

RK611

DISKLESS CONT DIAG 3
MD-11-DZR6C-B

EP-DZR6C-B-DL-B

APR 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 2

MADE IN USA

This microfiche card contains 144 frames of technical information, arranged in a 12x12 grid. The frames display various diagrams, including:

- Block diagrams of control systems.
- Flowcharts and logic diagrams.
- Tables of data and parameters.
- Textual descriptions and labels.

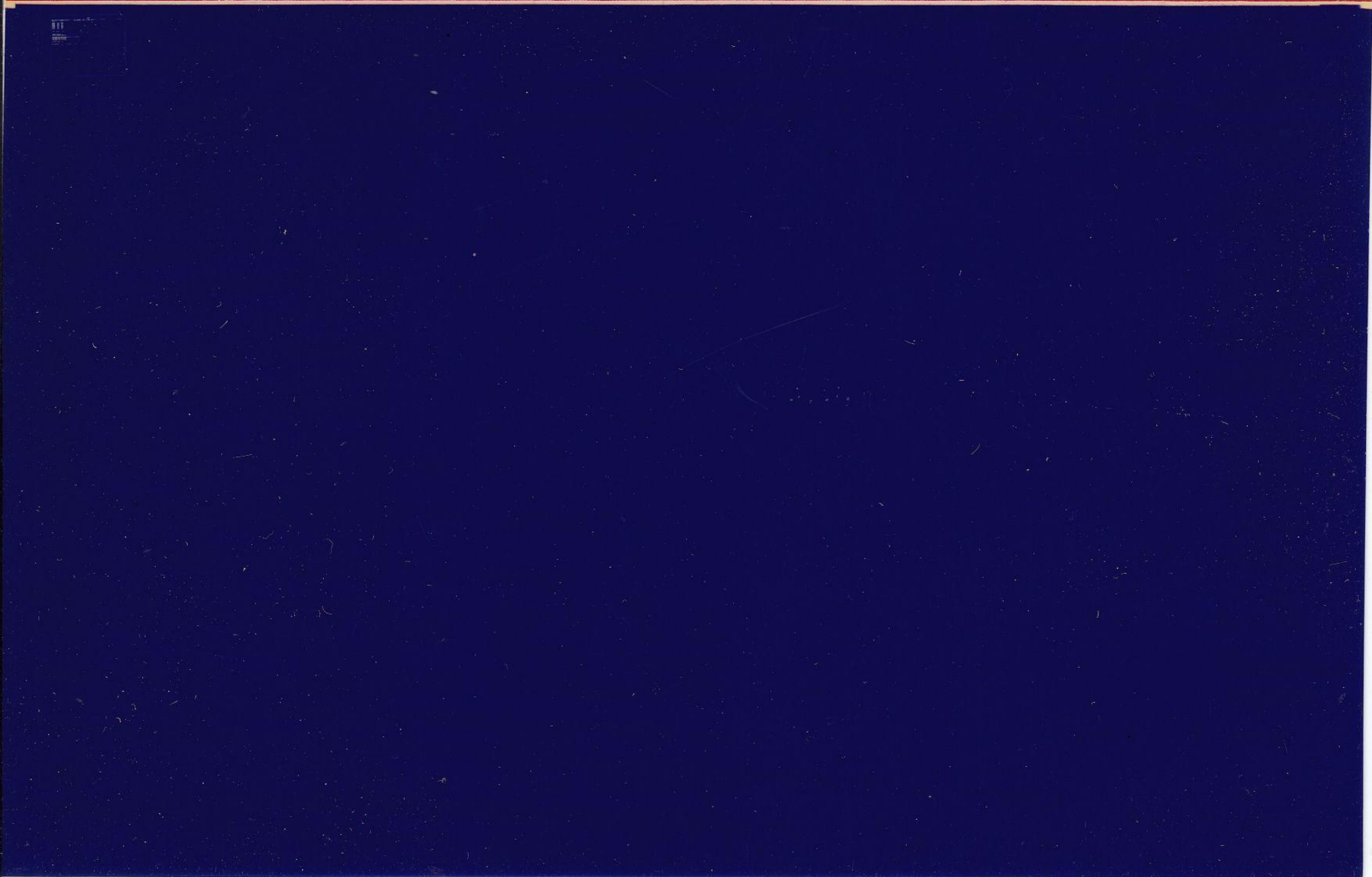
The diagrams are rendered in a light blue color on a dark background, typical of microfiche technology. The content is highly technical and specific to the MD-11-DZR6C-B system.

