 030352 104002 ERROR 2
5461 030354 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5462 030360 022700 000002 CMP #2,R0 ;IF THE LAST BIT OF THE LAST ECC WORD
```


5463	030364	001006				BNE	46\$; IS BEING WRITTEN, ECCW MUST BE SET
5464	030366	022701	000001			CMP	#1,R1		; IN EXPECTED MR1 TO INDICATE ECC
5465	030372	001003				BNE	46\$; WRITING IS DONE
5466	030374	052737	020000	003224		BIS	#ECCW,E.MR1		
5467	030402	005300			46\$:	DEC	R0		; CHECK IF THROUGH WITH WORD
5468	030404	001337				BNE	42\$; NO, CONTINUE
5469	030406	005301				DEC	R1		; CHECK IF FINISH WITH ECC
5470	030410	001332				BNE	40\$; NO, CONTINUE
5471	030412	012737	050021	001310		MOV	#EM334,EMW		; LOAD ERROR MESSAGE
5472	030420	005037	003262			CLR	BITCNT		; CLEAR BIT COUNT
5473	030424	012700	000017			MOV	#15,R0		; LOAD POSTAMBLE BIT COUNT
5474	030430	013737	003256	003260	47\$:	MOV	M1.BIT,M2.BIT		; SHIFT BIT
5475	030436	013737	003254	003256		MOV	PR.BIT,M1.BIT		
5476	030444	013737	003252	003254		MOV	P1.BIT,PR.BIT		
5477	030452	005037	003252			CLR	P1.BIT		
5478	030456	004737	034674			JSR	PC,WRTBIT		; WRITE NEXT BIT
5479	030462	104002				ERROR	2		
5480	030464	005237	003262			INC	BITCNT		; INCREMENT BIT COUNT
5481	030470	005300				DEC	R0		; CHECK IF THROUGH WITH POSTAMBLE
5482	030472	001356				BNE	47\$; NO, CONTINUE
5483	030474	012700	000014			MOV	#3*4,R0		; ISSUE CLOCKS TO COMPLETE COMMAND
5484	030500	012762	000440	000026	50\$:	MOV	#DMD!MCLK,RKMR1(R2)		; ISSUE CLOCK PULSES
5485	030506	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
5486	030514	005300				DEC	R0		
5487	030516	001370				BNE	50\$		
5488	030520	016237	000000	003140		MOV	RKCS1(R2),T.CS1		; SAVE CS1
5489	030526	016237	000010	003150		MOV	RKCS2(R2),T.CS2		; SAVE CS2
5490	030534	016237	000014	003154		MOV	RKER(R2),T.ER		; SAVE ERROR REG
5491	030542	012737	000222	003200		MOV	#RDY!WRDATA&<^C<GO>>,E.CS1		; LOAD EXPECTED CS1
5492	030550	012737	000100	003210		MOV	#IR,E.CS2		; LOAD EXPECTED CS2
5493	030556	005037	003214			CLR	E.ER		; LOAD EXPECTED ERROR REG.
5494	030562	016237	000020	003160		MOV	RKDCYL(R2),T.DCYL		; SAVE CYLINDER ADDR REC
5495	030570	016237	000006	003146		MOV	RKDA(R2),T.DA		; SAVE DISK AND REG
5496	030576	016237	000004	003144		MOV	RKBA(R2),T.BA		; SAVE BUS ADDR REG
5497	030604	016237	000002	003142		MOV	RKWC(R2),T.WC		; SAVE WORD COUNT
5498	030612	005037	003220			CLR	E.DCYL		; LOAD EXPECTED CYLINDER AND REG.
5499	030616	012737	000001	003206		MOV	#1,E.DA		; LOAD EXPECTED DISK
5500	030624	012737	055100	003204		MOV	#ECCBUF+<103*2>,E.BA		; LOAD EXPECTED
5501	030632	005037	003202			CLR	E.WC		; LOAD EXPECTED ECC WORD
5502	030636	023737	003200	003140		CMP	E.CS1,T.CS1		; CHECK CS1
5503	030644	001401				BEQ	55\$		
5504	030646	104154				ERROR	154		
5505	030650	023737	003210	003150	55\$:	CMP	E.CS2,T.CS2		; CHECK CS2
5506	030656	001401				BEQ	56\$		
5507	030660	104155				ERROR	155		
5508	030662	023737	003214	003154	56\$:	CMP	E.ER,T.ER		; CHECK ERROR REG
5509	030670	001401				BEQ	57\$		
5510	030672	104156				ERROR	156		
5511	030674	023737	003204	003144	57\$:	CMP	E.BA,T.BA		; CHECK BUS ADD
5512	030702	001401				BEQ	58\$		
5513	030704	104157				ERROR	157		
5514	030706	023737	003202	003142	58\$:	CMP	E.WC,T.WC		; CHECK WORD COUNT
5515	030714	001401				BEQ	59\$		
5516	030716	104160				ERROR	160		
5517	030720	023737	003220	003160	59\$:	CMP	E.DCYL,T.DCYL		; CHECK CYLINDER ADD
5518	030726	001401				BEQ	60\$		

```
5519 030730 104167 ERROR 161
5520 030732 023737 003206 003146 60$: CMP E.DA,T.DA ;CHECK DISK ADDR.
5521 030740 001401 BEQ TST46 ;:YES, GO ON TO NEXT TEST
5522 030742 104162 ERROR 162
5523 .SBTTL **18 BIT FORMAT WRITES
5524
5525 ;NOTE: SINCE 18-BIT FUNCTIONALITY OF THE RK611 IS NOT
5526 ;UTILIZED ON THE PDP-11, THE FOLLOWING TESTS, INTENDED
5527 ;FOR MANUFACTURING USE, ARE NOT EXECUTED UNLESS LOCA-
5528 ;TION 'SEC24' IS PATCHED TO A NON-ZERO VALUE.
5529
5530 :*****
5531 :*TEST 46 18 BIT WRITE DATA (PART 1)
5532 :*
5533 :* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5534 :* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
5535 :* OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
5536 :* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5537 :* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
5538 :* AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777.
5539 :* VRFIFY THAT TWO ZERO BITS ARE PRESENT FOR EACH WORD.
5540 :*
5541 :*****
5542 030744 000004 TST46: SCOPE
5543 030746 012737 000012 001200 MOV #10, $TIMES ;:DO 10. ITERATIONS
5544 030754 005737 003314 TST SEC24 ;SKIP 18 BIT TESTS?
5545 030760 001002 BNE 1$ ;BRANCH IF NOT
5546 030762 000137 032760 JMP $EOP ;ELSE GO TO END OF PASS
5547 030766
5548 1$:
5549 030766 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5550 030772 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5551 031000 012700 002000 MOV #2000,R0 ;SET A STALL COUNT
5552 031004 005300 5$: DEC R0 ;LOOP UNTIL COUNT 0
5553 031006 001376 BNE 5$
5554
5555 031010 012762 000040 000026 MOV #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5556 031016 012762 053656 000004 MOV #MOD2BF,RKBA(R2) ;LOAD BUFFER ADDRESS
5557 031024 012762 177400 000002 MOV #-400,RKWC(R2) ; --- WORD COUNT
5558
5559 031032 012762 010023 000000 MOV #WRDATA!CFMT,RKCS1(R2) ; --- START COMMAND
5560 031040 012700 005136 MOV #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5561 ; READY FOR SECTOR PULSE
5562 031044 012762 000440 000026 7$: MOV #DMD!MCLK,RKMR1(R2)
5563 031052 012762 000040 000026 MOV #DMD,RKMR1(R2)
5564 031060 005300 DEC R0
5565 031062 001370 BNE 7$
5566
5567 031064 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5568 031072 012762 000040 000026 MOV #DMD,RKMR1(R2)
5569
5570 031100 005037 003254 CLR PR.BIT ;GENERATE SYNC
5571 031104 005037 003256 CLR M1.BIT
5572 031110 012700 000377 MOV #255.,R0
5573
5574 031114 004737 035422 9$: JSR PC,RDBIT ;SIMULATE SYNC 0 BITS
```

```

5575 031120 005300      DEC      R0
5576 031122 001374      BNE      9$
5577
5578 031124 012737 000001 003254      MOV      #1,PR.BIT      ;SIMULATE SYNC 1 BIT
5579 031132 004737 035422      JSR      PC,RDBIT
5580
5581 031136 012703 052156      MOV      #HEAD11,R3      ;SIMULATE HEADER
5582 031142 012701 000003      MOV      #3,R1      ; CYL 0 TRK 0, SEC 0
5583 031146 012304      11$:      MOV      (R3)+,R4      ;GET HEADER WORD
5584 031150 012700 000020      MOV      #16.,R0      ;LOAD BITS PER WORD
5585 031154 013737 003254 003256 13$:      MOV      PR.BIT,M1.BIT      ;STORE PREVIOUS BIT
5586 031162 006004      ROR      R4      ;GET NEXT BIT
5587 031164 103403      BCS      15$
5588 031166 005037 003254      CLR      PR.BIT
5589 031172 000403      BR       16$
5590
5591 031174 012737 000001 003254 15$:      MOV      #1,PR.BIT
5592 031202 004737 035422 16$:      JSR      PC,RDBIT      ;SIMULATE NEXT BIT
5593 031206 005300      DEC      R0      ;READY FOR NEXT HEADER WORD?
5594 031210 001361      BNE      13$      ;NO - GET NEXT BIT THIS WORD
5595 031212 005301      DFC      R1      ;HEADER DONE?
5596 031214 001354      BNE      11$      ;NO - GET NEXT HEADER WORD
5597 031216 012700 000101      MOV      #65.,R0      ;SET COUNT FOR GAP.
5598 031222 013737 003254 003256 20$:      MOV      PR.BIT,M1.BIT      ;SIMULATE GAP
5599 031230 005037 003254      CLR      PR.BIT
5600 031234 004737 035422      JSR      PC,RDBIT
5601 031240 005300      DEC      R0
5602 031242 001367      BNE      20$
5603 031244 012700 000400      MOV      #256.,R0      ;SET COUNT FOR WRITE DATA SYNC
5604 031250 012737 047344 001310      MOV      #EM327,EMW      ;LOAD ERROR MESSAGE
5605 031256 005037 003252      CLR      P1.BIT      ;CLEAR BITS
5606 031262 005037 003254      CLR      PR.BIT
5607 031266 005037 003256      CLR      M1.BIT
5608 031272 005037 003260      CLR      M2.BIT
5609 031276 005037 003262      CLR      BITCNT      ;CLEAR BIT COUNTER
5610 031302 012737 062040 003224      MOV      #DMD!ECCW.MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5611 031310 004737 034674 22$:      JSR      PC,WRTBIT      ;SIMULATE SYNC 0
5612 031314 104002      ERROR    2
5613 031316 005237 003262      INC      BITCNT      ;BUMP BIT COUNT
5614 031322 005300      DEC      R0      ;LOOP UNTIL SYNC 0 WRITTEN
5615 031324 001371      BNE      22$
5616 031326 012737 000001 003252      MOV      #1,P1.BIT      ;SIMULATE SYNC 1
5617 031334 004737 034674      JSR      PC,WRTBIT
5618 031340 104002      ERROR    2
5619 031342 005037 003266      CLR      ECCHI      ;INITIALIZE ECC WORDS.
5620 031346 005037 003270      CLR      ECCLO
5621 031352 005037 003262      CLR      BITCNT      ;CLEAR BIT COUNT
5622 031356 012703 053656      MOV      #MOD2BF,R3      ;SET DATA POINTER
5623 031362 012701 00C006      MOV      #6.,R1      ;SET DATA COUNT
5624 031366 012304 24$:      MOV      (R3)+,R4      ;GET DATA WORD
5625 031370 012737 050267 001310      MOV      #EM338,EMW      ;LOAD MESSAGE (18 BIT DATA WRITE)
5626 031376 012700 000022      MOV      #18.,R0      ;LOAD BITS PER WORD
5627 031402 013737 003256 003260 25$:      MOV      M1.BIT,M2.BIT      ;SHIFT BITS
5628 031410 013737 003254 003256      MOV      PR.BIT,M1.BIT
5629 031416 013737 003252 003254      MOV      P1.BIT,PR.BIT
5630 031424 000241      CLC      ;CLEAR CARRY & GET NEXT BIT

```

```
5631 031426 006004 ROR R4
5632 031430 103403 BCS 27$
5633 031432 005037 003252 CLR P1.BIT
5634 031436 000403 BR 28$
5635
5636 031440 012737 000001 003252 27$: MOV #1,P1.BIT
5637 031446 004737 034674 28$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5638 031452 104002 ERROR 2
5639 031454 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET ECC PATTERN
5640 031462 004737 034526 JSR PC,ECCGEN ;COMPUTE EXPECTED ECC
5641 031466 023737 003174 003234 CMP T.ECPT,E.ECPT ;CHECK IF CORRECT
5642 031474 001401 BEQ 29$ ;YES - SKIP
5643 031476 104164 ERROR 164
5644 031500 005237 003262 29$: INC BITCNT
5645 031504 005300 DEC R0
5646 031506 020027 000002 CMP R0,#2 ;1ST 16 BITS JUST DONE?
5647 031512 001403 BEQ 31$ ;YES - SKIP
5648 031514 005700 TST R0 ;ELSE TEST IF WORD DONE
5649 031516 001331 BNE 25$ ;NO - DO NEXT BIT
5650 031520 000406 BR 32$ ;ELSE DO NEXT WORD
5651 031522 012737 050355 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5652 031530 012704 000000 MOV #0,R4 ;SET UPPER BITS OF WORD
5653 031534 000722 BR 25$ ;GO DO BITS
5654
5655 031536 005301 32$: DEC R1 ;ALL WORDS DONE?
5656 031540 001312 BNE 24$ ;NO - GET NEXT WCRD
5657
```

```
5658 :*****
5659 :*TEST 47      18 BIT WRITE DATA (PART 2)
5660 :*
5661 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5662 :* PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
5663 :* OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
5664 :*
5665 :* CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
5666 :* AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
5667 :* AND A GOOD HEADER.  CLOCK THROUGH 400 18 BIT WORDS
5668 :* AND THE 32 BIT ECC.  VERIFY THAT THE ECC IS
5669 :* WRITTEN CORRECTLY.
5670 :*
5671 :*****
5672 031542 000004 TST47: SCOPE
5673 031544 012737 000012 001200 MOV #10.,$TIMES ;DO 10. ITERATIONS
5674
5675
5676 031552 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5677 031556 012762 100000 000000 MOV #CCLR,RKCS(R2) ;CLEAR CONTROLLER
5678 031564 012700 002000 MOV #2000,R0 ;SET A STALL COUNT
5679 031570 005300 5$: DEC R0 ;LOOP UNTIL COUNT 0
5680 031572 001376 BNE 5$
5681
5682 031574 012762 000040 000026 MOV #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5683 031602 012762 054672 000004 MOV #ECCBUF,RKBA(R2) ;LOAD BUFFER ADDRESS
5684 031610 012762 177400 000002 MOV #-400,RKWC(R2) ; --- WORD COUNT
5685
5686 031616 012762 010023 000000 MOV #WRDATA!CFMT,RKCS1(R2) ; --- START COMMAND
5687 031624 012700 005136 MOV #66.*40 +<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5688 : ; READY FOR SECTOR PULSE
5689 031630 012762 000440 000026 7$: MOV #DMD!MCLK,RKMR1(R2)
5690 031636 012762 000040 000026 MOV #DMD,RKMR1(R2)
5691 031644 005300 DEC R0
5692 031646 001370 BNE 7$
5693
5694 031650 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5695 031656 012762 000040 000026 MOV #DMD,RKMR1(R2)
5696
5697 031664 005037 003254 CLR PR.BIT ;GENERATE SYNC
5698 031670 005037 003256 CLR M1.BIT
5699 031674 012700 000377 MOV #255.,R0
5700
5701 031700 004737 035422 9$: JSR PC,RDBIT ;SIMULATE SYNC 0 BITS
5702 031704 005300 DEC R0
5703 031706 001374 BNE 9$
5704
5705 031710 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNC 1 BIT
5706 031716 004737 035422 JSR PC,RDBIT
5707
5708 031722 012703 052156 MOV #HEAD11,R3 ;SIMULATE HEADER
5709 031726 012701 000003 MOV #3,R1 ; CYL 0 TRK 0, SEC 0
5710 031732 012304 11$: MOV (R3)+,R4 ;GET HEADER WORD
5711 031734 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
5712 031740 013737 003254 13$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5713 031746 006004 ROR R4 ;GET NEXT BIT
```

```
5714 031750 103403          BCS 15$
5715 031752 005037 003254   CLR  PR.BIT
5716 031756 000403          BR   16$
5717
5718 031760 012737 000001 003254 15$:  MOV  #1,PR.BIT
5719 031766 004737 035422 16$:  JSR  PC,RDBIT      ;SIMULATE NEXT BIT
5720 031772 005300          DEC  R0            ;READY FOR NEXT HEADER WORD?
5721 031774 001361          BNE  13$          ;NO - GET NEXT BIT THIS WORD
5722 031776 005301          DEC  R1            ;HEADER DONE?
5723 032000 001354          BNE  11$          ;NO - GET NEXT HEADER WORD
5724 032002 012700 000101  MOV  #65,R0       ;SET COUNT FOR GAP.
5725 032006 013737 003254 003256 20$:  MOV  PR.BIT,M1.BIT ;SIMULATE GAP
5726 032014 005037 003254   CLR  PR.BIT
5727 032020 004737 035422   JSR  PC,RDBIT
5728 032024 005300          DEC  R0
5729 032026 001367          BNE  20$
5730 032030 012700 000400  MOV  #256,R0      ;SET COUNT FOR WRITE DATA SYNC
5731 032034 012737 047344 001310  MOV  #EM327,EMW   ;LOAD ERROR MESSAGE
5732 032042 005037 003252   CLR  P1.BIT       ;CLEAR BITS
5733 032046 005037 003254   CLR  PR.BIT
5734 032052 005037 003256   CLR  M1.BIT
5735 032056 005037 003260   CLR  M2.BIT
5736 032062 005037 003262   CLR  BITCNT       ;CLEAR BIT COUNTER
5737 032066 012737 062040 003224  MOV  #DMD.ECCW.MEWD.WRTGAT,E.1/R1 ;SET EXPECTED MR1
5738 032074 004737 034674 22$:  JSR  PC,WRTBIT    ;SIMULATE SYNC 0
5739 032100 104002          ERROR 2
5740 032102 005237 003262   INC  BITCNT       ;BUMP BIT COUNT
5741 032106 005300          DEC  R0            ;LOOP UNTIL SYNC 0 WRITTEN
5742 032110 001371          BNE  22$
5743 032112 012737 000001 003252  MOV  #1,P1.BIT    ;SIMULATE SYNC 1
5744 032120 004737 034674   JSR  PC,WRTBIT
5745 032124 104002          ERROR 2
5746 032126 005037 003266   CLR  ECCHI        ;INITIALIZE ECC WORDS.
5747 032132 005037 003270   CLR  ECCLO
5748 032136 005037 003262   CLR  BITCNT       ;CLEAR BIT COUNT
5749 032142 012703 054672   MOV  #ECCBUF,R3   ;SET DATA POINTER
5750 032146 012701 000377   MOV  #255,R1      ;SET DATA COUNT
5751 032152 012304          MOV  (R3)+,R4     ;GET DATA WORD
5752 032154 012737 050267 001310  MOV  #EM338,EMW   ;LOAD MESSAGE (18 BIT DATA WRITE)
5753 032162 012700 000022   MOV  #18,R0       ;LOAD BITS PER WORD
5754 032166 013737 003256 003260 25$:  MOV  M1.BIT,M2.BIT ;SHIFT BITS
5755 032174 013737 003254 003256   MOV  PR.BIT,M1.BIT
5756 032202 013737 003252 003254   MOV  P1.BIT,PR.BIT
5757 032210 000241          CLC
5758 032212 006004          ROR  R4            ;CLEAR CARRY & GET NEXT BIT
5759 032214 103403          BCS  27$
5760 032216 005037 003252   CLR  P1.BIT
5761 032222 000403          BR   28$
5762
5763 032224 012737 000001 003252 27$:  MOV  #1,P1.BIT
5764 032232 004737 034674 28$:  JSR  PC,WRTBIT    ;SIMULATE NEXT BIT
5765 032236 104002          ERROR 2
5766 032240 016237 000032 003174  MOV  RKECPT(R2),T.ECPT ;GET ECC PATTERN
5767 032246 004737 034526   JSR  PC,ECCGEN    ;COMPUTE EXPECTED ECC
5768 032252 023737 003174 003234  CMP  T.ECPT,E.ECPT ;CHECK IF CORRECT
5769 032260 001401          BEQ  29$          ;YES - SKIP
```