RK611

DISKLESS CONT DIAG 3
MD-11-DZR6C-B

EP-DZR6C-B-DL-B
COPYRIGHT © 1977
FICHE 2 OF 2

APR 1977
digital
MADE IN U.S.A.



B01

EOF1DZM9ACSEQ

00010000

770323

POP10 411

=1HDR1DZR6CBSEQ

00010000

770323

CO1

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04

MACY11 27(1006) 28-JAN-77 14:16 PAGE 1

SEQ 0001

.REM % IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZR6C-B-D
PRODUCT NAME: RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3
DATE: FEBRUARY, 1977
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: ROY SPITZER

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.

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

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

COPYRIGHT (C) 1976 & 1977 BY DIGITAL EQUIPMENT CORPORATION

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

001

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

31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3

- A. TESTS THE LOADING OF THE DRIVE BUS MESSAGE SHIFT REGISTER FOR CLASS B COMMANDS.
- B. TESTS INDEX AND SECTOR PULSE DETECTION.
- C. TESTS SILO AND MFR TRANSFERS FROM MEMORY IN 16 AND 18 BIT MODE.
- D. TESTS NON-EXISTANT MEMORY AND UNIBUS PARITY ERROR DETECTION.
- E. TESTS READ AND WRITE MFM LOOPBACK.
- F. TESTS CLASS B INSTRUCTION ERRORS.

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

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 1 (DZR6A)
 RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (DZR6B)

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

55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156

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

3.5 RUN TIME

FIRST PASS 30 SECONDS
 SUBSEQUENT PASSES 8:40 MINUTES

4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPRIATE ADDRESS.

4.1 'SOFTWARE' SWITCH REGISTER

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

SWR = NNNNNN NEW ='

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

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

4.2 CONTROL C (↑C) OPERATION

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

GO1

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 5
DZREC8.P11 28-JAN-77 14:04

SEQ 0005

157
158
159
160
161
162
163
164
165

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

**DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

TEST 1 READ HEADER SEEK MESSAGE

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE RK611 CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0 HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY THAT A SEEK IS LOADED WITH THE PROPER BITS IN MESSAGE SET. REPEAT FOR A READ HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.

TEST 2 WRITE HEADER SEEK MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY THAT A SEEK IS LOADED WITH THE RTC BIT SET. REPEAT FOR A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.

TEST 3 READ HEADER DRIVE CLEAR MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS GENERATED AND THE PROPER BITS ARE SET.

TEST 4 WRITE HEADER DRIVE CLEAR MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK SEEK MESSAGE AND LOAD GENERATED DRIVE CLEAR INTO SHIFT REGISTER. MAKE SURE THE DRIVE CLEAR IS GENERATED AND THE PROPER BITS ARE SET.

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

211
212
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
255
256
257
258
259

**INDEX AND SECTOR PULSE DETECT ON

TEST 5 SECTOR PULSE DETECT IN READ HEADER (PART 1)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES SET.

TEST 6 SECTOR PULSE DETECT IN READ HEADER (PART 2)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES NOT SET.

TEST 7 SECTOR PULSE DETECT IN READ HEADER (PART 3)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CHECK MESSAGES. SIMULATE 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES NOT SET.

TEST 10 INDEX PULSE DETECTION IN WRITE HEADER

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06, IN 26 SECTOR FORMAT, TO CYLINDER 0, HEAD 0, DRIVE 0, WITH A ONE WORD TRANSFER. CLOCK THROUGH THE SEEK AND THE DRIVE CLEAR MESSAGES. ISSUE 200 CONTROLLER CLOCKS AND MAKE SURE WRITE GATE DOES NOT SET. SIMULATE SECTOR PULSE AND 200 CONTROLLER CLOCKS MAKING SURE WRITE GATE DOES NOT SET. SIMULATE INDEX PULSE AND MAKE SURE WRITE GATE SETS.

260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312

**NPR READING OF MEMORY

TEST 11 NPR OUTPUT DATA TRANSFER

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 777, HEAD 7, DRIVE 7. SPECIFY A ONE WORD DATA TRANSFER. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE. CLOCK IN FIRST WORD OF NPR TRANSFER. CHECK INPUT READY, OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF THE SILO. REPEAT FOR 8 DIFFERENT DATA PATTERNS.

TEST 12 PARTIAL SILO FILLING

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A ONE WORD DATA TRANSFER. CLOCK IN ALL SPECIFIED WORDS INTO THE SILO. CHECK WORD COUNT, BUS ADDRESS, INPUT READY, AND OUTPUT READY. MAKE SURE NO MORE THAN SPECIFIED DATA LENGTH IS CLOCKED INTO THE SILO. CHECK THE SILO FOR CORRECT DATA. REPEAT FOR WORD COUNTS 2-65.

TEST 13 SILO FILLING WITH NPR TRANSFERS

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A 66 WORD DATA TRANSFER. CLOCK IN ALL 66 WORDS INTO THE SILO. CHECK INPUT READY, OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF THE SILO.

TEST 14 SILO CAPACITY WITH NPR TRANSFERS

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A 66 WORD DATA TRANSFER. CLOCK IN 66 WORDS INTO THE SILO. MAKE SURE THAT SILO WILL STOP FILLING AT 66 WORDS. TAKE ONE WORD FROM SILO AND CHECK IT. CLOCK IN NEXT WORD. MAKE SURE NO MORE THAN ONE WORD IS CLOCKED IN THE SILO. TAKE ONE WORD FROM SILO AND CHECK IT. CLOCK IN NEXT WORD. MAKE SURE NO MORE THAN ONE WORD IS CLOCKED IN THE SILO. TAKE ONE WORD FROM SILO AND CHECK IT. ATTEMPT TO CLOCK IN NEXT WORD AND

K01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04

MACY11 27(1006) 28-JAN-77 14:16 PAGE 9

SEQ 0009

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
367

MAKE SURE NO WORDS ARE CLOCK INTO SILO. UNLOAD THE
SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.

TEST 15 BUS ADDRESS INHIBIT

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN
RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
SPECIFY A 66 WORD DATA TRANSFER WITH BUS ADDRESS
INHIBIT INCREMENT. CHECK WORD COUNT, BUS ADDRESS,
INPUT READY, OUTPUT READY, AND MAKE SURE ALL THE
WORDS IN THE SILO ARE THE CORRECT SAME WORD.

TEST 16 NON-EXISTENT MEMORY

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS
(760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR
OCCURS IN THE RK611 CONTROLLER.

TEST 17 BUS ADDRESS BIT 16

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
SPECIFY A ONE WORD DATA TRANSFER FROM 200000.
READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.
CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
OF MEMORY IS ON THE SYSTEM.

TEST 20 BUS ADDRESS BIT 17

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
SPECIFY A ONE WORD DATA TRANSFER FROM 400000.
READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 377776.
CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
OF MEMORY IS ON THE SYSTEM.

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
423

TEST 21 ADDRESSING GREATER THAN 96K

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0. SPECIFY A ONE WORD DATA TRANSFER FROM 600000. READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ. REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 577776. CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K OF MEMORY IS ON THE SYSTEM.

TEST 22 UNIBUS PARITY ERROR

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A ONE WORD DATA TRANSFER FROM A LOCATION WITH BAD PARITY. MAKE SURE A UNIBUS PARITY ERROR OCCURS.

REPEAT FOR A ONE WORD DATA TRANSFER FROM THE LOCATION PRIOR TO THE LOCATION WITH BAD PARITY. MAKE SURE UNIBUS PARITY ERROR DOES NOT OCCUR. REPEAT FOR A ONE WORD DATA TRANSFER FROM THE LOCATION AFTER THE LOCATION WITH BAD PARITY. MAKE SURE UNIBUS PARITY ERROR DOES NOT OCCUR.

REPEAT FOR A TWO WORD DATA TRANSFER STARTING WITH THE ADDRESS PRIOR TO THE LOCATION WITH BAD PARITY. MAKE SURE UNIBUS PARITY ERROR DOES OCCUR.