```

5770 032262 104164          ERROR 164
5771 032264 005237 003262    29$:  INC  BITCNT
5772 032270 005300          DEC  R0
5773 032272 020027 000002    CMP  R0,#2          ;1ST 16 BITS JUST DONE?
5774 032276 001403          BEQ  31$           ;YES - SKIP
5775 032300 005700          TST  R0           ;ELSE TEST IF WORD DONE
5776 032302 001331          BNE  25$           ;NO - DO NEXT BIT
5777 032304 000406          BR   32$           ;ELSE DO NEXT WORD
5778 032306 012737 050355 001310 31$:  MOV  #EM339,EMW    ;LOAD MESSAGE (BIT 16..17)
5779 032314 012704 000000    MOV  #0,R4        ;SET UPPER BITS OF WORD
5780 032320 000722          BR   25$           ;GO DO BITS
5781
5782 032322 005301          32$:  DEC  R1           ;ALL WORDS DONE?
5783 032324 001312          BNE  24$           ;NO - GET NEXT WORD
5784
5785 032326 012737 050267 001310    MOV  #EM338,EMW    ;SET ERROR MESSAGE
5786 032334 012304          MOV  (R3)+,R4      ;GET LAST WORD
5787 032336 012700 000022    MOV  #18.,R0       ;SET BIT COUNT
5788 032342 013737 003256 003260 34$:  MOV  M1.BIT,M2.BIT ;SHIFT BITS
5789 032350 013737 003254 003256    MOV  PR.BIT,M1.BIT
5790 032356 013737 003252 003254    MOV  P1.BIT,PR.BIT
5791 032364 000241          CLC                    ;CLEAR CARRY
5792 032366 006004          ROR  R4           ;TEST NEXT BIT
5793 032370 103403          BCS  36$           ;SKIP IF A ONE
5794 032372 005037 003252    CLR  P1.BIT        ;ELSE SET NEXT BIT FOR ZERO
5795 032376 000403          BR   37$
5796
5797 032400 012737 000001 003252 36$:  MOV  #1,P1.BIT     ;SET NEXT BIT FOR 1
5798 032406 004737 034674 37$:  JSR  PC,WRTBIT     ;SIMULATE BIT
5799 032412 104002          ERROR 2
5800 032414 016237 000032 003174    MOV  RKECPT(R2),T.ECPT ;GET PATTERN
5801 032422 004737 034526    JSR  PC,ECCGEN     ;GENERATE ECC PATTERN
5802 032426 023737 003174 003234    CMP  T.ECPT,E.ECPT ;TEST IF CORRECT
5803 032434 001401          BEQ  39$           ;YES - SKIP
5804 032436 104164          ERROR 164          ;ELSE REPORT
5805 032440 005237 003262    39$:  INC  BITCNT        ;BUMP BIT COUNT
5806 032444 022700 000002    CMP  #2,R0        ;IF NEXT BIT IS THE LAST DATA BIT
5807 032450 001003          BNE  41$           ;ECCW MUST BE RESET TO INDICATE
5808 032452 042737 020000 003224          BIC  #ECCW,E.MR1   ;ECC BEING WRITTEN
5809 032460 005300          DEC  R0
5810 032462 022700 000002    41$:  CMP  #2,R0        ;TEST IF 1ST 16 BITS DONE
5811 032466 001403          BEQ  42$           ;YES - SKIP
5812 032470 005700          TST  R0           ;ELSE TEST IF WORD DONE
5813 032472 001323          BNE  34$           ;NO - SKIP
5814 032474 000406          BR   45$           ;ELSE GO TO ECC PROCESSING
5815
5816 032476 012737 050355 001310 42$:  MOV  #EM339,EMW    ;SET MESSAGE FOR BITS 16 AND 17
5817 032504 012704 000000    MOV  #0,R4        ;SET UP UPPER BITS
5818 032510 000714          BR   34$           ;GO DO LAST TWO BITS
5819 032512 012737 047763 001310 45$:  MOV  #EM333,EMW    ;MESSAGE (ECC WRITE)
5820 032520 005037 003262    CLR  BITCNT        ;CLEAR BIT COUNT
5821 032524 012703 003266    MOV  #ECCHI,R3     ;LOAD POINTER TO ECC WORDS
5822 032530 012701 000002    MOV  #2,R1        ;SET FOR 2 WORDS
5823 032534 012304          43$:  MOV  (R3)+,R4      ;GET ECC WORD
5824 032536 012700 000020    MOV  #16.,R0       ;SET BIT COUNT
5825 032542 013737 003256 003260 44$:  MOV  M1.BIT,M2.BIT ;SHIFT BITS
  
```

```

5826 032550 013737 003254 003256 MOV PR.BIT,M1.BIT
5827 032556 013737 003252 003254 MOV P1.BIT,PR.BIT
5828 032564 006004 ROR R4 ;LOAD NEXT BIT
5829 032566 103403 BCS 46$
5830 032570 005037 003252 CLR P1.BIT
5831 032574 000403 BR 47$
5832
5833 032576 012737 000001 003252 46$: MOV #1,P1.BIT
5834 032604 004737 034674 47$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5835 032610 104002 ERROR 2
5836 032612 022700 000002 CMP #2,R0 ;IF THIS BIT IS THE LAST ECC BIT
5837 032616 001006 BNE 49$ ;ECCW MUST BE SET TO INDICATE
5838 032620 022701 000001 CMP #1,R1 ;ECC IS DONE
5839 032624 001003 BNE 49$
5840 032626 052737 020000 003224 BIS #ECCW,E.MR1
5841 032634 005237 003262 49$: INC BITCNT ;BUMP BIT COUNT
5842 032640 005300 DEC R0 ;TEST IF LAST BIT THIS WORD IS DONE
5843 032642 001337 BNE 44$ ;NO - LOOP
5844 032644 005301 DEC R1 ;TEST IF BOTH WORDS WRITTEN
5845 032646 001332 BNE 43$ ;NO - LOOP
5846 032650 012737 050021 001310 MOV #EM334,EMW ;LOAD MESSAGE (POSTAMBLE)
5847 032656 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5848 032662 012700 000017 MOV #15,R0 ;SET GAP COUNT
5849 032666 013737 003256 003260 51$: MOV M1.BIT,M2.BIT ;SHIFT REST OF BITS
5850 032674 013737 003254 003256 MOV PR.BIT,M1.BIT
5851 032702 013737 003252 003254 MOV P1.BIT,PR.BIT
5852 032710 005037 003252 CLR P1.BIT ;CLEAR NEXT BIT
5853 032714 004737 034674 JSR PC,WRTBIT ;SIMULATE BIT
5854 032720 104002 ERROR 2
5855 032722 005237 003262 INC BITCNT ;BUMP BIT COUNT
5856 032726 005300 DEC R0 ;GAP WRITTEN?
5857 032730 001356 BNE 51$ ;NO - LOOP
5858 032732 012737 000001 003234 MOV #1,E.ECPT ;SET EXPECTED PATTERN
5859 032740 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET '611 PATTERN
5860 032746 023737 003174 003234 CMP T.ECPT, E.ECPT ;TEST IF EQUAL
5861 032754 001401 BEQ 56$ ;YES - SKIP
5862 032756 104163 ERROR 163 ;ELSE REPORT
5863 032760
5864 .SBT'L END OF PASS ROUTINE
5865
5866 ;*****
5867 ;*INCREMENT THE PASS NUMBER ($PASS)
5868 ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYYY'
5869 ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
5870 ;*IF THERES A MONITOR GO TO IT
5871 ;*IF THERE ISN'T JUMP TO TST1
5872
5873 $EOP:
5874 032760 000004 SCOPE
5875 032762 005037 001102 CLR $TSTNM ;;ZERO THE TEST NUMBER
5876 032766 005037 001200 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
5877 032772 005237 001222 INC $PASS ;;INCREMENT THE PASS NUMBER
5878 032776 042737 100000 001222 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
5879 033004 005327 DEC (PC)+ ;;LOOP?
5880 033006 000001 $EOPCT: .WORD 1
5881 033010 003063 BGT $DOAGN ;;YES

```



```

5882 033012 012737
5883 033014 00000
5884 033016 033006
5885 033020 104401 033026
5886 033024 000407
5887
5888 033044
5889 033044 013746 001222
5890
5891 033050 104405
5892 033052 104401 033060
5893 033056 000421
5894
5895 033122
5896 033122 013746 001112
5897
5898 033126 104405
5899 033130 104401 001211
5900 033134 005037 001112
5901 033140 013700 000042
5902 033144 001405
5903 033146 000005
5904 033150 004710
5905 033152 000240
5906 033154 000240
5907 033156 000240
5908 033160
5909 033160 000137
5910 033162 004312
5911 033164 377 377 000
5912 033170
5913
5914
5915
5916
5917
5918
5919
5920 033170 013702 001270 000000
5921 033174 012762 100000
5922 033202 104401 042305
5923 033206 012706 001100
5924 033212 005037 001202
5925 033216 105037 001103
5926 033222 005737 000042
5927 033226 001404
5928 033230 005037 033006
5929 033234 000137 032760
5930
5931 033240 000000
5932 033242 000137 003342
5933
5934
5935
5936
5937

      MOV      (PC)+,@(PC)+      ;;RESTORE COUNTER
$ENDCT: .WORD      1
      $EOPCT
      TYPE     ,65$              ;;TYPE ASCIZ STRING
      BR      64$              ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
      MOV      $PASS,-(SP)      ;;SAVE $PASS FOR TYPEOUT
      ;;TYPE PASS NUMBER
      TYPDS
      TYPE     ,67$              ;;GO TYPE--DECIMAL ASCII WITH SIGN
      BR      66$              ;;TYPE ASCIZ STRING
      ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
      MOV      $ERTTL,-(SP)     ;;SAVE $ERTTL FOR TYPEOUT
      ;;TOTAL NUMBER OF ERRORS
      TYPDS
      TYPE     , $CRLF           ;;GO TYPE--DECIMAL ASCII WITH SIGN
      ;;TYPE CARRIAGE RETURN, LINE FEED
      CLR     $ERTTL           ;;CLEAR ERROR TOTAL
$GET42: MOV      @#42,R0        ;;GET MONITOR ADDRESS
      BRQ    $DOAGN           ;;BRANCH IF NO MONITOR
      RESET
$ENDAD: JSR     PC,(R0)        ;;CLEAR THE WORLD
      NOP
      NOP
      NOP
      ;;GO TO MONITOR
      ;;SAVE ROOM
      ;;FOR
      ;;ACT11
$DOAGN:
      JMP     @(PC)+           ;;RETURN
$PTNAD: .WORD      TST1
$ENULL: .BYTE     -1,-1,0     ;;NULL CHARACTER STRING
      .EVEN

.SBTTL CONTROLLED HALT ROUTINE
;* THIS ROUTINE IS ENTERED WHEN A ^C IS ENTERED IN THE KEYBOARD.
;* IF NO MONITOR IS PRESENT THE PROGRAM HALTS ELSE THE PROGRAM
;* IS FORCED TO LAST PASS AND JUMPS TO END OF PASS.
CTRHLT: MOV      $BASE,R2      ;;SET '611 BASE
      MOV     #CCLR,RKCS1(R2) ;;CLEAR CONTROLLER
      TYPE   ,OPR005          ;;TYPE HALT MESSAGE
      MOV    #STACK,SP       ;;CLEAR STACK
      CLR   $ESCAPE          ;;CLEAR ESCAPE
      CLR   $ERFLG          ;;CLEAR ERROR FLAG
      TST   42              ;;TEST IF STAND ALONE
      BEQ   1$              ;;YES - SKIP
      CLR   $EOPCT          ;;ZERO PASS COUNT
      JMP   $EOP            ;;GO TO END OF PASS
1$: HALT
      JMP   START1          ;;GO TO RESTART IF CONTINUE

.SBTTL SCOPE HANDLER ROUTINE
;*****
;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT

```

```
5938      ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
5939      ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
5940      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
5941      ;*SW14=1      LOOP ON TEST
5942      ;*SW11=1      INHIBIT ITERATIONS
5943      ;*SW09=1      LOOP ON ERROR
5944      ;*SW08=1      LOOP ON TEST IN SWR<7:0>
5945      ;*CALL
5946      ;*      SCOPE      ;;SCOPE=IOT
5947
5948      $SCOPE:
5949      033246 104407      (KSWR      ;;TEST FOR CHANGE IN SOFT-SWR
5950      033250 032777 040000 145662 1$:      BIT      #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
5951      033256 001131      BNE      $OVER      ;;YES IF SW14=1
5952      ;*****START OF CODE FOR THE XOR TESTER*****
5953      033260 000416      $XTSTR: BR      6$      ;;IF RUNNING ON THE 'XCR' TESTER CHANGE
5954      ;THIS INSTRUCTION TO A 'NOP' (NOP-240)
5955      033262 013746 000004      MOV      @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
5956      033266 012737 033306 000004      MOV      #5$,@#ERRVEC      ;;SET FOR TIMEOUT
5957      033274 005737 177060      TST      @#177060      ;;TIME OUT ON XOR?
5958      033300 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
5959      033304 000500      BR      $$VLAD      ;;GO TO THE NEXT TEST
5960      033306 022626      5$:      CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
5961      033310 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
5962      033314 000440      BR      7$      ;;LOOP ON THE PRESENT TEST
5963      033316      6$:;*****END OF CODE FOR THE XOR TESTER*****
5964      033316 032777 000400 145614      BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
5965      033324 001421      BEQ      2$      ;;BR IF NO
5966      033326 005046      CLR      -(SP)      ;;CLEAR A TEMP. LOCATION
5967      033330 117716 145604      MOV      @SWR,(SP)      ;;PICKUP THE DESIRED TEST NUMBER
5968      033334 001414      BEQ      8$      ;;BRANCH IF BAD TEST NUMBER IN SWR
5969      033336 022716 000047      CMP      #47,(SP)      ;;CHECK THE NUMBER IN THE SWR
5970      033342 002411      BLT      8$      ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
5971      033344 011637 001102      MOV      (SP),$TSTNM      ;;UPDATE THE TEST NUMBER
5972      033350 005316      DEC      (SP)      ;;BACKUP BY ONE
5973      033352 006316      ASL      (SP)      ;;SCALE THE TEST NUMBER AS AN INDEX
5974      033354 062716 033560      ADD      #$$SW08TBL,(SP)      ;;FORM THE ADDRESS OF TEST POINTER
5975      033360 013637 001106      MOV      @((SP)+,$LPADR      ;;SET LOOP ADDRESS TO DESIRED TEST
5976      033364 000466      BR      $OVER      ;;GO LOOP ON THE TEST
5977      033366 005726      8$:      TST      (SP)+      ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
5978      033370 105737 001103      2$:      TSTB     $ERFLG      ;;HAS AN ERROR OCCURRED?
5979      033374 001421      BEQ      3$      ;;BR IF NO
5980      033376 123737 001115 001103      CMPB     $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
5981      033404 101015      BHI      3$      ;;BR IF NO
5982      033406 032777 001000 145524      BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
5983      033414 001404      BEQ      4$      ;;BR IF NO
5984      033416 013737 001110 001106      7$:      MOV      $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
5985      033424 000446      BR      $OVER
5986      033426 105037 001103      4$:      CLRB     $ERFLG      ;;ZERO THE ERROR FLAG
5987      033432 005037 00120C      CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
5988      033436 000415      BR      1$      ;;ESCAPE TO THE NEXT TEST
5989      033440 032777 004000 145472      3$:      BIT      #BIT11,@SWR      ;;INHIBIT ITERATIONS?
5990      033446 001011      BNE      1$      ;;BR IF YES
5991      033450 005737 001222      TST      $PASS      ;;IF FIRST PASS OF PROGRAM
5992      033454 001406      BEQ      1$      ;;      INHIBIT ITERATIONS
5993      033456 005237 001104      INC      $ICNT      ;;INCREMENT ITERATION COUNT
```

```

5994 033462 023737 001200 C01104      CMP      $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
5995 033470 002024          BGE      $OVER           ;;BR IF MORE ITERATION REQUIRED
5996 033472 012737 00C001 001104 1S:      MOV      #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
5997 033500 013737 033556 001200          MOV      $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
5998 033506 105237 001102          INCB    $TSTNM          ;;COUNT TEST NUMBERS
5999 033512 113737 001102 001220          MOV     $TSTNM,$TESTN    ;;SET TEST NUMBER IN APT MAILBOX
6000 033520 011637 001106          MOV     (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
6001 033524 011637 001110          MOV     (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
6002 033530 005037 001202          CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
6003 033534 112737 000001 001115          MOV     #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6004 033542 013777 001102 145377 $OVER:   MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER
6005 033550 013716 001106          MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
6006 033554 000002          RTI                    ;;FIXES PS
6007 033556 003720          $MXCNT: 2000.         ;;MAX. NUMBER OF ITERATIONS
6008 033560          $SWOBTBL:
6009 033560 004314          .WORD  TST1+2          ;;STARTING ADDRESS OF TEST 1
6010 033562 004546          .WORD  TST2+2          ;;STARTING ADDRESS OF TEST 2
6011 033564 005000          .WORD  TST3+2          ;;STARTING ADDRESS OF TEST 3
6012 033566 005232          .WORD  TST4+2          ;;STARTING ADDRESS OF TEST 4
6013 033570 005462          .WORD  TST5+2          ;;STARTING ADDRESS OF TEST 5
6014 033572 005712          .WORD  TST6+2          ;;STARTING ADDRESS OF TEST 6
6015 033574 006142          .WORD  TST7+2          ;;STARTING ADDRESS OF TEST 7
6016 033576 006426          .WORD  TST10+2         ;;STARTING ADDRESS OF TEST 10
6017 033600 006676          .WORD  TST11+2         ;;STARTING ADDRESS OF TEST 11
6018 033602 007146          .WORD  TST12+2         ;;STARTING ADDRESS OF TEST 12
6019 033604 007422          .WORD  TST13+2         ;;STARTING ADDRESS OF TEST 13
6020 033606 010072          .WORD  TST14+2         ;;STARTING ADDRESS OF TEST 14
6021 033610 010516          .WORD  TST15+2         ;;STARTING ADDRESS OF TEST 15
6022 033612 011056          .WORD  TST16+2         ;;STARTING ADDRESS OF TEST 16
6023 033614 011450          .WORD  TST17+2         ;;STARTING ADDRESS OF TEST 17
6024 033616 012044          .WORD  TST20+2         ;;STARTING ADDRESS OF TEST 20
6025 033620 012440          .WORD  TST21+2         ;;STARTING ADDRESS OF TEST 21
6026 033622 013034          .WORD  TST22+2         ;;STARTING ADDRESS OF TEST 22
6027 033624 013430          .WORD  TST23+2         ;;STARTING ADDRESS OF TEST 23
6028 033626 014216          .WORD  TST24+2         ;;STARTING ADDRESS OF TEST 24
6029 033630 014706          .WORD  TST25+2         ;;STARTING ADDRESS OF TEST 25
6030 033632 015302          .WORD  TST26+2         ;;STARTING ADDRESS OF TEST 26
6031 033634 015754          .WORD  TST27+2         ;;STARTING ADDRESS OF TEST 27
6032 033636 016426          .WORD  TST30+2         ;;STARTING ADDRESS OF TEST 30
6033 033640 017100          .WORD  TST31+2         ;;STARTING ADDRESS OF TEST 31
6034 033642 017552          .WORD  TST32+2         ;;STARTING ADDRESS OF TEST 32
6035 033644 020340          .WORD  TST33+2         ;;STARTING ADDRESS OF TEST 33
6036 033646 021126          .WORD  TST34+2         ;;STARTING ADDRESS OF TEST 34
6037 033650 021714          .WORD  TST35+2         ;;STARTING ADDRESS OF TEST 35
6038 033652 022404          .WORD  TST36+2         ;;STARTING ADDRESS OF TEST 36
6039 033654 023074          .WORD  TST37+2         ;;STARTING ADDRESS OF TEST 37
6040 033656 023564          .WORD  TST40+2         ;;STARTING ADDRESS OF TEST 40
6041 033660 024254          .WORD  TST41+2         ;;STARTING ADDRESS OF TEST 41
6042 033662 024770          .WORD  TST42+2         ;;STARTING ADDRESS OF TEST 42
6043 033664 025514          .WORD  TST43+2         ;;STARTING ADDRESS OF TEST 43
6044 033666 026240          .WORD  TST44+2         ;;STARTING ADDRESS OF TEST 44
6045 033670 027476          .WORD  TST45+2         ;;STARTING ADDRESS OF TEST 45
6046 033672 030746          .WORD  TST46+2         ;;STARTING ADDRESS OF TEST 46
6047 033674 031544          .WORD  TST47+2         ;;STARTING ADDRESS OF TEST 47
6048
6049

```

:SBTTL LOOP ON INTERNAL ERROR

```
6050
6051 033676 032777 001000 145234 SCOP1$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
6052 033704 001405 BEQ 5$ ;NO RETURN
6053 033706 105737 001103 TSTB $ERFLG ;CHECK IF ERROR OCCURED
6054 033712 001402 BEQ 5$ ;NO, RETURN
6055 033714 013716 001110 MOV $LPERR,(SP) ;GO BACK TO BEGINNING OF LOOP
6056 033720 000002 5$: RTI ;RETURN
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068 033722 104413
6069 033724 113737 001102 001220
6070 033732 042737 177400 001220
6071 033740 113700 001114
6072 033744 042700 177400
6073 033750 005300
6074 033752 006300
6075 033754 006300
6076 033756 006300
6077 033760 062700 001300
6078 033764 012037 034000
6079 033770 001404
6080 033772 104401 001211
6081 033776 104401
6082 034000 000000
6083 034002 012037 034016
6084 034006 001404
6085 034010 104401 001211
6086 034014 104401
6087 034016 000000
6088 034020 012001
6089 034022 001445
6090 034024 005004
6091 034026 012000
6092 034030 012002
6093 034032 104401 001211
6094 034036 112003
6095 034040 105720
6096 034042 005703
6097 034044 001416
6098 034046 005704
6099 034050 001004
6100 034052 013146
6101 034054 104402
6102 034056 005303
6103 034060 001403
6104 034062 104401 042425
6105 034066 000771

*****
: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.
*****
TYPERR: SAVREG
        MOVB $STNM,$TESTN ;GET TEST NUMBER
        BIC #177400,$TESTN ;CLEAR UNUSED BITS
        MOVB $ITEMB,R0 ;ENTER ERROR NUMBER
        BIC #177400,R0 ;CLEAR UNUSED BITS
        DEC R0 ;FORM INDEX FOR ERROR TABLE
        ASL R0
        ASL R0
        ASL R0
1$: ADD # $ERRTB,R0 ;FORM ADDRESS OF ERROR ENTRY
        MOV (R0)+,2$ ;GET EM POINTER
        BEQ 3$ ;BRANCH IF THERE ISN'T ONE
        TYPE , $CRLF ;TYPE CARRIAGE RETURN LINE FEED
        TYPE ;TYPE ERROR MESSAGE (EM)
2$: .WORD 0 ;EM POINTER GOES HERE
3$: MOV (R0)+,4$ ;GET DH POINTER
        BEQ 5$ ;BRANCH IF THERE ISN'T ONE
        TYPE , $CRLF ;TYPE CR-LF
        TYPE ;TYPE DATA HEADER
4$: .WORD 0 ;DH POINTER GOES HERE
5$: MOV (R0)+,R1 ;GET DT POINTER
        BEQ 20$ ;BRANCH IF THERE ARE NONE
        CLR R4 ;RESET INDENT SWITCH
        MOV (R0)+,R0 ;GET DF POINTER
        MOV (R0)+,R2 ;STORE NUMBER OF DH'S
10$: MOVB (R0)+,R3 ;GET & STORE NUMBER OF DATA WORDS
        TSTB (R0)+ ;BUMP PAST FORMAT WORD
        TST R3 ;TEST IF ANY DATA FOR THIS HEADER
        BEQ 14$ ;NO - SKIP DATA PRINT
        TST R4 ;CHECK IF INDENT WORDS
        BNE 12$ ;YES, GO INDENT
11$: MOV @ (R1)+,-(SP) ;PUT FIRST DATA WORD ON STACK
        TYPOC ;TYPE IT
        DEC R3 ;MORE DATA WORDS
        BEQ 13$ ;NO-BRANCH
12$: TYPE ,SPACE2 ;TYPE SEPARATORS
        BR 11$ ;LOOP
```