NOTE: THIS TEST IS EXECUTED ONLY IF MEMORY PARITY EXISTS FOR SPECIFIED LOCATION AND IS NOT RUN ON AN 11/70.

TEST 23 SILO FILL IN 18 BIT MODE

CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIFY 66 WORD DATA TRANSFER. CLOCK ALL 66 WORD INTO THE SILO. CHECK THAT ALL 66 WORDS ARE CORRECT (16 LEAST SIGNIFICANT BITS).

TEST 24 BIT 16 AND 17 READING WITH NPR

CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.

MO1

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04

MACY11 27(1006) 28-JAN-77 14:16 PAGE 11

SEQ 0011

SPECIFY ONE WORD DATA TRANSFER FROM A LOCATION

WITH BAD PARITY. MAKE SURE A UNIBUS PARITY
DOES NOT OCCUR.

NOTE: THIS TEST IS EXECUTED ONLY IF MEMORY PARITY
ENABLE EXISTS FOR SPECIFIED LOCATION AND
IS NOT RUN ON AN 11/70.

**MFM READ LOOPBACK TESTS

TEST 25 READ LOOPBACK (PART 1)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE, 255 ZEROES, A
ONE, AND A HEADER CONSISTING OF THE THREE
FOLLOWING WORDS:

177777
000000
177777

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 26 READ LOOPBACK (PART 2)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE, 255 ZEROES, A ONE,
AND A HEADER CONSISTING OF THE THREE
FOLLOWING WORDS:

000000
177777
000000

MAKE SURE THAT READY COMES UP AFTER THIRD WORD
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

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

TEST 27 READ LOOPBACK (PART 3)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER J, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS:

125252
052525
125252

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 30 READ LOOPBACK (PART 4)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 225 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS:

044444
022222
111111

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 31 READ LOOPBACK (PART 5)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS.

052012
100520
052012

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

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

TEST 32 READ HEADER IN 18 BIT MODE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE, 255 ZEROES, A
ONE, AND A HEADER CONSISTING OF THE THREE
FOLLOWING WORDS:

177777
000000
177777

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 33 SYNCH DETECT IN READ HEADER

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE AND 350 ZEROES. MAKE
SURE READY REMAINS RESET AND THE SILO REMAINS
EMPTY.

TEST 34 ZERO SYNCH ON READ

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF
BIT TIME, A ONE, AND A HEADER CONSISTING OF THE
THREE FOLLOWING WORDS:

177777
000000
177777

MAKE SURE THAT READY COMES AFTER THE THIRD WORD
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626

**MFM WRITE LOOPBACK TESTS

TEST 35 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE AND 500 DATA BITS. MAKE SURE THAT ZEROES ARE WRITTEN. SIMULATE SECTOR PULSE AND MAKE SURE WRITE GATE RESETS.

TEST 36 WRITE LOOPBACK (PART 1)

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

177777
000000
177777

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 37 WRITE LOOPBACK (PART 2)

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

000000
177777
000000

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678

TEST 40 WRITE LOOPBACK (PART 3)

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

125252
052525
125252

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 41 WRITE LOOPBACK (PART 4)

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

044444
022222
111111

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MOD CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 42 WRITE LOOPBACK (PART 5)

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

052012
100520
052012

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730

TEST 43 WRITE LOOPBACK (PART 6)

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

155555
066666
155555

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 44 WRITE LOOPBACK (PART 7)

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

104210
104210
104210

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 45 WRITE TWO HEADERS

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

177777
000000
177777

FOLLOW THAT BY A SECTOR PULSE AND ONE THREE WORD HEADER CONSISTING OF THE FOLLOWING DATA:

731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778000000
177777
000000SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

TEST 46 DATA FIELD FILLING ON WRITE HEADER

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND
SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE
FOLLOWING DATA:125252
052525
125252
052525
125252
052525MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
ECC FIELD ARE WRITTEN CORRECTLY.