```

6106 034070 10440 001211 13$: TYPE ,SCLF ;TYPE <CR><LF>
6107 034074 005710 TST (R0) ;CHECK IF NEXT HEADER AVAILBLE
6108 034076 001401 BEQ 14$ ;NO, DO NOT CHANGE INDENT
6109 034100 005104 COM R4 ;CHANGE INDENT
6110 034102 005302 14$: DEC R2 ;MORE DH'S?
6111 034104 003414 BLE 20$ ;NO-BRANCH
6112 034106 012037 034126 15$: MOV (R0)+,18$ ;GET NEXT DH POINTER
6113 034112 001751 BEQ 10$ ;IF NO HEADER GET DATA
6114 034114 005704 TST R4 ;INDENT?
6115 034116 001402 BEQ 17$ ;NO-BRANCH
6116 034120 104401 042425 TYPE ,SPACE2 ;INDENT
6117 034124 104401 17$: TYPE ;TYPE DH
6118 034126 000000 18$: .WORD 0 ;DH POINTER GOES HERE
6119 034130 104401 001211 TYPE ,SCLF
6120 034134 000740 BR 10$ ;LOOP
6121 034136 104414 20$: RESREG
6122 034140 005237 003250 INC ERRCNT ;INCREMENT ERROR COUNT
6123 034144 032777 010000 144766 BIT #SW12,@SWR ;CHECK IF ABORT AFTER 20 ERRORS
6124 034152 001421 BEQ 25$ ;NO, RETURN
6125 034154 022737 000024 003250 CMP #20.,ERRCNT ;CHECK IF EROR THRESHOLD EXCEEDED
6126 034152 103015 BHIS 25$ ;NO, RETURN
6127 034164 104401 042430 TYPE ,ABORT ;TYPE 'PROGRAM HAS BEEN ABORTED BECAUSE
6128 ; ERROR THRESHOLD EXCEEDED'
6129 034170 005737 000042 TST 42 ;CHECK IF IN CHAIN MODE
6130 034174 001407 BEQ 30$ ;NO, HALT
6131 034176 012737 000001 033006 MOV #1,$EOPCT ;FORCE END OF PASS COUNT TO ONE FOR ABORT
6132 034204 012706 001100 MOV #STACK,SP ;INITIALIZE STACK
6133 034210 000137 032760 JMP $EOP ;BRING IN NEXT PROGRAM IN CHAIN
6134 034214 000000 30$: HALT
6135 034216 000207 25$: RTS PC
6136
6137 .SBTTL GENERATE HEADERS FOR SECTOR, TRACK, AND CYLINDER INCREMENTS
6138
6139 034220 013737 003300 003302 INCHDR: MOV CYLN,HEADER ;LOAD HEADER WORD 1
6140 034226 005037 003304 CLR HEADER+2 ;CALCULATE HEADER WORD 2
6141 034232 113737 003277 003305 MOV# TRACK,HEADER+3
6142 034240 006237 003304 ASR HEADER+2
6143 034244 006237 003304 ASR HEADER+2
6144 034250 006237 003304 ASR HEADER+2
6145 034254 153737 003276 003304 BISB SECTOR,HEADER+2
6146 034262 052737 140000 003304 BIS #140000,HEADER+2
6147 034270 013746 003302 MOV HFADER,-(SP) ;GENERATE XOR
6148 034274 013737 003304 003306 MOV HEADER+2,HEADER+4
6149 034302 043737 003302 003306 BIC HEADER,HEADER+4
6150 034310 043716 003304 BIC HEADER+2,(SP)
6151 034314 052637 003306 BIS (SP)+,HEADER+4
6152 034320 013737 003300 003220 MOV CYLN,E.DCYL ;INCREMENT DISK ADDRESS
6153 034326 013737 003276 003206 MOV SECTOR,E.DA
6154 034334 105237 003206 INCB E.DA
6155 034340 122737 000026 003206 CMPB #26,E.DA
6156 034346 001014 BNE 10$
6157 034350 105037 003206 CLRB E.DA
6158 034354 105237 003207 INCB E.DA+1
6159 034360 122737 000003 003207 CMPB #3,E.DA+1
6160 034366 001004 BNE 10$
6161 034370 005037 003206 CLR E.DA

```

```
6162 034374 005237 003220
6163 034400 000207
6164
6165
6166
6167 034402 005003
6168 034404 013701 003220
6169 034410 001406
6170 034412 012700 000020
6171 034416 006101
6172 034420 006003
6173 034422 005300
6174 034424 001374
6175 034426 010337 003226
6176 034432 013746 003206
6177 034436 001410
6178 034440 042716 000377
6179 034444 001403
6180 034446 006216
6181 034450 006216
6182 034452 006216
6183 034454 153716 003206
6184 034460 012601
6185 034462 032737 010000 003200
6186 034470 001402
6187 034472 052701 001000
6188 034476 005003
6189 034500 005701
6190 034502 001406
6191 034504 012700 000020
6192 034510 006101
6193 034512 006003
6194 034514 005300
6195 034516 001374
6196 034520 010337 003230
6197 034524 000207
6198
6199
6200
6201
6202 034526 005737 003252
6203 034532 001410
6204 034534 032737 000001 003266
6205 034542 001410
6206 034544 005037 003272
6207 034550 000241
6208 034552 000410
6209
6210 034554 032737 000001 003266
6211 034562 001770
6212 034564 012737 000001 003272
6213 034572 000261
6214 034574 006037 003270
6215 034600 006037 003266
6216 034604 005737 003272
6217 034610 001422

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
BEQ 6$ ;CHECK TRACK = 0
ASR (SP) ;SHIFT HEAD 3 BITS RIGHT
ASR (SP)
ASR (SP)
6$: BISB E.DA,(SP) ;OR IN SECTOR BITS
7$: MOV (SP)+,R1
BIT #CFMT,E.CS1 ;CHECK FORMAT = 24 SECTOR
BEQ 8$ ;NO, CALCULATE SECOND WORD
BIS #BIT9,R1 ;SET FORMAT BIT
8$: CLR R3 ;CLEAR EXPECTED SECOND WORD
TST R1 ;CHECK IF WORD = 0
BEQ 15$ ;YES, RETURN
MOV #16.,R0 ;LOAD SHIFT COUNT
10$: ROL R1
ROR R3
DEC R0
BNE 10$
15$: MOV R3,E.MR3 ;LOAD SECOND WORD
RTS PC ;RETURN

:*****
.SBTTL GENERATE ECC WORD
ECCGEN: TST P1.BIT ;CHECK IF 1
BEQ 3$ ;NO, CHECK IF BIT 31 = 1
BIT #1,ECCHI ;NO, CHECK IF BIT 31 = 1
BEQ 5$ ;NO, SET ECC XORED BIT
1$: CLR ECCXOR ;CLEAR ECC XORED BIT
CLC ;CLEAR CARRY FLOP
BR 7$ ;SHIFT IN ECC BIT
3$: BIT #1,ECCHI ;CHECK IF BIT 31 - 1
BEQ 1$ ;NO, CLEAR ECC XORED BIT
5$: MOV #1,ECCXOR ;SET ECC XOR
SEC ;SET CARRY FLOP
7$: ROR ECCLO
ROR ECCHI
TST ECCXOR ;CHECK IF XOR NEEDED
BEQ 10$ ;NO, RETURN
```

```
6218 034612 012746 020020      MOV      #BIT13!BIT4,-(SP) ;DO XOP OF ECC BITS
6219 034616 043716 003270      BIC      ECCL0,(SP)      ; 0, 2, 11, 21, 23
6220 034622 042737 020020 003270  BIC      #BIT13!BIT4,ECCL0
6221 034630 052637 003270      BIS      (SP)+,ECCL0
6222 034634 012746 002400      MOV      #BIT10!BIT8,-(SP)
6223 034640 043716 003266      BIC      ECCHI,(SP)
6224 034644 042737 002400 003266  BIC      #BIT10!BIT8,ECCHI
6225 034652 052637 003266      BIS      (SP)+,ECCHI
6226 034656 013737 003266 003234 10$:    MOV      ECCHI,E.ECPT ;STORE ECC PATTERN
6227 034664 042737 174000 003234  BIC      #174000,E.ECPT ;MASK UNSEEN BITS
6228 034672 000207      RTS      PC ;RETURN
6229                                     ;*****
6230                                     ;SBTTL SIMULATE ONE BIT WRITE IN MAINTENANCE MODE
6231
6232 034674 052737 042040 003224  WRTBIT: BIS      #DMD!MEWD.WRTGAT,E.MR1 ;INITIALIZE
6233 034702 042737 114000 003224  BIC      #RDGATE!PCA!PCD,E.MR1 ; EXPECTED MR1
6234 034710 005737 003254      TST      PR.BIT ;CHECK IF ONE
6235 034714 001122      BNE      20$ ;YES, SIMULATE A ONE
6236 034716 005737 003256      TST      M1.BIT ;CHECK IF PREVIOUS ONE
6237 034722 001023      BNE      10$ ;YES, NO TRANSITION
6238 034724 042737 002000 003224  BIC      #MEWD,E.MR1 ;INDICATE TRANSITION
6239 034732 005737 003252      TST      P1.BIT ;CHECK IF NEXT BIT = 1
6240 034736 001007      BNE      5$ ;ES, CHECK FOR PRECOMP ADVANCE
6241 034740 005737 003260      TST      M2.BIT ;CHECK FOR PRECOMP. DELAY
6242 034744 001412      BEQ      10$ ;NO, CLOCK IN ZERO
6243 034746 052737 010000 003224  BIS      #PCD,E.MR1 ;SET PRECOMP. DELAY
6244 034754 000406      BR       10$ ;CLOCK IN ZERO
6245
6246 034756 005737 003260      5$:    TST      M2.BIT ;CHECK FOR PRECOMP. ADVANCE
6247 034762 001003      BNE      10$ ;CLOCK IN ZERO
6248 034764 052737 074000 003224  BIS      #PCA,E.MR1 ;SET PRECOMPENSATION ADVANCE
6249 034772 012762 000440 000026 10$:    MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6250 035000 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6251 035006 016237 000026 003164  MOV      RKMR1(R2),T.MR1 ;STORE MR1
6252 035014 023737 003164 003224  CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6253 035022 001416      BEQ      15$ ;YES, CLOCK IN REST OF BIT
6254 035024 012737 035044 001202  MOV      #13$, $ESCAPE ;LOAD ESCAPE
6255 035032 012737 051447 001312  MOV      #EMW2,EMW+2 ;LOAD ERROR MESSAGE
6256 035040 011646      MOV      (SP),-(SP) ;STORE RETURN
6257 035042 000207      RTS      PC ;REPORT ERROR
6258 035044 032777 001000 144066 13$:    BIT      #SW9,@SWR ;CHECK IF LOOP ON ERROR
6259 035052 001402      BEQ      15$ ;NO, CONTINUE
6260 035054 000137 035406      JMP      63$ ;GO LOOP ON ERROR
6261
6262 035060 052737 002000 003224 15$:    BIS      #MEWD,E.MR1 ;RESET TRANSITION INDICATION
6263 035066 012762 000440 000026  MOV      #DMD.MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6264 035074 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6265 035102 016237 000026 003164  MOV      RKMR1(R2),T.MR1 ;STORE MR1
6266 035110 023737 003164 003224  CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6267 035116 001414      BEQ      18$ ;YES, RETURN
6268 035120 012737 035140 001202  MOV      #17$, $ESCAPE ;LOAD ESCAPE
6269 035126 012737 051537 001312  MOV      #EMW4,EMW+2 ;LOAD ERROR MESSAGE
6270 035134 011646      MOV      (SP),-(SP) ;SAVE RETURN
6271 035136 000207      RTS      PC ;REPORT ERROR
6272
6273 035140 032777 001000 143772 17$:    BIT      #SW9,@SWR ;CHECK IF LOOP ON ERROR
```

```
6274 035146 001117
6275 035150 005037 001202
6276 035154 062716 000002
6277 035160 000207
6278
6279
6280 035162 005737 003252
6281 035166 001007
6282 035170 005737 003256
6283 035174 001412
6284 035176 052737 004000 003224
6285 035204 000406
6286
6287 035206 005737 003256
6288 035212 001003
6289 035214 052737 010000 003224
6290 035222 012762 000440 000026
6291 035230 012762 000040 000026
6292 035236 016237 000026 003164
6293 035244 023737 003164 003224
6294 035252 001414
6295 035254 012737 035274 001202
6296 035262 012737 051447 001312
6297 035270 011646
6298 035272 000207
6299
6300 035274 032777 001000 143636
6301 035302 001041
6302 035304 042737 002000 003224
6303 035312 012762 000440 000026
6304 035320 012762 000040 000026
6305 035326 016237 000026 003164
6306 035334 023737 003164 003224
6307 035342 001414
6308 035344 012737 035364 001202
6309 035352 012737 051537 001312
6310 035360 011646
6311 035362 000207
6312
6313 035364 032777 001000 143546
6314 035372 001005
6315 035374 005037 001202
6316 035400 062716 000002
6317 035404 000207
6318
6319 035406 005037 001202
6320 035412 012706 001100
6321 035416 000177 143466
6322
6323
6324
6325
6326 035422 105737 003254
6327 035426 001024
6328 035430 105737 003256
6329 035434 001404

18$: BNE 63$
CLR $ESCAPE ;CLEAR ESCAPE
ADD #2,(SP) ;ADJUST ESCAPE
RTS PC ;RETURN

20$: TST P1.BIT ;CHECK IF NEXT BIT A ONE
BNE 30$ ;YES, CHECK FOR PRECOMP. DELAY
TST M1.BIT ;CHECK FOR PRECOMP. ADVANCE
BEQ 40$ ;NO, CHECK MR1
BIS #PCA,E.MR1 ;SET PRECOMP ADVANCE
BR 40$ ;CHECK MR1

30$: TST M1.BIT ;CHECK FOR PRECOMP. DELAY
BNE 40$ ;NO, CHECK MR1
BIS #PCD,E.MR1 ;SET PRECOMP. DELAY

40$: MOV #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
MOV #DMD,RKMR1(R2)
MOV RKMR1(R2),T.MR1 ;STORE MR1
CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
BFQ 45$ ;YES, CLOCK IN REST OF BIT
MOV #42,$ESCAPE ;LOAD ESCAPE
MOV #EMW2,EMW+2 ;LOAD ERROR MESSAGE
MOV (SP),-(SP) ;STORE RETURN
RTS PC ;REPORT ERROR

42$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
BNE 63$ ;YES, LOOP ERROR

45$: BIC #MEWD,E.MR1 ;INDICATE A TRANSITION
MOV #DMD!MCLK,RKMR1(R2) ;CLOCK TRANSITION
MOV #DMD,RKMR1(R2)
MOV RKMR1(R2),T.MR1 ;STORE MR1
CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
BEQ 50$ ;YES, RETURN
MOV #47,$ESCAPE ;LOAD ESCAPE
MOV #EMW4,EMW+2 ;LOAD ERROR MESSAGE
MOV (SP),-(SP) ;SAVE RETURN
RTS PC ;REPORT ERROR

47$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
BNE 63$ ;YES, REPORT ERROR

50$: CLR $ESCAPE ;LOAD ESCAPE
ADD #2,(SP) ;ADJUST RETURN
RTS PC ;RETURN

63$: CLR $ESCAPE ;CLEAR ESCAPE
MOV #STACK,SP ;FORCE STACK
JMP @SLPERR ;JUMP TO LOOP ADDRESS

*****
.SBTTL SIMULATE ONE BIT OF READ DATA IN MAINTENANCE MODE

RDBIT: TSTB PR.BIT ;CHECK IF ONE
BNE 10$ ;YES, SIMULATE ONE
TSTB M1.BIT ;CHECK IF PREVIOUS ONE
BEQ 4$ ;INSERT TRANSITION
```



```
6386 035746 005237 001214          INC      $MSGTYPE      ;; TELL APT TO TAKE ERROR
6387 035752 105037 035776          12$: CLR3   $FFLG      ;; CLEAR FATAL FLAG
6388 035756 105037 035775          CLR3   $LFLG      ;; CLEAR LOG FLAG
6389 035762 105037 035774          CLR3   $MFLG      ;; CLEAR MESSAGE FLAG
6390 035766 012601          MOV     (SP)+,R1    ;; POP STACK INTO R1
6391 035770 012600          MOV     (SP)+,R0    ;; POP STACK INTO R0
6392 035772 000207          RTS     PC          ;; RETURN
6393 035774          000          $MFLG: .BYTE 0      ;; MESSG. FLAG
6394 035775          000          $LFLG: .BYTE 0      ;; LOG FLAG
6395 035776          000          $FFLG: .BYTE 0      ;; FATAL FLAG
6396          036000          .EVEN
6397          000200          APTSIZE=200
6398          000001          APTENV=001
6399          000100          APTSPool=100
6400          000040          APTCSUP=040
6401          .SBTTL ERROR HANDLER ROUTINE
6402
6403          ;;*****
6404          ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
6405          ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
6406          ;;*AND GO TO TYPERR ON ERROR
6407          ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6408          ;;*SW15=1      HALT ON ERROR
6409          ;;*SW13=1      INHIBIT ERROR TYPEOUTS
6410          ;;*SW10=1     BELL ON ERROR
6411          ;;*SW09=1     LOOP ON ERROR
6412          ;;*CALL
6413          ;;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
6414
6415          $ERROR:
6416 036000 104407          CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
6417 036002 105237 001103          7$: INCB   $ERFLG      ;; SET THE ERROR FLAG
6418 036006 001775          BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
6419 036010 013777 001102 143124          MOV     $TSTNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
6420 036016 032777 002000 143114          BIT     #BIT10,@SWR    ;; BELL ON ERROR?
6421 036024 001402          BEQ      1$          ;; NO - SKIP
6422 036026 104401 001204          TYPE   , $BELL      ;; RING BELL
6423 036032 005237 001112          1$: INC     $ERTTL     ;; COUNT THE NUMBER OF ERRORS
6424 036036 011637 001116          MOV     (SP), $ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
6425 036042 162737 000002 001116          SUB     #2, $ERRPC
6426 036050 117737 143042 001114          MOVB   @ $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
6427 036056 032777 020000 143054          BIT     #BIT13,@SWR    ;; SKIP TYPEOUT IF SET
6428 036064 001004          BNE     20$         ;; SKIP TYPEOUTS
6429 036066 004737 033722          JSR    PC, TYPERR    ;; GO TO USER ERROR ROUTINE
6430 036072 104401 001211          TYPE   , $CRLF
6431 036076
6432 036076 122737 000001 001234          20$: CMPB   #APTENV, $ENV  ;; RUNNING IN APT MODE
6433 036104 001007          BNE     2$          ;; NO, SKIP APT ERROR REPORT
6434 036106 113737 001114 036120          MOVB   $ITEMB, 21$   ;; SET ITEM NUMBER AS ERROR NUMBER
6435 036114 004737 03555C          JSR    PC, $ATY4     ;; REPORT FATAL ERROR TO APT
6436 036120          000          21$: .BYTE 0
6437 036121          000          .BYTE 0
6438 036122 000777          22$: BR     22$          ;; APT ERROR LOOP
6439 036124 005777 143010          2$: TST   @SWR
6440 036130 100002          BPL    3$
6441 036132 000000          HALT
```

```
6442 036134 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
6443 036136 032777 001000 142774 3$: BIT #BIT09,@SWR  ;;LOOP ON ERROR SWITCH SET?
6444 036144 001402          BEQ 4$        ;;BR IF NO
6445 036146 013716 001110          MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
6446 036152 005737 001202          4$: TST $ESCAPE  ;;CHECK FOR AN ESCAPE ADDRESS
6447 036156 001402          BEQ 5$        ;;BR IF NONE
6448 036160 013716 001202          MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
6449 036164          5$:
6450 036164 022737 033150 000042  CMP #SENDAD,@#42  ;;ACT-11 AUTO-ACCEPT?
6451 036172 001001          BNE 6$        ;;BRANCH IF NO
6452 036174 000000          HALT          ;;YES
6453 036176          6$:
6454 036176 000002          RTI          ;;RETURN
6455
6456          .SBTTL TYPE ROUTINE
6457
6458          ;:*****
6459          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
6460          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
6461          ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
6462          ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
6463          ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
6464          ;*
6465          ;*CALL:
6466          ;*1) USING A TRAP INSTRUCTION
6467          ;* TYPE ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
6468          ;*OR
6469          ;* TYPE
6470          ;* MESADR
6471          ;*
6472
6473 036200 105737 001157  $TYPE: TSTB $TPFLG          ;;IS THERE A TERMINAL?
6474 036204 100002          BPL 1$        ;;BR IF YES
6475 036206 000000          HALT          ;;HALT HERE IF NO TERMINAL
6476 036210 000430          BR 3$        ;;LEAVE
6477 036212 010046          1$: MOV RO,-(SP)  ;;SAVE RO
6478 036214 017600 000002  MOV @2(SP),RO  ;;GET ADDRESS OF ASCIZ STRING
6479 036220 122737 000001 001234  CMPB #APTENV,$ENV  ;;RUNNING IN APT MODE
6480 036226 001011          BNE 62$      ;;NO,GO CHECK FOR APT CONSOLE
6481 036230 132737 000100 001235  BITB #APTSPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
6482 036236 001405          BEQ 62$      ;;NO,GO CHECK FOR CONSOLE
6483 036240 010037 036250  MOV RO,61$    ;;SETUP MESSAGE ADDRESS FOR APT
6484 036244 004737 035540  JSR PC,$ATY3  ;;SPOOL MESSAGE TO APT
6485 036250 000000          61$: .WORD 0  ;;MESSAGE ADDRESS
6486 036252 132737 000040 001235  62$: BITB #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
6487 036260 001003          BNE 60$      ;;YES,SKIP TYPE OUT
6488 036262 112046          2$: MOVB (RO)+,-(SP)  ;;PUSH CHARACTER TO BE TYPED ONTO STACK
6489 036264 001005          BNE 4$        ;;BR IF IT ISN'T THE TERMINATOR
6490 036266 005726          TST (SP)+    ;;IF TERMINATOR POP IT OFF THE STACK
6491 036270 012600          60$: MOV (SP)+,RO  ;;RESTORE RO
6492 036272 062716 000002  3$: ADD #2,(SP)  ;;ADJUST RETURN PC
6493 036276 000002          RTI          ;;RETURN
6494 036300 122716 000011  4$: CMPB #HT,(SP)  ;;BRANCH IF <HT>
6495 036304 001430          BEQ 8$        ;;
6496 036306 122716 000200  CMPB #CRLF,(SP)  ;;BRANCH IF NOT <CRLF>
6497 036312 001006          BNE 5$
```