TEST 47 WRITE HEADER FOR 26 SECTORS

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIF
66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTL

TEST 50 WRITE HEADER IN 24 SECTOR FORMAT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER
OF SIX WORDS IN 24 SECTOR FORMAT TO AN RK06, CYLINDER 0,
HEAD 0, DRIVE 0. CLOCK THROUGH THE SEEK AND DRIVE CLEAR
MESSAGES, SIMULATE INDEX PULSE, SECTOR PULSE, 3 HEADER
WORDS, SYNCH AND DATA, ANOTHER SECTOR PULSE, 3 HEADER
WORDS, AND AN INDEX PULSE. CHECK DATA WRITTEN TO MAKE
SURE ONLY LOW 16 BITS OF SILO ARE USED.

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

**TYPE B INSTRUCTION ERRORS

TEST 51 FORMAT ERROR (PART 1)

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO A RK06, IN 26 SECTOR FORMAT, CYLINDER 43, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF MAINTENANCE MODE. MAKE SURE FORMAT ERROR, DRIVE AVAILABLE AND CONTROLLER ERROR SET.

TEST 52 FORMAT ERROR (PART 2)

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO A RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF MAINTENANCE MODE. MAKE SURE FORMAT ERROR, DRIVE AVAILABLE AND CONTROLLER ERROR SET.

TEST 53 FAULT SETTING CONTROLLER ERROR

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE AVAILABLE AND CONTROLLER ERROR SET.

813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION

```

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

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

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

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.

```

867                                     %
868      .NLIST CND,MD,MC,TOC
869      .LIST ME
870      .ENABL ABS,AMA
871      $SWR= 167400
872      $TM= 1
873      .TITLE RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
874      *COPYRIGHT (C) 1976 & 1977
875      *DIGITAL EQUIPMENT CORP.
876      *MAYNARD, MASS. 01754
877      *
878      *PROGRAM BY ROY SPITZER
879      *
880      *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
881      *PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
882      *
883      .SBTTL OPERATIONAL SWITCH SETTINGS
884      *
885      *          SWITCH          USE
886      *          -----
887      *          15          HALT ON ERROR
888      *          14          LOOP ON TEST
889      *          13          INHIBIT ERROR TYPEOUTS
890      *          12          ABORT PROGRAM AFTER 20 ERRORS
891      *          11          INHIBIT ITERATIONS
892      *          10          BELL ON ERROR
893      *          9           LOOP ON ERROR
894      *          8           LOOP ON TEST IN SWR<7:0>
895      .SBTTL BASIC DEFINITIONS
896      *
897      *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
898      $STAC.= 1100
899      .EQUIV EMT,ERROR          ;;BASIC DEFINITION OF ERROR CALL
900      .EQUIV IOT,SCOPE         ;;BASIC DEFINITION OF SCOPE CALL
901      *
902      *MISCELLANEOUS DEFINITIONS
903      HT= 11                    ;;CODE FOR HORIZONTAL TAB
904      LF= 12                    ;;CODE FOR LINE FEED
905      CR= 15                    ;;CODE FOR CARRIAGE RETURN
906      CRLF= 200                 ;;CODE FOR CARRIAGE RETURN-LINE FEED
907      PS= 177776                ;;PROCESSOR STATUS WORD
908      .EQUIV PS,PSW
909      STKLM= 177774              ;;STACK LIMIT REGISTER
910      PIRQ= 177772               ;;PROGRAM INTERRUPT REQUEST REGISTER
911      OSWR= 177570              ;;HARDWARE SWITCH REGISTER
912      DDISP= 177570            ;;HARDWARE DISPLAY REGISTER
913      *
914      *GENERAL PURPOSE REGISTER DEFINITIONS
915      R0= %0                     ;;GENERAL REGISTER
916      R1= %1                     ;;GENERAL REGISTER
917      R2= %2                     ;;GENERAL REGISTER
918      R3= %3                     ;;GENERAL REGISTER
919      R4= %4                     ;;GENERAL REGISTER
920      R5= %5                     ;;GENERAL REGISTER
921      R6= %6                     ;;GENERAL REGISTER
922      R7= %7                     ;;GENERAL REGISTER

```

```

923      000006      SP=      %6      :::STACK POINTER
924      000007      PC=      %7      :::PROGRAM COUNTER
925
926      .: *PRIORITY LEVEL DEFINITIONS
927      000000      PR0=      0      :::PRIORITY LEVEL 0
928      000040      PR1=      40     :::PRIORITY LEVEL 1
929      000100      PR2=     100     :::PRIORITY LEVEL 2
930      000140      PR3=     140     :::PRIORITY LEVEL 3
931      000200      PR4=     200     :::PRIORITY LEVEL 4
932      000240      PR5=     240     :::PRIORITY LEVEL 5
933      000300      PR6=     300     :::PRIORITY LEVEL 6
934      000340      PR7=     340     :::PRIORITY LEVEL 7
935
936      .: *"SWITCH REGISTER" SWITCH DEFINITIONS
937      100000      SW15=    100000
938      040000      SW14=     40000
939      020000      SW13=     20000
940      010000      SW12=     10000
941      004000      SW11=     4000
942      002000      SW10=     2000
943      001000      SW09=     1000
944      000400      SW08=     400
945      000200      SW07=     200
946      000100      SW06=     100
947      000040      SW05=     40
948      000020      SW04=     20
949      000010      SW03=     10
950      000004      SW02=     4
951      000002      SW01=     2
952      000001      SW00=     1
953      .EQUIV     SW09, SW9
954      .EQUIV     SW08, SW8
955      .EQUIV     SW07, SW7
956      .EQUIV     SW06, SW6
957      .EQUIV     SW05, SW5
958      .EQUIV     SW04, SW4
959      .EQUIV     SW03, SW3
960      .EQUIV     SW02, SW2
961      .EQUIV     SW01, SW1
962      .EQUIV     SW00, SW0
963
964      .: *DATA BIT DEFINITIONS (BIT00 TO BIT15)
965      100000      BIT15=    100000
966      040000      BIT14=     40000
967      020000      BIT13=     20000
968      010000      BIT12=     10000
969      004000      BIT11=     4000
970      002000      BIT10=     2000
971      001000      BIT09=     1000
972      000400      BIT08=     400
973      000200      BIT07=     200
974      000100      BIT06=     100
975      000040      BIT05=     40
976      000020      BIT04=     20
977      000010      BIT03=     10
978      000004      BIT02=     4

```

BASIC DEFINITIONS

```

979          000002      BIT01= 2
980          000001      BIT00= 1
981          .EQUIV     BIT09,BIT9
982          .EQUIV     BIT08,BIT8
983          .EQUIV     BIT07,BIT7
984          .EQUIV     BIT06,BIT6
985          .EQUIV     BIT05,BIT5
986          .EQUIV     BIT04,BIT4
987          .EQUIV     BIT03,BIT3
988          .EQUIV     BIT02,BIT2
989          .EQUIV     BIT01,BIT1
990          .EQUIV     BIT00,BIT0
991
992          .: *BASIC "CPU" TRAP VECTOR ADDRESSES
993          000004      ERRVEC= 4          .: TIME OUT AND OTHER ERRORS
994          000010      RESVEC= 10         .: RESERVED AND ILLEGAL INSTRUCTIONS
995          000014      TBITVEC=14         .: "T" BIT
996          000014      TRIVEC= 14         .: TRACE TRAP
997          000014      BPTVEC= 14         .: BREAKPOINT TRAP (BPT)
998          000020      IOTVEC= 20         .: INPUT/OUTPUT TRAP (IOT) **SCOPE**
999          000024      PWAVEC= 24         .: POWER FAIL
1000         000030      EMTVEC= 30         .: EMULATOR TRAP (EMT) **ERROR**
1001         000034      TRAPVEC=34         .: "TRAP" TRAP
1002         000060      TKVEC= 60          .: TTY KEYBOARD VECTOR
1003         000064      TPVEC= 64          .: TTY PRINTER VECTOR
1004         000240      PIRQVEC=240       .: PROGRAM INTERRUPT REQUEST VECTOR
1005
1006         .SBTTL     MEMORY MANAGEMENT DEFINITIONS
1007
1008         ; *KT11 VECTOR ADDRESS
1009         000250      MMVEC= 250
1010
1011         ; *KT11 STATUS REGISTER ADDRESSES
1012
1013         177572      SR0= 177572
1014         177574      SR1= 177574
1015         177576      SR2= 177576
1016         172516      SR3= 172516
1017
1018         ; *KERNEL "I" PAGE DESCRIPTOR REGISTERS
1019
1020         172300      KIPOR0= 172300
1021         172302      KIPOR1= 172302
1022         172304      KIPOR2= 172304
1023         172306      KIPOR3= 172306
1024         172310      KIPOR4= 172310
1025         172312      KIPOR5= 172312
1026         172314      KIPOR6= 172314
1027         172316      KIPOR7= 172316
1028
1029         ; *KERNEL "I" PAGE ADDRESS REGISTERS
1030
1031         172340      KIPAR0= 172340
1032         172342      KIPAR1= 172342
1033         172344      KIPAR2= 172344
1034         172346      KIPAR3= 172346

```