```
6498 036314 005726          TST      (SP)+          ;;POP <CR><LF> EQUIV
6499 036316 104401          TYPE                                ;;TYPE A CR AND LF
6500 036320 001211          $CRLF
6501 036322 105037 036530    CLRB     $CHARCNT      ;;CLEAR CHARACTER COUNT
6502 036326 000755          BR       2$            ;;GET NEXT CHARACTER
6503 036330 004737 036412    5$:     JSR     PC,$TYPEPC ;;GO TYPE THIS CHARACTER
6504 036334 123726 001156    6$:     CMPB   $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
6505 036340 001350          BNE     2$            ;;IF NO GO GET NEXT CHAR.
6506 036342 013746 001154    MOV     $NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
6507                                ;;AND THE NULL CHAR.
6508 036346 105366 000001    7$:     DECB   1(SP)    ;;DOES A NULL NEED TO BE TYPED?
6509 036352 002770          BLT     6$            ;;BR IF NO--GO POP THE NULL OFF OF STACK
6510 036354 004737 036412    JSR     PC,$TYPEPC    ;;GO TYPE A NULL
6511 036360 105337 036530    DECB   $CHARCNT      ;;DO NOT COUNT AS A COUNT
6512 036364 000770          BR      7$            ;;LOOP
6513
6514                                ;HORIZONTAL TAB PROCESSOR
6515
6516 036366 112716 000040    8$:     MOVB   #' ,(SP) ;;REPLACE TAB WITH SPACE
6517 036372 004737 036412    9$:     JSR     PC,$TYPEPC ;;TYPE A SPACE
6518 036376 132737 000007 036530    BITB   #7,$CHARCNT   ;;BRANCH IF NOT AT
6519 036404 001372          BNE     9$            ;;TAB STOP
6520 036406 005726          TST     (SP)+          ;;POP SPACE OFF STACK
6521 036410 000724          BR      2$            ;;GET NEXT CHARACTER
6522 036412
6523 036412 105777 142526    $TYPEPC: TSTB   @STKS          ;;CHAR IN KYBD BUFFER?
6524 036416 100022          BPL     10$           ;;BR IF NOT
6525 036420 017746 142522    MOV     @STKB,-(SP)   ;;GET CHAR
6526 036424 042716 177600    BIC     #177600,(SP) ;;STRIP EXTRANEIOUS BITS
6527 036430 122716 000023    CMPB   #$XOFF,(SP)  ;;WAS CHAR XOFF
6528 036434 001012          BNE     102$          ;;BR IF NOT
6529 036436
6530 036436 105777 142502    101$:   TSTB   @STKS          ;;WAIT FOR CHAR
6531 036442 100375          BPL     101$          ;;
6532 036444 117716 142476    MOVB   @STKB,(SP)   ;;GET CHAR
6533 036450 042716 177600    BIC     #177600,(SP) ;;STRIP IT
6534 036454 122716 000021    CMPB   #$XON,(SP)  ;;WAS IT XON?
6535 036460 001366          BNE     101$          ;;BR IF NOT
6536 036462
6537 036462 005726          102$:   TST     (SP)+          ;;FIX STACK
6538 036464
6539 036464 105777 142460    10$:     TSTB   @STPS          ;;WAIT UNTIL PRINTER IS READY
6540 036470 100375          BPL     10$           ;;
6541 036472 116677 000002 142452    MOVB   2(SP),@STPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
6542
6543 036500 122766 000015 000002    CMPB   #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
6544 036506 001003          BNE     1$            ;;BRANCH IF NO
6545 036510 105037 036530    CLRB   $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
6546 036514 000406          BR     $TYPEPC      ;;EXIT
6547 036516 122766 000012 000002    1$:     CMPB   #LF,2(SP) ;;IS CHARACTER A LINE FEED?
6548 036524 001402          BEQ     $TYPEPC     ;;BRANCH IF YES
6549 036526 105227          INCB   (PC)+        ;;COUNT THE CHARACTER
6550 036530 000000    $CHARCNT: .WORD 0   ;;CHARACTER COUNT STORAGE
6551 036532 000207    $TYPEPC: RTS      PC
6552
6553                                .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
```

6554
6555
6556
6557
6558
6559
6560
6561
6562
6563
6564
6565
6566
6567
6568
6569
6570
6571
6572
6573
6574
6575
6576
6577

```

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

```

6578 036534 017646 000000
6579 036540 116637 000001 036757
6580 036546 112637 036761
6581 036552 062716 000002
6582 036556 000406
6583 036560 112737 000001 036757
6584 036566 112737 000006 036761
6585 036574 112737 000005 036756
6586 036602 010346
6587 036604 010446
6588 036606 010546
6589 036610 113704 036761
6590 036614 005404
6591 036616 062704 000006
6592 036622 110437 036760
6593 036626 113704 036757
6594 036632 016605 000012
6595 036636 005003
6596 036640 006105
6597 036642 000404
6598 036644 006105
6599 036646 006105
6600 036650 006105
6601 036652 010503
6602 036654 006103
6603 036656 105337 036760
6604 036662 100016
6605 036664 042703 177770
6606 036670 001002
6607 036672 005704
6608 036674 001403
6609 036676 005204

```

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

```



```

6666 037056 100407
6667 037060 106316
6668 037062 103003
6669 037064 116663 000001 177777
6670 037072 052702 000060
6671 037076 052702 000040
6672 037102 110223
6673 037104 005720
6674 037106 020027 000010
6675 037112 002746
6676 037114 003002
6677 037116 010502
6678 037120 000764
6679 037122 105726
6680 037124 100003
6681 037126 116663 177777 177776
6682 037134 105013
6683 037136 012605
6684 037140 012603
6685 037142 012602
6686 037144 012601
6687 037146 012600
6688 037150 104401 037176
6689 037154 016666 000002 000004
6690 037162 012616
6691 037164 000002
6692 037166 023420
6693 037170 001750
6694 037172 000144
6695 037174 000012
6696 037176 000004
6697
6698
6699
6700
6701 037206 000000
6702 037210 000000
6703 037212 000000
6704 037214 000001
6705 037215
6706 037216
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716 037216 005037 037206
6717 037222 012737 037214 037210
6718 037230 013737 037210 037212
6719 037236 012737 037266 000060
6720 037244 012737 000200 000062
6721 037252 005777 141670

```

```

      BMI      7$          ;;BR IF YES
5$:    ASLB    (SP)        ;;MSD?
      BCC     6$          ;;BR IF NO
      MOVB    1(SP),-1(R3) ;;YES--SET THE SIGN
6$:    BIS     #'0,R2     ;;MAKE THE BCD DIGIT ASCII
7$:    BIS     #' ,R2     ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
      MOVB    R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
      TST     (R0)+      ;;JUST INCREMENTING
      CMP     R0,#10     ;;CHECK THE TABLE INDEX
      BLT     2$          ;;GO DO THE NEXT DIGIT
      BGT     8$          ;;GO TO EXIT
      MOV     R5,R2      ;;GET THE LSD
      BR      6$          ;;GO CHANGE TO ASCII
8$:    TSTB    (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
      BPL     9$          ;;BR IF NO
9$:    MOVB    -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
      CLRB    (R3)       ;;SET THE TERMINATOR
      MOV     (SP)+,R5    ;;POP STACK INTO R5
      MOV     (SP)+,R3    ;;POP STACK INTO R3
      MOV     (SP)+,R2    ;;POP STACK INTO R2
      MOV     (SP)+,R1    ;;POP STACK INTO R1
      MOV     (SP)+,R0    ;;POP STACK INTO R0
      TYPE    ,SDBLK     ;;NOW TYPE THE NUMBER
      MOV     2(SP),4(SP) ;;ADJUST THE STACK
      MOV     (SP)+,(SP)
      RTI              ;;RETURN TO USER

$DTBL: 10000.
      1000.
      100.
      10.

$SDBLK: .BLKW 4
.SB*TL TTY INPUT ROUTINE

;*****
;ENABL  LSB
$TKCNT: .WORD 0          ;;NUMBER OF ITEMS IN QUEUE
$TKQIN: .WORD 0          ;;INPUT POINTER
$TKQOUT: .WORD 0         ;;OUTPUT POINTER
$TKQSRT: .BLKB 1        ;;TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
;
;*CALL:
;*      JSR    PC,$TKINT
;*      RETURN
;
$TKINT: CLR    $TKCNT     ;;CLEAR COUNT OF ITEMS IN QUEUE
      MOV    # $TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
      MOV    $TKQIN,$TKQOUT ;;QUEJE INTO THE INPUT & OUTPUT POINTERS.
      MOV    # $TKSRV,@ $TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
      MOV    #200,@ $TKVEC+2 ;;'BR' LEVEL 4
      TST   @ $TKB       ;;CLEAR DONE FLAG

```

```
6722 037256 012777 000100 141660      MOV    #100,@STKS      ;;ENABLE TTY KEYBOARD INTERRUPT
6723 037264 000207                RTS    PC              ;;RETURN TO CALLER
6724
6725      ;*TK SERVICE ROUTINE
6726      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
6727      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
6728      ;*IT IN THE QUEUE.
6729      ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
6730      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRL^C)
6731
6732 037266 117746 141654      $TKSRV: MOVB   @STKB,-(SP)      ;;PICKUP THE CHARACTER
6733 037272 042716 177600      BIC    #^C(177),(SP)      ;;STRIP THE JUNK
6734 037276 021627 000021      CMP    (SP),#$XON        ;;IS IT A RANDOM XON?
6735 037302 001002                BNE    30$              ;;BRANCH IF NO
6736 037304 005726                TST    (SP)+            ;;CLEAN RANDOM XON OFF STACK
6737 037306 000002                RTI                    ;;RETURN
6738 037310
6739 037310 021627 000003      30$:    CMP    (SP),#3        ;;IS IT A CONTROL C?
6740 037314 001007                BNE    1$              ;;BRANCH IF NO
6741 037316 104401 040414      TYPE   ,SCNTLC          ;;TYPE A CONTROL-C (^C)
6742 037322 004737 037216      JSR    PC,$TKINT        ;;INIT THE KEYBOARD
6743 037326 005726                TST    (SP)+            ;;CLEAN UP STACK
6744 037330 000137 033170      JMP    CTRL^C           ;;CONTROL C RESTART
6745 037334 021627 000007      1$:    CMP    (SP),#7        ;;IS IT A CONTROL G?
6746 037340 001004                BNE    2$              ;;BRANCH IF NO
6747 037342 022737 000176 001140      CMP    #SWREG,SWR       ;;IS SOFT-SWR SELECTED?
6748 037350 001500                BEQ    6$              ;;GO TO SWR CHANGE
6749
6750 037352
6751 037352 022737 000001 037206      2$:    CMP    #1,$TKCNT      ;;IS THE QUEUE FULL?
6752 037360 001004                BNE    3$              ;;BRANCH IF NO
6753 037362 104401 001204      TYPE   ,SBELL          ;;RING THE TTY BELL
6754 037366 005726                TST    (SP)+            ;;CLEAN CHARACTER OFF OF STACK
6755 037370 000451                BR     5$              ;;EXIT
6756 037372 021627 000023      3$:    CMP    (SP),#23      ;;IS IT A CONTROL-S?
6757 037376 001021                BNE    32$             ;;BRANCH IF NO
6758 037400 005077 141540      CLR    @STKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
6759 037404 005726                TST    (SP)+            ;;CLEAN CHAR OFF STACK
6760 037406 105777 141532      31$:   TSTB   @STKS          ;;WAIT FOR A CHAR
6761 037412 100375                BPL    31$            ;;LOOP UNTIL ITS THERE
6762 037414 117746 141526      MOVB   @STKB,-(SP)      ;;GET THE CHARACTER
6763 037420 042716 177600      BIC    #^C(177),(SP)    ;;MAKE IT 7-BIT ASCII
6764 037424 022627 000021      CMP    (SP)+,#21        ;;IS IT A CONTROL-Q?
6765 037430 001366                BNE    31$            ;;BRANCH IF NO
6766 037432 012777 000100 141504      MOV    #100,@STKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
6767 037440 000002                RTI                    ;;RETURN
6768 037442 005237 037206      32$:   INC    $TKCNT         ;;COUNT THIS CHARACTER
6769 037446 021627 000140      CMP    (SP),#140       ;;IS IT UPPER CASE?
6770 037452 002405                BLT    4$              ;;BRANCH IF YES
6771 037454 021627 000175      CMP    (SP),#175       ;;IS IT A SPECIAL CHAR?
6772 037460 003002                BGT    4$              ;;BRANCH IF YES
6773 037462 042716 000040      BIC    #40,(SP)        ;;MAKE IT UPPER CASE
6774 037466 112677 177516      4$:    MOVB   (SP)+,@STKQIN  ;;AND PUT IT IN QUEUE
6775 037472 005237 037210      INC    $TKQIN          ;;UPDATE THE POINTER
6776 037476 023727 037210 037215      CMP    $TKQIN,$STKGEND ;;GO OFF THE END?
6777 037504 001003                BNE    5$              ;;BRANCH IF NO
```

;;RAN001/
;;RAN001/
;;RAN001/
;;RAN001/
;;RAN001/


```
6778 037506 012737 037214 037210      MOV      #STKQSR,STKQIN ;;RESET THE POINTER
6779 037514 000002                5$:      RTI                ;;RETURN
6780
6781
6782                ;*****
6783                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
6784                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
6785                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
6786                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
6786 037516 022737 000176 001140  $CKSWR:  CMP      #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
6787 037524 001124                BNE      15$              ;;EXIT IF NOT
6788 037526 105777 141412      TSTB     @STKS            ;;IS A CHAR WAITING?
6789 037532 100121                BPL      15$              ;;IF NOT, EXIT
6790 037534 117746 141406      MOVB     @STKB,-(SP)      ;;YES
6791 037540 042716 177600      BIC      #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
6792 037544 021627 000007      CMP      (SP),#7         ;;IS IT A CONTROL-G?
6793 037550 001300                BNE      2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
6794
6795
6796                ;*****
6797                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
6798                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
6799                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
6800 037552 123727 001134 000001  6$:      CMPB     $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
6801 037560 001674                BEQ      2$              ;;BRANCH IF YES
6802 037562 005726                TST      (SP)+           ;;CLEAR CONTROL-G OFF STACK
6803 037564 004737 037216      JSR      PC,$TKINT      ;;FLUSH THE TTY INPUT QUEUE
6804 037570 005077 141350      CLR      @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
6805 037574 112737 000001 001135      MOVB     #1,$INTAG      ;;SET INTERRUPT MODE INDICATOR
6806
6807 037602 104401 040426      $GTSWR:  TYPE     ,$CNTLG      ;;ECHO THE CONTROL-G (^G)
6808 037606 104401 040433      TYPE     ,$MSWR         ;;TYPE CURRENT CONTENTS
6809 037612 013746 000176      MOV      SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
6810 037616 104402                TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6811 037620 104401 040444      TYPE     ,$MNEW         ;;PROMPT FOR NEW SWR
6812 037624 005046      19$:     CLR      -(SP)      ;;CLEAR COUNTER
6813 037626 005046                CLR      -(SP)          ;;THE NEW SWR
6814 037630 105777 141310      7$:      TSTB     @STKS          ;;CHAR THERE?
6815 037634 100375                BPL      7$              ;;IF NOT TRY AGAIN
6816
6817 037636 117746 141304      MOVB     @STKB,-(SP)    ;;PICK UP CHAR
6818 037642 042716 177600      BIC      #^C177,(SP)   ;;MAKE IT 7-BIT ASCII
6819
6820 037646 021627 000003      CMP      (SP),#3        ;;IS IT A CONTROL-C?
6821 037652 001015                BNE      9$              ;;BRANCH IF NOT
6822 037654 104401 040414      TYPE     ,$CNTLC        ;;YES, ECHO CONTROL-C (^C)
6823 037660 062706 000006      ADD      #6,SP          ;;CLEAN UP STACK
6824 037664 123727 001135 000001  CMPB     $INTAG,#1      ;;REENABLE TTY KEYBOARD INTERRUPTS?
6825 037672 001003                BNE      8$              ;;BRANCH IF NO
6826 037674 012777 000100 141242      MOV      #100,@STKS    ;;ALLOW TTY KEYBOARD INTERRUPTS
6827 037702 000137 033170      8$:      JMP      CTRHLT        ;;CONTROL-C RESTART
6828
6829
6830 037706 021627 000025      9$:      CMP      (SP),#25       ;;IS IT A CONTROL-U?
6831 037712 001005                BNE      10$             ;;BRANCH IF NOT
6832 037714 104401 040421      TYPE     ,$CNTLU        ;;YES, ECHO CONTROL-U (^U)
6833 037720 062706 000006      20$:     ADD      #6,SP          ;;IGNORE PREVIOUS INPUT
```

```
6834 037724 000737 BR 19$ ;;LET'S TRY IT AGAIN
6835
6836
6837 037726 021627 000015 10$: CMP (SP),#15 ;;IS IT A <CR>?
6838 037732 001022 BNE 16$ ;;BRANCH IF NO
6839 037734 005766 000004 TST 4(SP) ;;YES, IS IT THE FIRST CHAR?
6840 037740 001403 BEQ 11$ ;;BRANCH IF YES
6841 037742 016677 000002 141170 MOV 2(SP),@SWP ;;SAVE NEW SWR
6842 037750 062706 000006 11$: ADD #6,SP ;;CLEAR UP STACK
6843 037754 104401 001211 14$: TYPE $CRLF ;;ECHO <CR> AND <LF>
6844 037760 123727 001135 000001 CMPB $INTAG,#1 ;;RE-ENABLE TTY KBD INTERRUPTS?
6845 037766 001003 BNE 15$ ;;BRANCH IF NOT
6846 037770 012777 000100 141146 MOV #100,@$TKS ;;RE-ENABLE TTY KBD INTERRUPTS
6847 037776 000002 15$: RTI ;;RETURN
6848 040000 004737 036412 16$: JSR PC,$TYPEC ;;ECHO CHAR
6849 040004 021627 000060 CMP (SP),#60 ;;CHAR < 0?
6850 040010 002420 BLT 18$ ;;BRANCH IF YES
6851 040012 021627 000067 CMP (SP),#67 ;;CHAR > 7?
6852 040016 003015 BGT 18$ ;;BRANCH IF YES
6853 040020 042726 000060 BIC #60,(SP)+ ;;STRIP-OFF ASCII
6854 040024 005766 000002 TST 2(SP) ;;IS THIS THE FIRST CHAR
6855 040030 001403 BEQ 17$ ;;BRANCH IF YES
6856 040032 006316 ASL (SP) ;;NO, SHIFT PRESENT
6857 040034 006316 ASI (SP) ;; CHAR OVER TO MAKE
6858 040036 006316 ASL (SP) ;; ROOM FOR NEW ONE.
6859 040040 005266 000002 17$: INC 2(SP) ;;KEEP COUNT OF CHAR
6860 040044 056616 177776 BIS -2(SP),(SP) ;;SET IN NEW CHAR
6861 040050 000667 BR 7$ ;;GET THE NEXT ONE
6862 040052 104401 001210 18$: TYPE $QUES ;;TYPE ?<CR><LF>
6863 040056 000720 BR 20$ ;;SIMULATE CONTROL-U
6864
6865
6866
6867
6868 *****
6869 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
6870 *CALL:
6871 * RDCHR ;;GET A CHARACTER FROM THE QUEUE
6872 * RETURN HERE ;;CHARACTER IS ON THE STACK
6873 * ;;WITH PARITY BIT STRIPPED OFF
6874
6875 $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC AND
6876 040062 016666 000004 000002 MOV 4(SP),2(SP) ;;THE PS
6877 040070 005066 000004 CLR 4(SP) ;;GET READY FOR A CHARACTER
6878 040074 005046 CLR -(SP) ;;PUT NEW PS ON STACK
6879 040076 012746 040104 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
6880 040102 000002 RTI ;;POP NEW PC AND PS
6881 040104 64$:
6882 040104 005737 037206 1$: TST $TKCNT ;;WAIT ON A CHARACTER
6883 040110 001775 BEQ 1$
6884 040112 005337 037206 DEC $TKCNT ;;DECREMENT THE COUNTER
6885 040116 117766 177070 000004 MOVB @STKQOUT,4(SP) ;;GET ONE CHARACTER
6886 040124 005237 037212 INC $TKQOUT ;;UPDATE THE POINTER
6887 040130 023727 037212 037215 CMP $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
6888 040136 001003 BNE 2$ ;;BRANCH IF NO
6889 040140 012737 037214 037212 MOV #$TKQSRST,$TKQOUT ;;RESET THE POINTER
```