```

1035      172350      KIPAR4= 172350
1036      172352      KIPAR5= 172352
1037      172354      KIPAR6= 172354
1038      172356      KIPAR7= 172356
1039
1040      000114      MEMVEC= 114      ;MEMORY PARITY VECTOR
1041      172100      MEMBAS= 172100  ;MEM PARITY OPTION
1042      000004      WR.PAR= 4      ;WRITE BAD PARITY
1043      000001      PAR.EN= 1      ;ENABLE PARITY ENABLE
1044      120210      AVECT1= 120210 ;DEFINE RK611 VECTOR ADDRESS
1045      000005      APRIOR= 5      ;DEFINE RK611 PRIORITY
1046      177440      ABASE= 177440 ;DEFINE BASE OF RK611 REGISTERS
1047
1048      .SBTTL      RK611 CONTROLLER REGISTER DEFINITION
1049
1050      000000      RKCS1= 0      ;CONTROL AND STATUS REGISTER 1
1051      000002      RKWC= 2      ;WORD COUNT REGISTER
1052      000004      RKBA= 4      ;BUS ADDRESS REGISTER
1053      000006      RKDA= 6      ;DESIRED TRACK SECTOR REGISTER
1054      000010      RKCS2= 10     ;CONTROL AND STATUS REGISTER 2
1055      000012      RKDS= 12     ;DRIVE STATUS REGISTER
1056      000014      RKER= 14     ;ERROR REGISTER
1057      000016      RKASOF= 16    ;ATTENTION SUMMARY AND OFFSET REGISTER
1058      000020      RKDCYL= 20    ;DESIRED CYLINDER REGISTER
1059      000024      RKDB= 24     ;DATA BUFFER
1060      000026      RKMRI= 26    ;MAINTENANCE REGISTER 1
1061      000034      RKMRI2= 34   ;MAINTENANCE REGISTER 2
1062      000036      RKMRI3= 36   ;MAINTENANCE REGISTER 3
1063      000030      RKECPS= 30   ;ECC POSITION INFORMATION
1064      000032      RKECPT= 32   ;ECC PATTERN INFORMATION
1065      000022      RKSPAR= 22   ;SPARE REGISTER
1066
1067      .SBTTL      DRIVE COMMANDS
1068
1069      000001      SELDRV= 01    ;SELECT DRIVE
1070      000003      PACK= 03    ;PACK ACKNOWLEDGE
1071      000005      CLEAR= 05    ;DRIVE CLEAR
1072      000007      UNLOAD= 07   ;UNLOAD
1073      000011      SRTSPL= 11   ;START SPINDLE
1074      000013      RECAL= 13   ;RECALIBRATE
1075      000015      OFFSET= 15   ;OFFSET
1076      000017      SEEK= 17     ;SEEK
1077      000021      RDDATA= 21   ;READ DATA
1078      000023      WRDATA= 23   ;WRITE DATA
1079      000025      RDHEAD= 25   ;READ HEADER
1080      000027      WRHEAD= 27   ;WRITE HEADER AND DATA
1081      000031      WRTCHK= 31   ;WRITE CHECK
1082      000300      INTR= 300    ;GENERATE INTERRUPT TO CPU
1083
1084      .SBTTL      CONTROL AND STATUS REGISTER 1 BITS
1085
1086      000001      GO= BIT0      ;GO BIT
1087      000100      IE= BIT6      ;INTERRUPT ENABLE
1088      000200      RDY= BIT7      ;CONTROLLER READY
1089      000400      BA16= BIT8     ;BUS ADDRESS BIT 16
1090      001000      BA17= BIT9     ;BUS ADDRESS BIT 17

```

M02

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 24