```
6890 040146 000002 2$: RTI ;;RETURN
6891 .....
6892 *THIS ROUTINE WILL INPLT A STRING FROM THE TTY
6893 *CALL:
6894 * RDLIN ;;INPUT A STRING FROM THE TTY
6895 * RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
6896 * ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
6897
6898 040150 010346 $RDLIN: MOV R3,-(SP) ;;SAVE R3
6899 040152 005046 CLR -(SP) ;;CLEAR THE RUBOUT KEY
6900 040154 012703 040404 1$: MOV $TTYIN,R3 ;;GET ADDRESS
6901 040160 022703 040414 2$: CMP $TTYIN+8.,R3 ;;BUFFER FULL?
6902 040164 101456 BLOS 4$ ;;BR IF YES
6903 040166 104410 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
6904 040170 112613 MOVB (SP)+,(R3) ;;GET CHARACTER
6905 040172 122713 000177 10$: CMPB #177,(R3) ;;IS IT A RUBOUT
6906 040176 001022 BNE 5$ ;;BR IF NO
6907 040200 005716 TST (SP) ;;IS THIS THE FIRST RUBOUT?
6908 040202 001007 BNE 6$ ;;BR IF NO
6909 040204 112737 000134 040402 MOVB #'\\,9$ ;;TYPE A BACK SLASH
6910 040212 104401 040402 TYPE ,9$
6911 040216 012716 177777 MOV #-1,(SP) ;;SET THE RUBOUT KEY
6912 040222 005303 6$: DEC R3 ;;BACKUP BY ONE
6913 040224 020327 040404 CMP R3,$TTYIN ;;STACK EMPTY?
6914 040230 103434 BLO 4$ ;;BR IF YES
6915 040232 111337 040402 MOVB (R3),9$ ;;SETUP TO TYPEOLT THE DELETED CHAR.
6916 040236 104401 040402 TYPE ,9$ ;;GO TYPE
6917 040242 000746 BR 2$ ;;GO READ ANOTHER CHAR.
6918 040244 005716 5$: TST (SP) ;;RUBOUT KEY SET?
6919 040246 001406 BEQ 7$ ;;BR IF NO
6920 040250 112737 000134 040402 MOVB #'\\,9$ ;;TYPE A BACK SLASH
6921 040256 104401 040402 TYPE ,9$
6922 040262 005016 CLR (SP) ;;CLEAR THE RUBOUT KEY
6923 040264 122713 000025 7$: CMPB #25,(R3) ;;IS CHARACTER A CTRL U?
6924 040270 001003 BNE 8$ ;;BR IF NO
6925 040272 104401 040421 TYPE ,$CNTLU ;;TYPE A CONTROL 'U'
6926 040276 000726 BR 1$ ;;GO START OVER
6927 040300 122713 000022 8$: CMPB #22,(R3) ;;IS CHARACTER A '^R'?
6928 040304 001011 BNE 3$ ;;BRANCH IF NO
6929 040306 105013 CLRB (R3) ;;CLEAR THE CHARACTER
6930 040310 104401 001211 TYPE , $CRLF ;;TYPE A 'CR' & 'LF'
6931 040314 104401 040404 TYPE , $TTYIN ;;TYPE THE INPUT STRING
6932 040320 000717 BR 2$ ;;GO PICKUP ANOTHER CHACTER
6933 040322 104401 001210 4$: TYPE , $QUES ;;TYPE A '?'
6934 040326 000712 BR 1$ ;;CLEAR THE BUFFER AND LOOP
6935 040330 111337 040402 3$: MOVB (R3),9$ ;;ECHO THE CHARACTER
6936 040334 104401 040402 TYPE ,9$
6937 040340 122723 000015 CMPB #15,(R3)+ ;;CHECK FOR RETURN
6938 040344 001305 BNE 2$ ;;LOOP IF NOT RETURN
6939 040346 105063 177777 CLRB -1(R3) ;;CLEAR RETURN (THE 15)
6940 040352 104401 001212 TYPE , $LF ;;TYPE A LINE FEED
6941 040356 005726 TST (SP)+ ;;CLEAN RUBOUT KEY FROM THE STACK
6942 040360 012603 MOV (SP)+,R3 ;;RESTORE R3
6943 040362 011646 MOV (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
6944 040364 016666 000004 000002 MOV 4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
6945 040372 012766 040404 000004 MOV $TTYIN,4(SP)
```

6946	040400	000002				RTI	::RETURN
6947	040402	000			9\$:	.BYTE 0	::STORAGE FOR ASCII CHAR. TO TYPE
6948	040403	000				.BYTE 0	::TERMINATOR
6949	040404	000010			\$TTYIN:	.BLKB 8.	::RESERVE 8 BYTES FOR TTY INPUT
6950	040414	041536	005015	000	\$CNTLC:	.ASCIZ /^C/<15><12>	::CONTROL 'C'
6951	040421	136	006525	000012	\$CNTLU:	.ASCIZ /^U/<15><12>	::CONTROL 'U'
6952	040426	043536	005015	000	\$CNTLG:	.ASCIZ /^G/<15><12>	::CONTROL 'G'
6953	040433	015	051412	051127	\$MSWR:	.ASCIZ <15><12>/SWR - /	
6954	040440	036440	000040				
6955	040444	020040	042516	020127	\$MNEW:	.ASCIZ / NEW = /	
6956	040452	020075	000				
6957		040456					
6958					.EVEN		
6959					.SBTTL	READ AN OCTAL NUMBER FROM THE TTY	
6960							
6961					::*****		
6962					::*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND		
6963					::*CHANGE IT TO BINARY.		
6964					::*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL		
6965					::*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED		
6966					::*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST		
6967					::*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.		
6968					::*CALL:		
6969					::*	RDGCT	::READ AN OCTAL NUMBER
6970					::*	RETURN HERE	::LOW ORDER BITS ARE ON TOP OF THE STACK
6971					::*		::HIGH ORDER BITS ARE IN \$HIOCT
6972	040456	011646			\$RDOCT:	MOV (SP),-(SP)	::PROVIDE SPACE FOR THE
6973	040460	016666	000004	000002		MOV 4(SP),2(SP)	::INPUT NUMBER
6974	040466	010046				MOV R0,-(SP)	::PUSH R0 ON STACK
6975	040470	010146				MOV R1,-(SP)	::PUSH R1 ON STACK
6976	040472	010246				MOV R2,-(SP)	::PUSH R2 ON STACK
6977	040474	104411			1\$:	RD IN	::READ AN ASCII LINE
6978	040476	012600				MOV (SP)+,R0	::GET ADDRESS OF 1ST CHARACTER
6979	040500	010037	040604			MOV R0,5\$::AND SAVE IT
6980	040504	005001				CLR R1	::CLEAR DATA WORD
6981	040506	005002				CLR R2	
6982	040510	112046			2\$:	MOVB (R0)+,-(SP)	::PICKUP THIS CHARACTER
6983	040512	001420				BEQ 3\$::IF ZERO GET OUT
6984	040514	122716	000060			CMPB #'0,(SP)	::MAKE SURE THIS CHARACTER
6985	040520	003026				BGT 4\$::IS AN OCTAL DIGIT
6986	040522	122716	000067			CMPB #'7,(SP)	
6987	040526	002423				BLT 4\$	
6988	040530	006301				ASL R1	::*2
6989	040532	006102				ROL R2	
6990	040534	006301				ASL R1	::*4
6991	040536	006102				ROL R2	
6992	040540	006301				ASL R1	::*8
6993	040542	006102				ROL R2	
6994	040544	042716	177770			BIC #'C7,(SP)	::STRIP THE ASCII JUNK
6995	040550	062601				ADD (SP)+,R1	::ADD IN THIS DIGIT
6996	040552	000756				BR 2\$::LOOP
6997	040554	005726			3\$:	TST (SP)+	::CLEAN TERMINATOR FROM STACK
6998	040556	010166	000012			MOV R1,12(SP)	::SAVE THE RESULT
6999	040562	010237	040614			MOV R2,\$HIOCT	
7000	040566	012602				MOV (SP)+,R2	::POP STACK INT R2
7001	040570	012601				MOV (SP)+,R1	::POP STACK INT R1

7002 040572 012600
7003 040574 000002
7004 040576 005726
7005 040600 105010
7006 040602 104401
7007 040604 000000
7008 040606 104401 001210
7009 040612 000730
7010 040614 000000
7011
7012
7013
7014
7015
7016
7017
7018
7019
7020
7021
7022
7023
7024
7025
7026
7027
7028 040616
7029 040616 010046
7030 040620 010146
7031 040622 010246
7032 040624 010346
7033 040626 010446
7034 040630 010546
7035 040632 016646 000022
7036 040636 016646 000022
7037 040642 016646 000022
7038 040646 016646 000022
7039 040652 000002
7040
7041
7042
7043
7044 040654
7045 040654 012666 000022
7046 040660 012666 000022
7047 040664 012666 000022
7048 040670 012666 000022
7049 040674 012605
7050 040676 012604
7051 040700 012603
7052 040702 012602
7053 040704 012601
7054 040706 012600
7055 040710 000002
7056
7057

```
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI ;;RETURN
4$: TST (SP)+ ;;CLEAN PARTIAL FROM STACK
CLR (R0) ;;SET A TERMINATOR
TYPE ;;TYPE UP THRU THE BAD CHAR.
5$: .WORD 0
TYPE $QUES ;;'"' 'CR' & 'F'
BR 1$ ;;TRY AGAIN
$HI OCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

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

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

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

.SBTTL POWER DOWN AND UP ROUTINE
```

```
7058  
7059  
7060  
7061 :POWER DOWN ROUTINE  
7062 040712 017737 140222 003312 $PWRDN: MOV @SWR,SAVSWR ;SAVE SWITCH REGISTER  
7063 040720 012737 040740 000024 MOV #PWRUP,PWRVEC ;SET UP VECTOR  
7064 040726 012737 000340 000026 MOV #PR7,PWRVEC+2  
7065 040734 000000 HALT  
7066 040736 000776 BR -2 ;HANG UP  
7067  
7068  
7069  
7070 :POWER UP ROUTINE  
7071 040740 005037 041030 $PWRUP: CLR $PWRCT ;LOAD WAIT COUNT  
7072 040744 012737 000144 041032 MOV #100,$PWRCT+2  
7073 040752 005237 041030 1$: INC $PWRCT ;WAIT FOR TELETYPE  
7074 040756 001375 BNE 1$  
7075 040760 005337 041032 DEC $PWRCT+2  
7076 040764 001372 BNE 1$  
7077 040766 012737 040712 000024 MOV #PWRDN,PWRVEC ;SET UP FOR POWER DOWN VECTOR  
7078 040774 012737 000340 000026 MOV #PR7,PWRVEC+2  
7079 041002 012706 001100 MOV #STACK,SP ;FORCE STACK  
7080 041006 104401 041034 TYPE $POWER ;TYPE POWER  
7081 041012 013777 003312 140120 MOV SAVSWR,@SWR ;RESTORE SWITCH REGISTER  
7082 041020 013702 001270 MOV $BASE,R2 ;REINITIALISE R2 FOR '611 BASE  
7083 041024 000177 140056 JMP @SLPADR ;GO BACK TO LAST TEST  
7084  
7085 041030 000000 000000 $PWRCT: .WORD 0,0 ;TELETYPE TIME OUT  
7086 041034 047520 042527 000122 $POWER: .ASCIZ /POWER/  
7087 .EVEN  
7088 .SBTTL TRAP DECODER  
7089  
7090  
7091  
7092  
7093  
7094  
7095  
7096 041042 010046 $TRAP: MOV R0,-(SP) ;;SAVE R0  
7097 041044 016600 000002 MOV 2(SP),R0 ;;GET TRAP ADDRESS  
7098 041050 005740 TST -(R0) ;;BACKUP BY 2  
7099 041052 111000 MOV#B (R0),R0 ;;GET RIGHT BYTE OF TRAP  
7100 041054 006300 ASL R0 ;;POSITION FOR INDEXING  
7101 041056 016000 041076 MOV $TRPAD(R0),R0 ;;INDEX TO TABLE  
7102 041062 000200 RTS R0 ;;GO TO ROUTINE  
7103  
7104  
7105  
7106  
7107 041064 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN  
7108 041066 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN  
7109 041074 000002 RTI ;;RESTORE THE PSW  
7110  
7111 .SBTTL TRAP TABLE  
7112  
7113  
;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
```

7114
7115
7116
7117
7118 041076 041064
7119 041100 036200
7120 041102 036560
7121 041104 036534
7122 041106 036574
7123 041110 036762
7124
7125 041112 037606
7126
7127 041114 037516
7128 041116 040060
7129 041120 040150
7130 041122 040456
7131 041124 040616
7132 041126 040654
7133 041130 033676
7134
7135
7136 041132 001220 003244
7137 041136 001220 001116 003224
7138 041144 003164 003252 003254
7139 041152 003256 003260 003262
7140 041160 001220 001116 003200
7141 041166 003140 003226 003166
7142 041174 003230 003170
7143 041200 001220 001116 003200
7144 041206 003140 003210 003150
7145 041214 003214 003154 003204
7146 041222 003144 003202 003142
7147 041230 001220 001116 003200
7148 041236 003140 003210 003150
7149 041244 003214 003154
7150 041250 001220 001116 003222
7151 041256 003162 003264
7152 041262 001220 001116 003200
7153 041270 003140 003150 003154
7154 041276 001220 001116 003224
7155 041304 003164
7156 041306 001220 001116 003220
7157 041314 003160 003206 003146
7158 041322 001220 001116 003200
7159 041330 003140 003210 003150
7160 041336 003214 003154
7161 041342 053650 053652 053654
7162 041350 001220 001116 003200
7163 041356 003140 003210 003150
7164 041364 003214 003154 003310
7165 041372 001220 001116 003224
7166 041400 003164 003310
7167 041404 001220 001116 003234
7168 041412 003174 003262
7169 041416 001220 001116 003200

;*BY THE "TRAP" INSTRUCTION.

: ROUTINE

\$TRPAD: .WORD \$TRAP2
\$TYPE ::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
\$TYPOC ::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS ::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON ::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPDS ::CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)

\$GTSWR ::CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING

\$CKSWR ::CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
\$RDCHR ::CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ::CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
\$RDOCT ::CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
\$SAVREG ::CALL=SAVREG TRAP+13(104413) SAVE R0-R5 ROUTINE
\$RESREG ::CALL=RESREG TRAP+14(104414) RESTORE R0-R5 ROUTINE
\$SCOPI\$::CALL=SCOPI TRAP+15(104415) INTERNAL LOOP ON ERROR

.SBTTL DATA TABLE FOR PRINT OUT

DT000: .WORD \$TESTN,TRAPC
DT002: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT

DT003: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3

DT041: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,E.BA,T.BA,E.WC,T.WC

DT046: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER

DT051: .WORD \$TESTN,\$ERRPC,E.DB,T.DB,WRDCNT
DT052: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,T.CS2,T.ER
DT053: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1
DT054: .WORD \$TESTN,\$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
DT071: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER

DT074: .WORD VRCHDR,VRCHDR+2,VRCHDR+4
DT074: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,HDRCNT

DT077: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,HDRCNT
DT134: .WORD \$TESTN,\$ERRPC,E.ECPT,T.ECPT,BITCNT
DT144: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER

7170	041424	003140	003210	003150		
7171	041432	003214	003154			
7172	041436	003204	003144	003202	.WORD	E.BA,T.BA,E.WC,T.WC,E.DCYL,T.DCYL,E.DA,T.DA
7173	041444	003142	001220	003160		
7174	041452	003206	003146			
7175	041456	001220	001116	003234	DT151: .WORD	\$TESTN,\$ERRPC,E.ECPT,T.ECPT
7176	041464	003174				
7177					.SBTTL	DATA FORMAT FOR PRINT OUT
7178						
7179	041466	000001			DF000: .WORD	1
7180	041470	002	000		.BYTE	2,0
7181	041472	000005			DF002: .WORD	5
7182	041474	000	000		.BYTE	0,0
7183	041476	042541			.WORD	DH000A
7184	041500	000	000		.BYTE	0,0
7185	041502	042557			.WORD	DH000B
7186	041504	002	000		.BYTE	2,0
7187	041506	042623			.WORD	DH002A
7188	041510	000	000		.BYTE	0,0
7189	041512	042707			.WORD	DH002B
7190	041514	007	000		.BYTE	7,0
7191	041516	000005			DF003: .WORD	5 ;ERROR 3-24
7192	041520	000	000		.BYTE	0,0
7193	041522	042541			.WORD	DH000A
7194	041524	000	000		.BYTE	0,0
7195	041526	042557			.WORD	DH000B
7196	041530	002	000		.BYTE	2,0
7197	041532	042775			.WORD	DH003A
7198	041534	000	000		.BYTE	0,0
7199	041536	043054			.WORD	DH003B
7200	041540	006	000		.BYTE	6,0
7201	041542	000005			DF025: .WORD	5 ;ERROR 25-40
7202	041544	000	000		.BYTE	0,0
7203	041546	042541			.WORD	DH000A
7204	041550	000	000		.BYTE	0,0
7205	041552	042557			.WORD	DH000B
7206	041554	002	000		.BYTE	2,0
7207	041556	043133			.WORD	DH025A
7208	041560	000	000		.BYTE	0,0
7209	041562	043212			.WORD	DH025B
7210	041564	006	000		.BYTE	6,0
7211	041566	000007			DF041: .WORD	7
7212	041570	000	000		.BYTE	0,0
7213	041572	042541			.WORD	DH000A
7214	041574	000	000		.BYTE	0,0
7215	041576	042557			.WORD	DH000B
7216	041600	002	000		.BYTE	2,0
7217	041602	043272			.WORD	DH041A
7218	041604	000	000		.BYTE	0,0
7219	041606	043351			.WORD	DH041B
7220	041610	006	000		.BYTE	6,0
7221	041612	043426			.WORD	DH041C
7222	041614	000	000		.BYTE	0,0
7223	041616	043465			.WORD	DH041D
7224	041620	004	000		.BYTE	4,0
7225	041622	000005			DF046: .WORD	5

7226	041624	000	000	.BYTE	0,0	
7227	041626	04254		.WORD	DH000A	
7228	041630	000	000	.BYTE	0,0	
7229	041632	042557		.WORD	DH000B	
7230	041634	002	000	.BYTE	2,0	
7231	041636	043272		.WORD	DH041A	
7232	041640	000	000	.BYTE	0,0	
7233	041642	043351		.WORD	DH041B	
7234	041644	006	000	.BYTE	6,0	
7235	041646	000005		.WORD	5	DF051:
7236	041650	000	000	.BYTE	0,0	
7237	041652	042541		.WORD	DH000A	
7238	041654	000	000	.BYTE	0,0	
7239	041656	042557		.WORD	DH000B	
7240	041660	002	000	.BYTE	2,0	
7241	041662	043522		.WORD	DH051A	
7242	041664	000	000	.BYTE	0,0	
7243	041666	043547		.WORD	DH051B	
7244	041670	003	000	.BYTE	3,0	
7245	041672	000005		.WORD	5	DF052:
7246	041674	000	000	.BYTE	0,0	
7247	041676	042541		.WORD	DH000A	
7248	041700	000	000	.BYTE	0,0	
7249	041702	042557		.WORD	DH000B	
7250	041704	002	000	.BYTE	2,0	
7251	041706	043575		.WORD	DH052A	
7252	041710	000	000	.BYTE	0,0	
7253	041712	043614		.WORD	DH052B	
7254	041714	004	000	.BYTE	4,0	
7255	041716	000005		.WORD	5	DF053:
7256	041720	000	000	.BYTE	0,0	
7257	041722	042541		.WORD	DH000A	
7258	041724	000	000	.BYTE	0,0	
7259	041726	042557		.WORD	DH000B	
7260	041730	002	000	.BYTE	2,0	
7261	041732	043651		.WORD	DH053A	
7262	041734	000	000	.BYTE	0,0	
7263	041736	043670		.WORD	DH053B	
7264	041740	002	000	.BYTE	2,0	
7265	041742	000005		.WORD	5	DF054: ;ERROR-54-61
7266	041744	000	000	.BYTE	0,0	
7267	041746	042541		.WORD	DH000A	
7268	041750	000	000	.BYTE	0,0	
7269	041752	042557		.WORD	DH000B	
7270	041754	002	000	.BYTE	2,0	
7271	041756	043706		.WORD	DH054A	
7272	041760	000	000	.BYTE	0,0	
7273	041762	043745		.WORD	DH054B	
7274	041764	004	000	.BYTE	4,0	
7275	041766	000007		.WORD	7	DF071: ;ERRORS 71-73
7276	041770	000	000	.BYTE	0,0	
7277	041772	042541		.WORD	DH000A	
7278	041774	000	000	.BYTE	0,0	
7279	041776	042557		.WORD	DH000B	
7280	042000	002	000	.BYTE	2,0	
7281	042002	043272		.WORD	DH041A	

7282	042004	000	000		.BYTE	0.0
7283	042006	043357			.WORD	DH041B
7284	042010	006	000		.BYTE	6.0
7285	042012	044002			.WORD	DH071A
7286	042014	000	000		.BYTE	0.0
7287	042016	044023			.WORD	DH071B
7288	042020	003	000		.BYTE	3.0
7289	042022	000005		DF074:	.WORD	5
7290	042024	000	000		.BYTE	0.0
7291	042026	042541			.WORD	DH000A
7292	042030	000	000		.BYTE	0.0
7293	042032	042557			.WORD	DH000B
7294	042034	002	000		.BYTE	2.0
7295	042036	044052			.WORD	DH074A
7296	042040	000	000		.BYTE	0.0
7297	042042	044141			.WORD	DH074B
7298	042044	007	000		.BYTE	7.0
7299	042046	000005		DF077:	.WORD	5
7300	042050	000	000		.BYTE	0.0
7301	042052	042541			.WORD	DH000A
7302	042054	000	000		.BYTE	0.0
7303	042056	042557			.WORD	DH000B
7304	042060	002	000		.BYTE	2.0
7305	042062	044225			.WORD	DH077A
7306	042064	000	000		.BYTE	0.0
7307	042066	044254			.WORD	DH077B
7308	042070	003	000		.BYTE	3.0
7309	042072	000005		DF134:	.WORD	5
7310	042074	000	000		.BYTE	0.0
7311	042076	042541			.WORD	DH000A
7312	042100	000	000		.BYTE	0.0
7313	042102	042557			.WORD	DH000B
7314	042104	002	000		.BYTE	2.0
7315	042106	044300			.WORD	DH134A
7316	042110	000	000		.BYTE	0.0
7317	042112	044324			.WORD	DH134B
7318	042114	003	000		.BYTE	3.0
7319	042116	000007		DF144:	.WORD	7
7320	042120	000	000		.BYTE	0.0
7321	042122	042541			.WORD	DH000A
7322	042124	000	000		.BYTE	0.0
7323	042126	042557			.WORD	DH000B
7324	042130	002	000		.BYTE	2.0
7325	042132	043272			.WORD	DH041A
7326	042134	000	000		.BYTE	0.0
7327	042136	043351			.WORD	DH041B
7328	042140	006	000		.BYTE	6.0
7329	042142	044352			.WORD	DH144A
7330	042144	000	000		.BYTE	0.0
7331	042146	044451			.WORD	DH144B
7332	042150	010	000		.BYTE	10.0
7333	042152	000005		DF151:	.WORD	5
7334	042154	000	000		.BYTE	0.0
7335	042156	042541			.WORD	DH000A
7336	042160	000	000		.BYTE	0.0
7337	042162	042557			.WORD	DH000B

7618	045102	043040	047522	020115	
7619	045110	042522	042101	042040	
7620	045116	052101	000101		
7621	045122	052101	042524	050115	EM304: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE DATA/
7622	045130	044524	043516	052040	
7623	045136	020117	044103	041505	
7624	045144	020113	046103	040505	
7625	045152	020122	042515	051523	
7626	045160	043040	047522	020115	
7627	045166	051127	052111	020105	
7628	045174	040504	040524	000	
7629	045201	101	052124	046505	EM305: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE CHECK/
7630	045206	052120	047111	020107	
7631	045214	047524	041440	042510	
7632	045222	045503	041440	042514	
7633	045230	051101	046440	051505	
7634	045236	020123	051106	046517	
7635	045244	053440	044522	042524	
7636	045252	041440	042510	045503	
7637	045260	000			
7638	045261	101	052124	046505	EM306: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
7639	045266	052120	047111	020107	
7640	045274	047524	041440	042510	
7641	045302	045503	044040	040505	
7642	045310	020104	042507	042516	
7643	045316	040522	044524	047117	
7644	045324	005015			
7645	045326	044527	044124	053040	.ASCIZ /WITH VARIOUS CYLINDER VALUES/
7646	045334	051101	047511	051525	
7647	045342	041440	046131	047111	
7648	045350	042504	020122	040526	
7649	045356	052514	051505	000	
7650	045363	101	052124	046505	EM307: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
7651	045370	052120	047111	020107	
7652	045376	047524	041440	042510	
7653	045404	045503	044040	040505	
7654	045412	020104	042507	042516	
7655	045420	040522	044524	047117	
7656	045426	005015			
7657	045430	044527	044124	053040	.ASCIZ /WITH VARIOUS TRACK VALUES/
7658	045436	051101	047511	051525	
7659	045444	052040	040522	045503	
7660	045452	053040	046101	042525	
7661	045460	000123			
7662	045462	052101	042524	050115	EM308: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
7663	045470	044524	043516	052040	
7664	045476	020117	044103	041505	
7665	045504	020113	042510	042101	
7666	045512	043440	047105	051105	
7667	045520	052101	047511	006516	
7668	045526	012			
7669	045527	127	052111	020110	.ASCIZ /WITH VARIOUS SECTOR VALUES/
7670	045534	040526	044522	052517	
7671	045542	020123	042523	052103	
7672	045550	051117	053040	046101	
7673	045556	042525	000123		

7674	045562	052101	042524	050115	EM309: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>
7675	045570	044524	043516	052040	
7676	045576	020117	044103	041505	
7677	045604	020113	042510	042101	
7678	045612	043440	047105	051105	
7679	045620	052101	047511	006516	
7680	045626	012			
7681	045627	127	052111	020110	.ASCIZ /WITH VARIOUS FORMAT VALUES/
7682	045634	040526	044522	052517	
7683	045642	020123	047506	046522	
7684	045650	052101	053040	046101	
7685	045656	042525	000123		
7686	045662	052101	042524	050115	EM310: .ASCIZ /ATTEMPTING TO CHECK NPR DATA TRANSFER FOR WRITE DATA/
7687	045670	044524	043516	052040	
7688	045676	020117	044103	041505	
7689	045704	020113	050116	020122	
7690	045712	040504	040524	052040	
7691	045720	040522	051516	042506	
7692	045726	020122	047506	020122	
7693	045734	051127	052111	020105	
7694	045742	040504	040524	000	
7695	045747	101	052124	046505	EM311: .ASCIZ /ATTEMPTING TO CHECK HEADER RECOGNITION/
7696	045754	052120	047111	020107	
7697	045762	047524	041440	042510	
7698	045770	045503	044040	040505	
7699	045776	042504	020122	042522	
7700	046004	047503	047107	052111	
7701	046012	047511	000116		
7702	046016	052101	042524	050115	EM312: .ASCIZ /ATTEMPTING TO CHECK SECTOR INCREMENT/
7703	046024	044524	043516	052040	
7704	046032	020117	044103	041505	
7705	046040	020113	042523	052103	
7706	046046	051117	044440	041516	
7707	046054	042522	042515	052116	
7708	046062	000			
7709	046063	101	052124	046505	EM313: .ASCIZ /ATTEMPTING TO CHECK TRACK INCREMENT/
7710	046070	052120	047111	020107	
7711	046076	047524	041440	042510	
7712	046104	045503	052040	040522	
7713	046112	045503	044440	041516	
7714	046120	042522	042515	052116	
7715	046126	000			
7716	046127	101	052124	046505	EM314: .ASCIZ /ATTEMPTING TO CHECK CYLINDER INCREMENT/
7717	046134	052120	047111	020107	
7718	046142	047524	041440	042510	
7719	046150	045503	041440	046131	
7720	046156	047111	042504	020122	
7721	046164	047111	051103	046505	
7722	046172	047105	000124		
7723	046176	052101	042524	050115	EM315: .ASCII /ATTEMPTING TO CHECK SECTOR PULSE DETECTION/<15><12>
7724	046204	044524	043516	052040	
7725	046212	020117	044103	041505	
7726	046220	020113	042523	052103	
7727	046226	051117	050040	046125	
7728	046234	042523	042040	052105	
7729	046242	041505	044524	047117	

7730	046250	005015			
7731	046252	044527	044124	053440	.ASCIZ /WITH WRITE DATA/
7732	046260	044522	042524	042040	
7733	046266	052101	000101		
7734	046272	052101	042524	050115	EM316: .ASCIZ /ATTEMPTING TO FORCE BAD SECTOR ERROR/
7735	046300	044524	043516	052040	
7736	046306	020117	047506	041522	
7737	046314	020105	040502	020104	
7738	046322	042523	052103	051117	
7739	046330	042440	051122	051117	
7740	046336	000			
7741	046337	101	052124	046505	EM317: .ASCII /ATTEMPTING TO FORCE HEADER VRC ERROR/<15><12>
7742	046344	052120	047111	020107	
7743	046352	047524	043040	051117	
7744	046360	042503	044040	040505	
7745	046366	042504	020122	051126	
7746	046374	020103	051105	047522	
7747	046402	006522	012		
7748	046405	127	052111	020110	.ASCIZ /WITH BAD SECTOR PRESENT/
7749	046412	040502	020104	042523	
7750	046420	052103	051117	050040	
7751	046426	042522	042523	052116	
7752	046434	000			
7753	046435	101	052124	046505	EM318: .ASCIZ /ATTEMPTING TO FORCE HVRC ERROR/
7754	046442	052120	047111	020107	
7755	046450	047524	043040	051117	
7756	046456	042503	044040	051126	
7757	046464	020103	051105	047522	
7758	046472	000122			
7759	046474	052101	042524	050115	EM319: .ASCIZ /ATTEMPTING TO FORCE OPERATION INCOMPLETE/
7760	046502	044524	043516	052040	
7761	046510	020117	047506	041522	
7762	046516	020105	050117	051105	
7763	046524	052101	047511	020116	
7764	046532	047111	047503	050115	
7765	046540	042514	042524	000	
7766	046545	101	052124	046505	EM320: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37/
7767	046552	044520	043516	052040	
7768	046560	020117	047506	041522	
7769	046566	020105	050117	020111	
7770	046574	044527	044124	044040	
7771	046602	051126	020103	051105	
7772	046610	047522	020122	047117	
7773	046616	044040	040505	042504	
7774	046624	020122	033463	000	
7775	046631	101	052124	046505	EM321: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36/
7776	046636	044520	043516	052040	
7777	046644	020117	047506	041522	
7778	046652	020105	050117	020111	
7779	046660	044527	044124	044040	
7780	046666	051126	020103	051105	
7781	046674	047522	020122	047117	
7782	046702	044040	040505	042504	
7783	046710	020122	033063	000	
7784	046715	101	052124	046505	EM322: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0/
7785	046722	044520	043516	052040	

7786	046730	020117	047506	041522	
7787	046736	020105	050117	020111	
7788	046744	044527	044124	044040	
7789	046752	051126	020103	051105	
7790	046760	047522	020122	047117	
7791	046766	044040	040505	042504	
7792	046774	020122	000060		
7793	047000	044103	041505	044513	EM323: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS BAD SECTOR ERROR/
7794	047006	043516	044040	040505	
7795	047014	042504	020122	042522	
7796	047022	047503	047107	052111	
7797	047030	047511	020116	044527	
7798	047036	044124	050040	042522	
7799	047044	044526	052517	020123	
7800	047052	040502	020104	042523	
7801	047060	052103	051117	042440	
7802	047066	051122	051117	000	
7803	047073	103	042510	045503	EM324: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER VRC ERROR/
7804	047100	047111	020107	042510	
7805	047106	042101	051105	051040	
7806	047114	041505	043517	044516	
7807	047122	044524	047117	053440	
7808	047130	052111	020110	051120	
7809	047136	053105	047511	051525	
7810	047144	044040	040505	042504	
7811	047152	020122	051126	020103	
7812	047160	051105	047522	000122	
7813	047166	047506	041522	047111	EM325: .ASCIZ /FORCING BAD SECTOR ERROR WITH PREVIOUS HEADER VRC ERROR/
7814	047174	020107	040502	020104	
7815	047202	042523	052103	051117	
7816	047210	042440	051122	051117	
7817	047216	053440	052111	020110	
7818	047224	051120	053105	047511	
7819	047232	051525	044040	040505	
7820	047240	042504	020122	051126	
7821	047246	020103	051105	047522	
7822	047254	000122			
7823	047256	047506	041522	047111	EM326: .ASCIZ /FORCING HEAD VRC ERROR WITH PREVIOUS BAD SECTOR ERROR/
7824	047264	020107	042510	042101	
7825	047272	053040	041522	042440	
7826	047300	051122	051117	053440	
7827	047306	052111	020110	051120	
7828	047314	053105	047511	051525	
7829	047322	041040	042101	051440	
7830	047330	041505	047524	020122	
7831	047336	051105	047522	000122	
7832	047344	052101	042524	050115	EM327: .ASCIZ /ATTEMPTING TO WRITE SYNCH WITH WRITE DATA/
7833	047352	044524	043516	052040	
7834	047360	020117	051127	052111	
7835	047366	020105	054523	041516	
7836	047374	020110	044527	044124	
7837	047402	053440	044522	042524	
7838	047410	042040	052101	000101	
7839	047416	052101	042524	050115	EM328: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD WITH WRITE DATA/
7840	047424	044524	043516	052040	
7841	047432	020117	051127	052111	

7842	047440	020105	040504	040524	
7843	047446	043040	042511	042114	
7844	047454	053440	052111	020110	
7845	047462	051127	052111	020105	
7846	047470	040504	040524	000	
7847	047475	101	052124	046505	EM329: .ASCIZ /ATTEMPTING TO CHECK ECC PATTERN CLEARING/
7848	047502	052120	047111	020107	
7849	047510	047524	041440	042510	
7850	047516	045503	042440	041503	
7851	047524	050040	052101	042524	
7852	047532	047122	041440	042514	
7853	047540	051101	047111	000107	
7854	047546	052101	042524	050115	EM330: .ASCIZ /ATTEMPTING TO CHECK ECC GENERATION/
7855	047554	044524	043516	052040	
7856	047562	020117	044103	041505	
7857	047570	020113	041505	020103	
7858	047576	042507	042516	040522	
7859	047604	044524	047117	000	
7860	047611	101	052124	046505	EM331: .ASCIZ /ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR/
7861	047616	052120	047111	020107	
7862	047624	040502	020104	042523	
7863	047632	052103	051117	042440	
7864	047640	051122	051117	051440	
7865	047646	052105	044524	043516	
7866	047654	041440	047117	051124	
7867	047662	046117	042514	020122	
7868	047670	051105	047522	000122	
7869	047676	052101	042524	050115	EM332: .ASCIZ /ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR/
7870	047704	044524	043516	044040	
7871	047712	040505	042504	020122	
7872	047720	051126	020103	051105	
7873	047726	047522	020122	042523	
7874	047734	052124	047111	020107	
7875	047742	047503	052116	047522	
7876	047750	046114	051105	042440	
7877	047756	051122	051117	000	
7878	047763	101	052124	046505	EM333: .ASCIZ /ATTEMPTING TO WRITE ECC WORDS/
7879	047770	052120	047111	020107	
7880	047776	047524	053440	044522	
7881	050004	042524	042440	041503	
7882	050012	053440	051117	051504	
7883	050020	000			
7884	050021	101	052124	046505	EM334: .ASCIZ /ATTEMPTING TO WRITE POSTAMBLE/
7885	050026	052120	047111	020107	
7886	050034	047524	053440	044522	
7887	050042	042524	050040	051517	
7888	050050	040524	041115	042514	
7889	050056	000			
7890	050057	101	052124	046505	EM335: .ASCIZ /ATTEMPTING COMPLETE EXECUTION OF WRITE DATA/
7891	050064	052120	047111	020107	
7892	050072	047503	050115	042514	
7893	050100	042524	042440	042530	
7894	050106	052503	044524	047117	
7895	050114	047440	020106	051127	
7896	050122	052111	020105	040504	
7897	050130	040524	000		

7898	050133	101	052124	046505	EM336: .ASCIZ /ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR/
7899	050140	052120	047111	020107	
7900	050146	051105	047522	020122	
7901	050154	046103	040505	020122	
7902	050162	044527	044124	041440	
7903	050170	047117	051124	046117	
7904	050176	042514	020122	046103	
7905	050204	040505	000122		
7906	050210	052101	042524	050115	EM337: .ASCIZ /ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL/
7907	050216	044524	043516	050040	
7908	050224	051101	044524	046101	
7909	050232	051440	041505	047524	
7910	050240	020122	051127	052111	
7911	050246	020105	044527	044124	
7912	050254	055040	051105	020117	
7913	050262	044506	046114	000	
7914	050267	101	052124	046505	EM338: .ASCIZ /ATTEMPTING TO WRITE 18 BIT DATA FIELD WITH WRITE DATA/
7915	050274	052120	047111	020107	
7916	050302	047524	053440	044522	
7917	050310	042524	030440	020070	
7918	050316	044502	020124	040504	
7919	050324	040524	043040	042511	
7920	050332	042114	053440	052111	
7921	050340	020110	051127	052111	
7922	050346	020105	040504	040524	
7923	050354	000			
7924	050355	101	052124	046505	EM339: .ASCII /ATTEMPTING TO WRITE BIT 16-17 OF 18 BIT/
7925	050362	052120	047111	020107	
7926	050370	047524	053440	044522	
7927	050376	042524	041040	052111	
7928	050404	030440	076466	033461	
7929	050412	047440	020106	034061	
7930	050420	041040	052111		
7931	050424	005015	040504	040524	.ASCIZ <15><12>/DATA FIELD WITH WRITE DATA/
7932	050432	043040	042511	042114	
7933	050440	053440	052111	020110	
7934	050446	051127	052111	020105	
7935	050454	040504	040524	000	
7936	050461	101	052124	046505	EM340: .ASCIZ /ATTEMPTING WRITE DATA IN 18 BIT MODE/
7937	050466	052120	047111	020107	
7938	050474	051127	052111	020105	
7939	050502	040504	040524	044440	
7940	050510	020116	034061	041040	
7941	050516	052111	046440	042117	
7942	050524	000105			
7943	050526	051503	020061	047111	EM4000: .ASCIZ /CS1 INCORRECT/
7944	050534	047503	051122	041505	
7945	050542	000124			
7946	050544	042515	051523	040440	EM4001: .ASCIZ /MESS A INCORRECT/
7947	050552	044440	041516	051117	
7948	050560	042522	052103	000	
7949	050565	115	051505	020123	EM4002: .ASCIZ /MESS B INCORRECT/
7950	050572	020102	047111	047503	
7951	050600	051122	041505	000124	
7952	050606	042510	042101	051105	EM4003: .ASCIZ /HEADER WORD 1 INCORRECT/
7953	050614	053440	051117	020104	

7954	050622	020061	047111	047503	
7955	050630	051122	041505	000124	
7956	050636	042510	042101	051105	EM4004: .ASCIZ /HEADER WORD 2 INCORRECT/
7957	050644	053440	051117	020104	
7958	050652	020062	047111	047503	
7959	050660	051122	041505	000124	
7960	050666	051503	020062	047111	EM4005: .ASCIZ /CS2 INCORRECT/
7961	050674	047503	051122	041505	
7962	050702	000124			
7963	050704	051105	047522	020122	EM4006: .ASCIZ /ERROR REG INCORRECT/
7964	050712	042522	020107	047111	
7965	050720	047503	051122	041505	
7966	050726	000124			
7967	050730	052502	020123	042101	EM4007: .ASCIZ /BUS ADDRESS INCORRECT/
7968	050736	051104	051505	020123	
7969	050744	047111	047503	051122	
7970	050752	041505	000124		
7971	050756	047527	042122	041440	EM4008: .ASCIZ /WORD COUNT INCORRECT/
7972	050764	052517	052116	044440	
7973	050772	041516	051117	042522	
7974	051000	052103	000		
7975	051003	103	030523	044440	EM4009: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
7976	051010	041516	051117	042522	
7977	051016	052103	040440	052106	
7978	051024	051105	051040	040505	
7979	051032	044504	043516	042040	
7980	051040	052101	020101	052502	
7981	051046	043106	051105	000	
7982	051053	103	031123	044440	EM4010: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
7983	051060	041516	051117	042522	
7984	051066	052103	040440	052106	
7985	051074	051105	051040	040505	
7986	051102	044504	043516	042040	
7987	051110	052101	020101	052502	
7988	051116	043106	051105	000	
7989	051123	105	051122	051117	EM4011: .ASCIZ /ERROR REG INCORRECT AFTER READING DATA BUFFER/
7990	051130	051040	043505	044440	
7991	051136	041516	051117	042522	
7992	051144	052103	040440	052106	
7993	051152	051105	051040	040505	
7994	051160	044504	043516	042040	
7995	051166	052101	020101	052502	
7996	051174	043106	051105	000	
7997	051201	104	052101	020101	EM4012: .ASCIZ /DATA READ FROM MEMORY INCORRECT/
7998	051206	042522	042101	043040	
7999	051214	047522	020115	042515	
8000	051222	047515	054522	044440	
8001	051230	041516	051117	042522	
8002	051236	052103	000		
8003	051241	115	030522	044440	EM4013: .ASCIZ /MR1 INCORRECT AFTER GAP IN WRITE DATA/
8004	051246	041516	051117	042522	
8005	051254	052103	040440	052106	
8006	051262	051105	043440	050101	
8007	051270	044440	020116	051127	
8008	051276	052111	020105	040504	
8009	051304	040524	000		

8010	051307	104	051511	020113	EM4014: .ASCIZ /DISK ADDRESS REG. INCORRECT/
8011	051314	042101	051104	051505	
8012	051322	020123	042522	027107	
8013	051330	044440	041516	051117	
8014	051336	042522	052103	000	
8015	051343	103	046131	047111	EM4015: .ASCIZ /CYLINDER ADDRESS REG. INCORRECT/
8016	051350	042504	020122	042101	
8017	051356	051104	051505	020123	
8018	051364	042522	027107	044440	
8019	051372	041516	051117	042522	
8020	051400	052103	000		
8021	051403	115	030522	044440	EM4016: .ASCIZ /MR1 INCORRECT/
8022	051410	041516	051117	042522	
8023	051416	052103	000		
8024	051421	105	041503	050040	EM4017: .ASCIZ /ECC PAT REG INCORRECT/
8025	051426	052101	051040	043505	
8026	051434	044440	041516	051117	
8027	051442	042522	052103	000	
8028	051447	115	030522	044440	FMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
8029	051454	041516	051117	042522	
8030	051462	052103	047440	020116	
8031	051470	051461	020124	047504	
8032	051476	047127	040527	042122	
8033	051504	052040	040522	051516	
8034	051512	051511	047511	020116	
8035	051520	043117	046440	044501	
8036	051526	052116	041440	047514	
8037	051534	045503	000		
8038	051537	115	030522	044440	EMW4: .ASCIZ /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/
8039	051544	041516	051117	042522	
8040	051552	052103	047440	020116	
8041	051560	047062	020104	047504	
8042	051566	047127	040527	042122	
8043	051574	052040	040522	051516	
8044	051602	052111	047511	020116	
8045	051610	043117	046440	044501	
8046	051616	052116	041440	047514	
8047	051624	045503	000		
8048					
8049					
8050		051630			
8051	051630				
8052	051630	000	000		
8053	051632	001	001		
8054	051634	002	002		
8055	051636	003	003		
8056	051640	004	004		
8057	051642	005	005		
8058	051644	006	006		
8059	051646	007	007		
8060	051650	010	010		
8061	051652	011	011		
8062	051654	012	012		
8063	051656	013	013		
8064	051660	014	014		
8065	051662	015	015		

.SBT'L DATA BUFFERS

.EVEN
NPRBUF:

.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

8066	051664	016	016	.BYTE	16,16
8067	051666	017	017	.BYTE	17,17
8068	051670	020	020	.BYTE	20,20
8069	051672	021	021	.BYTE	21,21
8070	051674	022	022	.BYTE	22,22
8071	051676	023	023	.BYTE	23,23
8072	051700	024	024	.BYTE	24,24
8073	051702	025	025	.BYTE	25,25
8074	051704	026	026	.BYTE	26,26
8075	051706	027	027	.BYTE	27,27
8076	051710	030	030	.BYTE	30,30
8077	051712	031	031	.BYTE	31,31
8078	051714	032	032	.BYTE	32,32
8079	051716	033	033	.BYTE	33,33
8080	051720	034	034	.BYTE	34,34
8081	051722	035	035	.BYTE	35,35
8082	051724	036	036	.BYTE	36,36
8083	051726	037	037	.BYTE	37,37
8084	051730	040	040	.BYTE	40,40
8085	051732	041	041	.BYTE	41,41
8086	051734	042	042	.BYTE	42,42
8087	051736	043	043	.BYTE	43,43
8088	051740	044	044	.BYTE	44,44
8089	051742	045	045	.BYTE	45,45
8090	051744	046	046	.BYTE	46,46
8091	051746	047	047	.BYTE	47,47
8092	051750	050	050	.BYTE	50,50
8093	051752	051	051	.BYTE	51,51
8094	051754	052	052	.BYTE	52,52
8095	051756	053	053	.BYTE	53,53
8096	051760	054	054	.BYTE	54,54
8097	051762	055	055	.BYTE	55,55
8098	051764	056	056	.BYTE	56,56
8099	051766	057	057	.BYTE	57,57
8100	051770	060	060	.BYTE	60,60
8101	051772	061	061	.BYTE	61,61
8102	051774	062	062	.BYTE	62,62
8103	051776	063	063	.BYTE	63,63
8104	052000	064	064	.BYTE	64,64
8105	052002	065	065	.BYTE	65,65
8106	052004	066	066	.BYTE	66,66
8107	052006	067	067	.BYTE	67,67
8108	052010	070	070	.BYTE	70,70
8109	052012	071	071	.BYTE	71,71
8110	052014	072	072	.BYTE	72,72
8111	052016	073	073	.BYTE	73,73
8112	052020	074	074	.BYTE	74,74
8113	052022	075	075	.BYTE	75,75
8114	052024	076	076	.BYTE	76,76
8115	052026	077	077	.BYTE	77,77
8116	052030	100	100	.BYTE	100,100
8117	052032	101	101	.BYTE	101,101
8118	052034	102	102	.BYTE	102,102
8119	052036	103	103	.BYTE	103,103
8120	052040	000000		.WORD	000000
8121	052042	140000		.WORD	140000

HEAD1:

8178	052236	177777	.WORD	177777
8179	052240	177777	.WORD	177777
8180	052242	177777	.WORD	177777
8181	052244	177777	.WORD	177777
8182	052246	177777	.WORD	177777
8183	052250	001252	OP11: .WORD	001252
8184	052252	141123	.WORD	141123
8185	052254	140371	.WORD	140371
8186	052256	001251	.WORD	001251
8187	052260	141123	.WORD	141123
8188	052262	140372	.WORD	140372
8189	052264	001257	.WORD	001257
8190	052266	141123	.WORD	141123
8191	052270	140374	.WORD	140374
8192	052272	001243	.WORD	001243
8193	052274	141123	.WORD	141123
8194	052276	140360	.WORD	140360
8195	052300	001273	.WORD	001273
8196	052302	141123	.WORD	141123
8197	052304	140350	.WORD	140350
8198	052306	001213	.WORD	001213
8199	052310	141123	.WORD	141123
8200	052312	140330	.WORD	140330
8201	052314	001353	.WORD	001353
8202	052316	141123	.WORD	141123
8203	052320	140270	.WORD	140270
8204	052322	001053	.WORD	001053
8205	052324	141123	.WORD	141123
8206	052326	140170	.WORD	140170
8207	052330	001653	.WORD	001653
8208	052332	141123	.WORD	141123
8209	052334	140770	.WORD	140770
8210	052336	000253	.WORD	000253
8211	052340	141123	.WORD	141123
8212	052342	141370	.WORD	141370
8213	052344	003253	.WORD	003253
8214	052346	141123	.WORD	141123
8215	052350	142370	.WORD	142370
8216	052352	005253	.WORD	005253
8217	052354	141123	.WORD	141123
8218	052356	144370	.WORD	144370
8219	052360	011253	.WORD	011253
8220	052362	141123	.WORD	141123
8221	052364	150370	.WORD	150370
8222	052366	021253	.WORD	021253
8223	052370	141123	.WORD	141123
8224	052372	160370	.WORD	160370
8225	052374	041253	.WORD	041253
8226	052376	141123	.WORD	141123
8227	052400	100370	.WORD	100370
8228	052402	101253	.WORD	101253
8229	052404	141123	.WORD	141123
8230	052406	040370	.WORD	040370
8231	052410	001253	.WORD	001253
8232	0524 2	141122	.WORD	141122
8233	052414	140371	.WORD	140371

8234	052416	001253	.WORD	001253
8235	052420	141121	.WORD	141121
8236	052422	140372	.WORD	140372
8237	052424	001253	.WORD	001253
8238	052426	141127	.WORD	141127
8239	052430	140374	.WORD	140374
8240	052432	001253	.WORD	001253
8241	052434	141133	.WORD	141133
8242	052436	140360	.WORD	140360
8243	052440	001253	.WORD	001253
8244	052442	141103	.WORD	141103
8245	052444	140350	.WORD	140350
8246	052446	001253	.WORD	001253
8247	052450	141163	.WORD	141163
8248	052452	140330	.WORD	140330
8249	052454	001253	.WORD	001253
8250	052456	141023	.WORD	141023
8251	052460	140270	.WORD	140270
8252	052462	001253	.WORD	001253
8253	052464	141323	.WORD	141323
8254	052466	140170	.WORD	140170
8255	052470	001253	.WORD	001253
8256	052472	141523	.WORD	141523
8257	052474	140770	.WORD	140770
8258	052476	001253	.WORD	001253
8259	052500	140123	.WORD	140123
8260	052502	141370	.WORD	141370
8261	052504	001253	.WORD	001253
8262	052506	143123	.WORD	143123
8263	052510	142370	.WORD	142370
8264	052512	001253	.WORD	001253
8265	052514	145123	.WORD	145123
8266	052516	144370	.WORD	144370
8267	052520	001253	.WORD	001253
8268	052522	151123	.WORD	151123
8269	052524	150370	.WORD	150370
8270	052526	001253	.WORD	001253
8271	052530	161123	.WORD	161123
8272	052532	160370	.WORD	160370
8273	052534	001250	.WORD	001250
8274	052536	141123	.WORD	141123
8275	052540	140373	.WORD	140373
8276	052542	141253	.WORD	141253
8277	052544	141123	.WORD	141123
8278	052546	000370	.WORD	000370
8279	052550	000240	.WORD	000240
8280	052552	140000	.WORD	140000
8281	052554	140240	.WORD	140240
8282	052556	000240	.WORD	000240
8283	052560	140000	.WORD	140000
8284	052562	140240	.WORD	140240
8285	052564	000240	.WORD	000240
8286	052566	140000	.WORD	140000
8287	052570	140240	.WORD	140240
8288	052572	000240	.WORD	000240
8289	052574	140000	.WORD	140000

OPI2:

8290	052576	140240	.WORD	140240
8291	052600	000240	.WORD	000240
8292	052602	140000	.WORD	140000
8293	052604	140240	.WORD	140240
8294	052606	000240	.WORD	000240
8295	052610	140000	.WORD	140000
8296	052612	140240	.WORD	140240
8297	052614	000240	.WORD	000240
8298	052616	140000	.WORD	140000
8299	052620	140240	.WORD	140240
8300	052622	000240	.WORD	000240
8301	052624	140000	.WORD	140000
8302	052626	140240	.WORD	140240
8303	052630	000240	.WORD	000240
8304	052632	140000	.WORD	140000
8305	052634	140240	.WORD	140240
8306	052636	000240	.WORD	000240
8307	052640	140000	.WORD	140000
8308	052642	140240	.WORD	140240
8309	052644	000240	.WORD	000240
8310	052646	140000	.WORD	140000
8311	052650	140240	.WORD	140240
8312	052652	000240	.WORD	000240
8313	052654	140000	.WORD	140000
8314	052656	140240	.WORD	140240
8315	052660	000240	.WORD	000240
8316	052662	140000	.WORD	140000
8317	052664	140240	.WORD	140240
8318	052666	000240	.WORD	000240
8319	052670	140000	.WORD	140000
8320	052672	140240	.WORD	140240
8321	052674	000240	.WORD	000240
8322	052676	140000	.WORD	140000
8323	052700	140240	.WORD	140240
8324	052702	000240	.WORD	000240
8325	052704	140000	.WORD	140000
8326	052706	140240	.WORD	140240
8327	052710	000240	.WORD	000240
8328	052712	140000	.WORD	140000
8329	052714	140240	.WORD	140240
8330	052716	000240	.WORD	000240
8331	052720	140000	.WORD	140000
8332	052722	140240	.WORD	140240
8333	052724	000240	.WORD	000240
8334	052726	140000	.WORD	140000
8335	052730	140240	.WORD	140240
8336	052732	000240	.WORD	000240
8337	052734	140000	.WORD	140000
8338	052736	140240	.WORD	140240
8339	052740	000240	.WORD	000240
8340	052742	140000	.WORD	140000
8341	052744	140240	.WORD	140240
8342	052746	000240	.WORD	000240
8343	052750	140000	.WORD	140000
8344	052752	140240	.WORD	140240
8345	052754	000240	.WORD	000240

CZR6DDO RK611 DSKLS PRT4
CZR6DD.P11 27-AUG-81 10:35

MACV11 30(1046) 28-AUG-81 10:56 J 12
DATA BUFFERS PAGE 153

SEQ 0152

8346	052756	140000	.WORD	140000
8347	052760	140240	.WORD	140240
8348	052762	000240	.WORD	000240
8349	052764	140000	.WORD	140000
8350	052766	140240	.WORD	140240

8351	052770	000240	.WORD	000240
8352	052772	140000	.WORD	140000
8353	052774	140240	.WORD	140240
8354	052776	000240	.WORD	000240
8355	053000	140000	.WORD	140000
8356	053002	140240	.WORD	140240
8357	053004	000240	.WORD	000240
8358	053006	140000	.WORD	140000
8359	053010	140240	.WORD	140240
8360	053012	000240	.WORD	000240
8361	053014	140000	.WORD	140000
8362	053016	140240	.WORD	140240
8363	053020	000240	.WORD	000240
8364	053022	140000	.WORD	140000
8365	053024	140240	.WORD	140240
8366	053026	000240	.WORD	000240
8367	053030	140000	.WORD	140000
8368	053032	140240	.WORD	140240
8369	053034	000240	.WORD	000240
8370	053036	140000	.WORD	140000
8371	053040	140040	.WORD	140040
8372	053042	000240	.WORD	000240
8373	053044	140000	.WORD	140000
8374	053046	140040	.WORD	140040
8375	053050	000300	.WORD	000300
8376	053052	140000	.WORD	140000
8377	053054	140300	.WORD	140300
8378	053056	000300	.WORD	000300
8379	053060	140000	.WORD	140000
8380	053062	140300	.WORD	140300
8381	053064	000300	.WORD	000300
8382	053066	140000	.WORD	140000
8383	053070	140300	.WORD	140300
8384	053072	000300	.WORD	000300
8385	053074	140000	.WORD	140000
8386	053076	140300	.WORD	140300
8387	053100	000300	.WORD	000300
8388	053102	140000	.WORD	140000
8389	053104	140300	.WORD	140300
8390	053106	000300	.WORD	000300
8391	053110	140000	.WORD	140000
8392	053112	140300	.WORD	140300
8393	053114	000300	.WORD	000300
8394	053116	140000	.WORD	140000
8395	053120	140300	.WORD	140300
8396	053122	000300	.WORD	000300
8397	053124	140000	.WORD	140000
8398	053126	140300	.WORD	140300
8399	053130	000300	.WORD	000300
8400	053132	140000	.WORD	140000
8401	053134	140300	.WORD	140300
8402	053136	000300	.WORD	000300
8403	053140	140000	.WORD	140000
8404	053142	140300	.WORD	140300
8405	053144	000300	.WORD	000300
8406	053146	140000	.WORD	140000

OPI3:

8407	053150	140300	.WORD	140300
8408	053152	000300	.WORD	000300
8409	053154	140000	.WORD	140000
8410	053156	140300	.WORD	140300
8411	053160	000300	.WORD	000300
8412	053162	140000	.WORD	140000
8413	053164	140300	.WORD	140300
8414	053166	000300	.WORD	000300
8415	053170	140000	.WORD	140000
8416	053172	140300	.WORD	140300
8417	053174	000300	.WORD	000300
8418	053176	140000	.WORD	140000
8419	053200	140300	.WORD	140300
8420	053202	000300	.WORD	000300
8421	053204	140000	.WORD	140000
8422	053206	140300	.WORD	140300
8423	053210	000300	.WORD	000300
8424	053212	140000	.WORD	140000
8425	053214	140300	.WORD	140300
8426	053216	000300	.WORD	000300
8427	053220	140000	.WORD	140000
8428	053222	140300	.WORD	140300
8429	053224	000300	.WORD	000300
8430	053226	140000	.WORD	140000
8431	053230	140300	.WORD	140300
8432	053232	000300	.WORD	000300
8433	053234	140000	.WORD	140000
8434	053236	140300	.WORD	140300
8435	053240	000300	.WORD	000300
8436	053242	140000	.WORD	140000
8437	053244	140300	.WORD	140300
8438	053246	000300	.WORD	000300
8439	053250	140000	.WORD	140000
8440	053252	140300	.WORD	140300
8441	053254	000300	.WORD	000300
8442	053256	140000	.WORD	140000
8443	053260	140300	.WORD	140300
8444	053262	000300	.WORD	000300
8445	053264	140000	.WORD	140000
8446	053266	140300	.WORD	140300
8447	053270	000300	.WORD	000300
8448	053272	140000	.WORD	140000
8449	053274	140300	.WORD	140300
8450	053276	000300	.WORD	000300
8451	053300	140000	.WORD	140000
8452	053302	140300	.WORD	140300
8453	053304	000300	.WORD	000300
8454	053306	140000	.WORD	140000
8455	053310	140300	.WORD	140300
8456	053312	000300	.WORD	000300
8457	053314	140000	.WORD	140000
8458	053316	140300	.WORD	140300
8459	053320	000300	.WORD	000300
8460	053322	140000	.WORD	140000
8461	053324	140300	.WORD	140300
8462	053326	000300	.WORD	000300

8463	053330	140000	.WORD	140000
8464	053332	140300	.WORD	140300
8465	053334	000300	.WORD	000300
8466	053336	140000	.WORD	140000
8467	053340	140200	.WORD	140200
8468	053342	000300	.WORD	000300
8469	053344	140000	.WORD	140000
8470	053346	140300	.WORD	140300
8471	053350	000040	OPI4: .WORD	000040
8472	053352	140000	.WORD	140000
8473	053354	140000	.WORD	140000
8474	053356	000040	.WORD	000040
8475	053360	140000	.WORD	140000
8476	053362	140040	.WORD	140040
8477	053364	000040	.WORD	000040
8478	053366	140000	.WORD	140000
8479	053370	140040	.WORD	140040
8480	053372	000040	.WORD	000040
8481	053374	140000	.WORD	140000
8482	053376	140040	.WORD	140040
8483	053400	000040	.WORD	000040
8484	053402	140000	.WORD	140000
8485	053404	140040	.WORD	140040
8486	053406	000040	.WORD	000040
8487	053410	140000	.WORD	140000
8488	053412	140040	.WORD	140040
8489	053414	000040	.WORD	000040
8490	053416	140000	.WORD	140000
8491	053420	140040	.WORD	140040
8492	053422	000040	.WORD	000040
8493	053424	140000	.WORD	140000
8494	053426	140040	.WORD	140040
8495	053430	000040	.WORD	000040
8496	053432	140000	.WORD	140000
8497	053434	140040	.WORD	140040
8498	053436	000040	.WORD	000040
8499	053440	140000	.WORD	140000
8500	053442	140040	.WORD	140040
8501	053444	000040	.WORD	000040
8502	053446	140000	.WORD	140000
8503	053450	140040	.WORD	140040
8504	053452	000040	.WORD	000040
8505	053454	140000	.WORD	140000
8506	053456	140040	.WORD	140040
8507	053460	000040	.WORD	000040
8508	053462	140000	.WORD	140000
8509	053464	140040	.WORD	140040
8510	053466	000040	.WORD	000040
8511	053470	140000	.WORD	140000
8512	053472	140040	.WORD	140040
8513	053474	000040	.WORD	000040
8514	053476	140000	.WORD	140000
8515	053500	140040	.WORD	140040
8516	053502	000040	.WORD	000040
8517	053504	140000	.WORD	140000
8518	053506	140040	.WORD	140040

8519	053510	000040				.WORD	000040
8520	053512	140000				.WORD	140000
8521	053514	140040				.WORD	140040
8522	053516	000040				.WORD	000040
8523	053520	140000				.WORD	140000
8524	053522	140040				.WORD	140040
8525	053524	000040				.WORD	000040
8526	053526	140000				.WORD	140000
8527	053530	140040				.WORD	140040
8528	053532	000040				.WORD	000040
8529	053534	140000				.WORD	140000
8530	053536	140040				.WORD	140040
8531	053540	000040				.WORD	000040
8532	053542	140000				.WORD	140000
8533	053544	140040				.WORD	140040
8534	053546	000040				.WORD	000040
8535	053550	140000				.WORD	140000
8536	053552	140040				.WORD	140040
8537	053554	000040				.WORD	000040
8538	053556	140000				.WORD	140000
8539	053560	140040				.WORD	140040
8540	053562	000040				.WORD	000040
8541	053564	140000				.WORD	140000
8542	053566	140040				.WORD	140040
8543	053570	000040				.WORD	000040
8544	053572	140000				.WORD	140000
8545	053574	140040				.WORD	140040
8546	053576	000040				.WORD	000040
8547	053600	140000				.WORD	140000
8548	053602	140040				.WORD	140040
8549	053604	000040				.WORD	000040
8550	053606	140000				.WORD	140000
8551	053610	140040				.WORD	140040
8552	053612	000040				.WORD	000040
8553	053614	140000				.WORD	140000
8554	053616	140040				.WORD	140040
8555	053620	000040				.WORD	000040
8556	053622	140000				.WORD	140000
8557	053624	140040				.WORD	140040
8558	053626	000040				.WORD	000040
8559	053630	140000				.WORD	140000
8560	053632	140040				.WORD	140040
8561	053634	000040				.WORD	000040
8562	053636	140000				.WORD	140000
8563	053640	140040				.WORD	140040
8564	053642	000040				.WORD	000040
8565	053644	140000				.WORD	140000
8566	053646	140040				.WORD	140040
8567	053650	001253			VRCHDR:	.WORD	001253
8568	053652	141123				.WORD	141123
8569	053654	140370				.WORD	140370
8570	053656	177777	177777	177777	MOD2BF:	.WORD	177777,177777,177777,177777,177777,177777
8571	053664	177777	177777	177777			
8572	053672	000400			BUFF:	.BLKW	400
8573	054672	125252	125252	125252	ECCBUF:	.WORD	125252,125252,125252,125252,125252,125252
8574	054700	125252	125252	125252			

8575	054706	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8576	054714	177777	177777	177777		
8577	054722	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8578	054730	177777	000000	000000		
8579	054736	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8580	054744	000000	000000	000000		
8581	054752	052525	052525	052525	.WORD	052525,052525,052525,025252,052525,052525
8582	054760	025252	052525	052525		
8583	054766	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8584	054774	150442	064221	132110		
8585	055002	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8586	055010	105504	042642	021321		
8587	055016	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8588	055024	011055	104426	042213		
8589	055032	133333	066666	155555	.WORD	133333,066666,155555,155555,133333,066666
8590	055040	155555	133333	066666		
8591	055046	066666	155555	155555	.WORD	066666,155555,155555,133333,133333,133333
8592	055054	133333	133333	133333		
8593	055062	133333	133333	133333	.WORD	133333,133333,133333,133333,177777,177777
8594	055070	133333	177777	177777		
8595	055076	177777	052525	052525	.WORD	177777,052525,052525,052525,177777,177777
8596	055104	052525	177777	177777		
8597	055112	052525	052525	177777	.WORD	052525,052525,177777,052525,177252,177252
8598	055120	052525	177252	177252		
8599	055126	172765	172765	072307	.WORD	172765,172765,072307,135143,156461,167230
8600	055134	135143	156461	167230		
8601	055142	073514	035646	016723	.WORD	073514,035646,016723,107351,143564,061672
8602	055150	107351	143564	061672		
8603	055156	030735	114356	046167	.WORD	030735,114356,046167,123073,151453,164616
8604	055164	123073	151453	164616		
8605	055172	125252	125252	125252	.WORD	125252,125252,125252,125252,125252,125252
8606	055200	125252	125252	125252		
8607	055206	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8608	055214	177777	177777	177777		
8609	055222	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8610	055230	177777	000000	000000		
8611	055236	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8612	055244	000000	000000	000000		
8613	055252	052525	052525	052525	.WORD	052525,052525,052525,025252,052525,052525
8614	055260	025252	052525	052525		
8615	055266	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8616	055274	150442	064221	132110		
8617	055302	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8618	055310	105504	042642	021321		
8619	055316	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8620	055324	011055	104426	042213		
8621	055332	133333	066666	155555	.WORD	133333,066666,155555,155555,133333,066666
8622	055340	155555	133333	066666	.WORD	066666,155555,155555,133333,133333,133333
8623	055346	066666	155555	155555	.WORD	066666,155555,155555,133333,133333,133333
8624	055354	133333	133333	133333		
8625	055362	133333	133333	133333	.WORD	133333,133333,133333,133333,177777,177777
8626	055370	133333	177777	177777		
8627	055376	177777	052525	052525	.WORD	177777,052525,052525,052525,177777,177777
8628	055404	052525	177777	177777		
8629	055412	052525	052525	177777	.WORD	052525,052525,177777,052525,177252,177252
8630	055420	052525	177252	177252		

8631	055426	172765	172765	072307	.WORD	172765,172765,072307,135143,156461,167230
8632	055434	135143	156461	167230		
8633	055442	073514	035646	016723	.WORD	073514,035646,016723,107351,143564,061672
8634	055450	107351	143564	061672		
8635	055456	030735	114356	046167	.WORD	030735,114356,046167,123073,151453,164616
8636	055464	123073	151453	164616		
8637	055472	125252	125252	125252	.WORD	125252,125252,125252,125252,125252,125252
8638	055500	125252	125252	125252		
8639	055506	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8640	055514	177777	177777	177777		
8641	055522	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8642	055530	177777	000000	000000		
8643	055536	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8644	055544	000000	000000	000000		
8645	055552	052525	052525	052525	.WORD	052525,052525,052525,052525,052525,052525
8646	055560	025252	052525	052525		
8647	055566	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8648	055574	150442	064221	132110		
8649	055602	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8650	055610	105504	042642	021321		
8651	055616	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8652	055624	011055	104426	042213		
8653	055632	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8654	055640	177777	177777	177777		
8655	055646	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8656	055654	177777	177777	177777		
8657	055662	177777	177777	177777	.WORD	177777,177777,177777,177777
8658	055670	177777				
8659	055672	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8660	055700	177777	177777	177777		
8661		000001				

.END

ABASE = 177440	1002#	1237	1278	
ABORT 042430	6127	7376#		
ACDW1 = 000000	1237	1280		
ACDW2 = 000000	1237	1281		
ACLO = 000010	1097#			
ACPUOP= 000000	1237	1252		
ADDW0 = 000000	1237			
ADDW1 = 000000	1237			
ADDW10= 000000	1237			
ADDW11= 000000	1237			
ADDW12= 000000	1237			
ADDW13= 000000	1237			
ADDW14= 000000	1237			
ADDW15= 000000	1237			
ADDW2 = 000000	1237			
ADDW3 = 000000	1237			
ADDW4 = 000000	1237			
ADDW5 = 000000	1237			
ADDW6 = 000000	1237			
ADDW7 = 000000	1237			
ADDW8 = 000000	1237			
ADDW9 = 000000	1237			
ADEVCT= 000000	1237	1243		
ADEVM - 000000	1237	1279		
AENV = 000000	1237	1248		
AENVM - 000000	1237	1249		
AFATAL= 000000	1237	1240		
AMADR1= 000000	1237	1265		
AMADR2= 000000	1237	1269		
AMADR3= 000000	1237	1272		
AMADR4= 000000	1237	1275		
AMAMS1= 000000	1237	1259		
AMAMS2= 000000	1237	1267		
AMAMS3= 000000	1237	1270		
AMAMS4= 000000	1237	1273		
AMSGAD= 000000	1237	1245		
AMSGLG= 000000	1237	1246		
AMSGTY= 000000	1237	1239		
AMTYP1= 000000	1237	1260		
AMTYP2= 000000	1237	1268		
AMTYP3= 000000	1237	1271		
AMTYP4= 000000	1237	1274		
APASS = 000000	1237	1242		
APRIOR= 000240	1001#	1237		
APTCSU= 000040	6400#	6486		
APTENV= 000001	6356	6398#	6432	6479
APTSIZ 000200	2131	6397#		
APTSPD= 000100	6358	6399#	6481	
ASWREG= 000000	1237	1250		
ATESTN= 000000	1237	1241		
AUNIT 000000	1237	1244		
AUSWR = 000000	1237	1251		
AVECT1= 000210	1000#	1237	1276	
AVECT2= 000000	1237	1277		
BAI 000020	1060#			
BA*6 000400	1045#			

DT151	041456	2001	7175#												
ECCBUF	054672	5156	5218	5303	5349	5411	5440	5500	5683	5749	8573#				
ECCGEN	034526	5012	5126	5232	5425	5640	5767	5801	6202#						
ECCHI	003266	2063#	4994*	5108*	5213*	5250	5406*	5447	5619*	5746*	5821	6204	6210	6215*	
		6223	6224*	6225*	6226										
ECCLO	003270	2064#	4995*	5109*	5214*	5407*	5620*	5747*	6214*	6219	6220*	6221*			
ECCW =	020000	1121#	2887	2969	3420	3711	3762	3814	3865	3918	4022	4127	4253	4372	
		4871	4983	5097	5204	5242	5269	5397	5435	5466	5610	5737	5808	5840	
ECCXOR	003272	2065#	6206*	6212*	6216										
ECC1	052164	4823	4883	8162#											
ECC2	052204	4935	4996	8165#											
ECC3	052224	5049	5110	8173#											
ECH =	000100	1081#													
EMTVEC =	000030	995#	2102*	2103*											
EMW	001310	1305#	4865*	4881*	4977*	4993*	5091*	5107*	5201*	5215*	5247*	5274*	5394*	5408*	
		5444*	5471*	5604*	5625*	5651*	5731*	5752*	5778*	5785*	5816*	5819*	5846*	6255*	
		6269*	6296*	6309*											
EMW2	051447	6255	6296	8028#											
EMW4	051537	6269	6309	8038#											
EM000	044565	1300	7582#												
EM300	044632	1311	1317	1323	7589#										
EM301	044707	1329	1335	1341	7597#										
EM302	044765	1347	1353	1359	7605#										
EM303	045044	1365	1371	1377	7613#										
EM304	045122	383	1389	1395	7621#										
EM305	045201	1401	1407	1413	7629#										
EM306	045261	1420	1427	1434	7638#										
EM307	045363	1441	1448	1455	7650#										
EM308	045462	1462	1469	1475	7662#										
EM309	045562	1483	1490	1497	7674#										
EM310	045662	1503	1509	1515	1521	1527	1533	1539	1545	1551	7686#				
EM311	045747	1557	1563	7695#											
EM312	046016	1569	1575	7702#											
EM313	046063	1581	1587	7709#											
EM314	046127	1593	1599	7716#											
EM315	046176	1606	7723#												
EM316	046272	1612	1618	1624	7734#										
EM317	046337	1631	1638	1645	7741#										
EM318	046435	1651	1657	1663	7753#										
EM319	046474	1669	1675	1681	1687	7759#									
EM320	046545	1693	1699	1705	1711	7766#									
EM321	046631	1717	1723	1729	1735	7775#									
EM322	046715	1741	1747	1753	1759	7784#									
EM323	047000	1765	1771	1777	1783	7793#									
EM324	047073	1789	1795	1801	1807	7803#									
EM325	047166	1813	1819	1825	1831	7813#									
EM326	047256	1837	1843	1849	1855	7823#									
EM327	047344	4865	4977	5091	5201	5394	5604	5731	7832#						
EM328	047416	4881	4993	5107	5215	5408	7839#								
EM329	047475	1861	7847#												
EM330	047546	1867	7854#												
EM331	047611	1873	1879	1885	7860#										
EM332	047676	1891	1897	1903	7869#										
EM333	047763	5247	5444	5819	7878#										
EM334	050021	5274	5471	5846	7884#										
EM335	050057	1909	1915	1921	1927	1933	1939	1945	7890#						

EM336	050133	1951	7898#												
EM337	050210	1957	1963	1969	1975	1981	1987	1993	7906#						
EM338	050267	5625	5752	5785	7914#										
EM339	050355	5651	5778	5816	7924#										
EM340	050461	1999	2005	7936#											
EM4000	050526	1312	1330	1348	1366	1384	1402	1421	1442	1463	1484	1504	1558	1613	
		1632	1652	1670	1594	1718	1742	1766	1790	1814	1838	1874	1892	1910	
		1952	1958	7943#											
EM4001	050544	1318	1336	1354	1372	1390	1408	7946#							
EM4002	050565	1324	1342	1360	1378	1396	1414	7949#							
EM4003	050606	1428	1449	1470	1491	7952#									
EM4004	050636	1435	1456	1477	1498	7956#									
EM4005	050666	1510	1619	1639	1658	1676	1700	1724	1748	1772	1796	1820	1844	1880	
		1898	1916	1964	7960#										
EM4006	050704	1516	1625	1646	1664	1682	1706	1730	1754	1778	1802	1826	1850	1886	
		1904	1922	1970	7963#										
EM4007	050730	1522	1928	1976	7967#										
EM4008	050756	1528	1934	1982	7971#										
EM4009	051003	1534	7975#												
EM4010	051053	1540	7982#												
EM4011	051123	1546	7989#												
EM4012	051201	1552	7997#												
EM4013	051241	1564	1688	1712	1736	1760	1784	1808	1832	1856	8003#				
EM4014	051307	1570	1582	1594	1946	1994	8010#								
EM4015	051343	1576	1588	1600	1940	1988	8015#								
EM4016	051403	1607	8021#												
EM4017	051421	1862	1868	2000	2006	8024#									
ERRCNT	003250	2056#	2135*	6122*	6125										
ERRVEC-	000004	988#	2116	2117*	2128*	5955	5956*	5958*	5961*						
E.ASOF	003216	2038#													
E.BA	003204	2033#	2754*	2765	5303*	5314	5500*	5511	7143	7172					
E.CS1	003200	2031#	2247*	2250	2293*	2296	2340*	2343	2386*	2389	2432*	2435	2479*	2482	
		2514*	2526	2535	2571*	2583	2592	2625*	2637	2646	2680*	2692	2701	2714	
		2716*	2751*	2756	2786	2842*	2849	2857	2873	3287*	3290	3369*	3372	3417*	
		3458	3475*	3476*	3484*	3485	3505*	3570	3651*	3654	3708*	3748	3811*	3851	
		3915*	3955	4019*	4059	4124*	4165	4182*	4183*	4191*	4192	4250*	4291	4308*	
		4309*	4317*	4318	4369*	4410	4427*	4428*	4436*	4437	4511*	4514	4527*	4528	
		4601*	4604	4617*	4618	4691*	4694	4707*	4708	4781*	4784	4797*	4798	5294*	
		5305	5491*	5502	6185	7140	7143	7147	7152	7158	7162	7169			
E.CS2	003210	2035#	2752*	2762	2777*	2792	2804*	3288*	3293	3370*	3375	3418*	3461	3506*	
		3573	3652*	3657	3709*	3751	3812*	3854	3916*	3958	4020*	4062	4125*	4168	
		4251*	4294	4370*	4413	4512*	4517	4602*	4607	4692*	4697	4782*	4787	5295*	
		5308	5492*	5505	7143	7147	7158	7162	7169						
E.DA	003206	2034#	2513*	2525	2570*	2582	2605*	2606	2624*	2636	2659*	2660	2679*	2691	
		3044	3126	3207	5302*	5323	5499*	5520	6153*	6154*	6155	6157*	6158*	6159	
		6161*	6176	6183	7156	7172									
E.DB	003222	2040#	2778*	2798	7150										
E.DCVL	003220	2039#	2512*	2524	2548*	2549	2551*	2552	2569*	2581	2623*	2635	2678*	2690	
		3047	3129	3210	5301*	5320	5498*	5517	6152*	6162*	6168	7156	7172		
E.DS	003212	2036#													
E.ECPS	003232	2044#													
E.ECPT	003234	2045#	4882*	4899	5013	5127	5234	5427	5641	5768	5802	5858*	5860	6226*	
		6227*	7167	7175											
E.ER	003214	2037#	2755*	2771	2795	2809*	3296	3371*	3378	3419*	3464	3474*	3507*	3576	
		3653*	3660	3710*	3754	3813*	3857	3917*	3961	3969*	4021*	4065	4073*	4126*	
		4171	4181*	4252*	4297	4307*	4371*	4416	4426*	4513*	4520	4603*	4610	4693*	

	4024	4129	4255	4374	4471	4561	4651	4741	4832	4944	5058	5165	5358
	5567	5694											
M1.BIT 003256	2059#	2844*	2864*	2881*	2939*	2950*	2963*	3013*	3024*	3037*	3095*	3106*	3119*
	3176*	3187*	3200*	3255*	3266*	3279*	3337*	3348*	3361*	3427*	3437*	3450*	3538*
	3549*	3562*	3619*	3630*	3643*	3716*	3726*	3740*	3819*	3829*	3843*	3923*	3933*
	3947*	4027*	4037*	4051*	4134*	4144*	4157*	4260*	4270*	4283*	4379*	4389*	4402*
	4474*	4485*	4498*	4564*	4575*	4588*	4654*	4665*	4678*	4744*	4755*	4768*	4835*
	4846*	4859*	4868*	4887*	4888*	4947*	4958*	4971*	4980*	5000	5001*	5061*	5072*
	5085*	5094*	5114	5115*	5168*	5179*	5192*	5199*	5221	5222*	5253	5254*	5277
	5278*	5361*	5372*	5385*	5392*	5414	5415*	5450	5451*	5474	5475*	5571*	5585*
	5598*	5607*	5627	5628*	5698*	5712*	5725*	5734*	5754	5755*	5788	5789*	5825
	5826*	5849	5850*	6236	6282	6287	6328	7137					
M2.BIT 003260	2060#	4869*	4887*	4981*	5000*	5095*	5114*	5200*	5221*	5253*	5277*	5393*	5414*
	5450*	5474*	5608*	5627*	5735*	5754*	5788*	5825*	5849*	6241	6246	7137	
NED = 010000	1068#												
NEM = 004000	1067#												
NEWPAS 004300	2212#												
NPRBUF 051630	2739	2754	2774	8051#									
NXF = 000004	1077#												
OFFSET= 000015	1031#												
OFST = 000004	1096#												
OPI = 020000	1088#	3474	4181	4307	4426								
OPI1 052250	3416	8183#											
OPI2 052550	4123	8279#											
OPI3 053050	4249	8375#											
OPI4 053350	4368	8471#											
OPR001 042176	2162	7345#											
OPR002 042225	2165	2176	2192	7349#									
OPR003 042233	2172	7351#											
OPR004 042263	2183	7356#											
OPR005 042305	5922	7360#											
OPR006 042345	2159	7366#											
OR - 000200	1063#	2752	2777	3288	3370	3418	3506	3652	3709	3812	3916	4020	4125
	4251	4370	4512	4602									
PACK - 000003	1026#												
PARM 003316	1148	2076#											
PAT = 000020	1112#												
PCA = 004000	1119#	6233	6248	6284									
PCD 010000	1120#	6237	6243	6289									
PGE - 002000	1066#												
PIP 020000	1104#												
PIRQ = 177772	905#												
PIRQVE= 000240	999#												
PR.BIT 003254	2058#	2843*	2854*	2864	2867*	2870*	2881	2882*	2938*	2944*	2950	2953*	2956*
	2963	2964*	3012*	3018*	3024	3027*	3030*	3037	3038*	3094*	3100*	3106	3109*
	3112*	3119	3120*	3175*	3181*	3187	3190*	3193*	3200	3201*	3254*	3260*	3266
	3269*	3272*	3279	3280*	3336*	3342*	3348	3351*	3354*	3361	3362*	3426*	3432*
	3437	3440*	3443*	3450	3451*	3537*	3543*	3549	3552*	3555*	3562	3563*	3618*
	3624*	3630	3633*	3636*	3643	3644*	3715*	3721*	3726	3729*	3732*	3740	3741*
	3818*	3824*	3829	3832*	3835*	3843	3844*	3922*	3928*	3933	3936*	3939*	3947
	3948*	4026*	4032*	4037	4040*	4043*	4051	4052*	4133*	4139*	4144	4147*	4150*
	4157	4158*	4259*	4265*	4270	4273*	4276*	4283	4284*	4378*	4384*	4389	4392*
	4395*	4402	4403*	4473*	4479*	4485	4488*	4491*	4498	4499*	4563*	4569*	4575
	4578*	4581*	4588	4589*	4653*	4659*	4665	4668*	4671*	4678	4679*	4743*	4749*
	4755	4758*	4761*	4768	4769*	4834*	4840*	4846	4849*	4852*	4859	4860*	4867*
	4888	4889*	4946*	4952*	4958	4961*	4964*	4971	4972*	4979*	5001	5002*	5060*

	5066*	5072	5075*	5078*	5085	5086*	5093*	5115	5116*	5167*	5173*	5179	5182*
	5185*	5192	5193*	5198*	5222	5223*	5254	5255*	5278	5279*	5360*	5366*	5372
	5375*	5378*	5385	5386*	5391*	5415	5416*	5451	5452*	5475	5476*	5570*	5578*
	5585	5588*	5591*	5598	5599*	5606*	5628	5629*	5697*	5705*	5712	5715*	5718*
	5725	5726*	5733*	5755	5756*	5789	5790*	5826	5827*	5850	5851*	6234	6326
	7137												
PRO = 000000	922#	2212											
PR1 = 000040	923#												
PR2 = 000100	924#												
PR3 = 000140	925#												
PR4 = 000200	926#												
PR5 = 000240	927#	100*	2050										
PR6 = 000300	928#												
PR7 = 000340	929#	2087	7064	7078									
PS = 177776	902#	903											
PSW = 177776	903#												
PWRVEC = 000024	994#	2106*	2107*	7063*	7064*	7077*	7078*						
P1.BIT 003252	2057#	4866*	4877*	4889	4892*	4895*	4978*	4989*	5002	5005*	5008*	5092*	5103*
	5116	5119*	5122*	5197*	5210*	5223	5226*	5229*	5255	5258*	5261*	5279	5280*
	5390*	5403*	5416	5419*	5422*	5452	5455*	5458*	5476	5477*	5605*	5616*	5629
	5633*	5636*	5732*	5743*	5756	5760*	5763*	5790	5794*	5797*	5827	5830*	5833*
	5851	5852*	6202	6239	6280	7137							
RDBIT 035422	2847	2855	2871	2383	2941	2945	2957	2965	3015	3019	3031	3039	3097
	3101	3113	3121	3178	3182	3194	3202	3257	3261	3273	3281	3339	3343
	3355	3363	3429	3433	3444	3452	3540	3544	3556	3564	3621	3625	3637
	3645	3718	3722	3733	3742	3821	3825	3836	3845	3925	3929	3940	3949
	4029	4033	4044	4053	4136	4140	4151	4159	4262	4266	4277	4285	4381
	4385	4396	4404	4476	4480	4492	4500	4566	4570	4582	4590	4656	4660
	4672	4680	4744	4750	4762	4770	4837	4841	4853	4861	4949	4953	4965
	4973	5063	5067	5079	5087	5170	5174	5186	5194	5363	5367	5379	5387
	5574	5579	5592	5600	5701	5706	5719	5727	6326#				
RDCHR = 104410	6903	7128#											
RDDATA = 000021	1033#	2237	2247	2376	2386	2514	2571	2625	2680	4644	4691		
RDGATE = 100000	1123#	6233											
RDHEAD = 000025	1035#												
RDLIN = 104411	6977	7129#											
RDOCT = 104412	2166	2177	2193	7130#									
RDY = 000200	1044#	3476	3484	4183	4191	4309	4317	4428	4436	4511	4601	4691	4781
	5294	5491											
RECAL = 000013	1030#												
RESREG = 104414	6121	7132#											
RESTR = 003326	1146	2079#											
RESVEC = 000010	989#												
RKASOF = 000016	1013#												
RKBA = 000004	1008#	2233*	2279*	2326*	2372*	2418*	2465*	2523*	2580*	2634*	2689*	2739*	2749
	2832*	2923*	3002*	3084*	3165*	3244*	3326*	3404*	3525*	3608*	3695*	3798*	3902*
	4006*	4111*	4237*	4356*	4460*	4550*	4640*	4730*	4823*	4935*	5049*	5156*	5299
	5349*	5496	5556*	5683*									
RKCS1 = 000000	1006#	2230*	2237*	2244	2253*	2276*	2283*	2290	2299*	2323*	2330*	2337	2346*
	2369*	2376*	2383	2392*	2415*	2422*	2429	2438*	2462*	2469*	2476	2485*	2520*
	2526*	2532	2538*	2577*	2583*	2589	2595*	2631*	2637*	2643	2649*	2686*	2692*
	2698	2704*	2736*	2740*	2746	2759*	2783	2789*	2829*	2833*	2848	2856	2872
	2917*	2924*	2997*	3003*	3079*	3085*	3160*	3166*	3239*	3245*	3284	3321*	3327*
	3366	3399*	3408*	3455	3480*	3481	3513*	3529*	3567	3603*	3609*	3648	3693*
	3699*	3745	3796*	3802*	3848	3900*	3906*	3952	4004*	4010*	4056	4100*	4115*
	4162	4187*	4188	4232*	4241*	4288	4313*	4314	4351*	4360*	4407	4432*	4433

CZR6DD0 RK611 DSKLS PRT4
CZR6DD.P11 27-AUG-81 10:38

J 14
MACV11 30(1046) 28-AUG-81 10:56 PAGE 181
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ C178

.SSAVE	1#	867#	7011
.SSB2C	1#		
.SSB2O	1#		
.SSCOP	1#	867#	5934
.SSIZE	1#	867#	
.SSUPR	1#		
.STRAP	1#	867#	7088
.STYPB	1#		
.STYPD	1#	867#	6630
.STYPE	1#	867#	6456
.STYPO	1#	867#	6553
.S4OCA	1#		
.1170	1#		

. ABS. 055706 000

ERRORS DETECTED: 0

CZR6DD,CZR6DD.LST/SOL/CRF/NL: *OC=CZR6DD.SML,CZR6DD.P11
RUN-TIME: 28 32 2 SECONDS
RUN-TIME RATIO: 197/64=3.0
CORE USED: 43k (86 PAGES)