 DZR6CB.P11 28-JAN-77 14:04 CONTROL AND STATUS REGISTER 1 BITS

SEQ 0024

1091	002000	CDT=	BIT10	: CONTROLLER DRIVE TYPE (0=RK06)
1092	004000	CTO=	BIT11	: CONTROLLER TIMED OUT WAITING FOR
1093				: DRIVE RESPONSE
1094	010000	CFMT=	BIT12	: CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1095	020000	SPAR=	BIT13	: DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1096	040000	DI=	BIT14	: DRIVE INTERRUPT
1097	100000	CERR=	BIT15	: CONTROLLER ERROR
1098	100000	CCLR=	BIT15	: CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1101				
1102	000007	DRVMSK=	7	: MASK FOR DRIVE SELECTION CODE
1103	000010	RLS=	BIT3	: DESELECT OR RELEASE DRIVE IN BITS 0-2
1104	000020	BAI=	BIT4	: BUS ADDRESS INCREMENT INHIBIT
1105	00004C	SCLR=	BITS	: CLEAR CONTROLLER AND ALL DRIVES
1106	000100	IR=	BIT6	: INPUT READY
1107	000200	OR=	BIT7	: OUTPUT READY
1108	000400	UFE=	BIT8	: UNIT FIELD ERROR
1109	001000	MDS=	BIT9	: MULTIPLE DRIVE SELECT
1110	002000	PGE=	BIT10	: PROGRAMMING ERROR
1111	004000	NEM=	BIT11	: NON-EXISTENT MEMORY
1112	010000	NED=	BIT12	: NON-EXISTENT DRIVE
1113	020000	UPE=	BIT13	: UNIBUS PARITY ERROR
1114	040000	WCE=	BIT14	: WRITE CHECK ERROR
1115	100000	DLT=	BIT15	: DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1116				
1117				
1118				
1119	000001	ILF=	BIT0	: ILLEGAL FUNCTION CODE
1120	000002	SKI=	BIT1	: SEEK INCOMPLETE
1121	000004	NXF=	BIT2	: NON-EXECUTABLE DRIVE FUNCTION
1122	000010	DRPAR=	BIT3	: DRIVE DETECTED DRIVE BUS PARITY ERROR
1123	000020	FMTE=	BIT4	: FORMAT ERROR
1124	000040	DTYPE=	BITS	: DRIVE TYPE ERROR
1125	000100	ECH=	BIT6	: ECC HARD
1126	000200	BSE=	BIT7	: BAD SECTOR ERROR
1127	000400	HVRC=	BIT8	: HEADER VRC ERROR
1128	001000	COE=	BIT9	: CYLINDER ADDRESS OVERFLOW ERROR
1129	002000	IDAE=	BIT10	: INVALID DISK ADDRESS ERROR
1130	004000	WLE=	BIT11	: WRITE LOCK ERROR
1131	010000	DTE=	BIT12	: DRIVE TIMING ERROR
1132	020000	OPI=	BIT13	: OPERATION (SEARCH) INCOMPLETE
1133	040000	UNS=	BIT14	: DRIVE UNSAFE
1134	100000	DCK=	BIT15	: DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1135				
1136				
1137				
1138	000001	DRA=	BIT0	: DRIVE AVAILABLE (CONTROLLER IS SET IF
1139				: THIS BIT IS RESET)
1140	000004	OFST=	BIT2	: DRIVE OFFSET
1141	000010	ACLO=	BIT3	: AC LOW
1142	000020	SPDLSS=	BIT4	: SPEED LOSS
1143	000040	DROT=	BITS	: DRIVE OFF TRACK
1144	000100	VV=	BIT6	: VOLUME VALID
1145	000200	DRDY=	BIT7	: DRIVE READY
1146	000400	DDT=	BIT8	: DRIVE TYPE (0=RK06)

```

1147      004000      WRL=      BIT11      ;WRITE LOCK
1148      020000      PIP=      BIT13      ;POSITIONING IN PROGRESS
1149      040000      DSC=      BIT14      ;DRIVE STATUS CHANGE
1150      100000      SVAL=      BIT15      ;STATUS VALID
1151
1152      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1153
1154      000017      MESMSK= 17      ;MESSAGE MASK
1155
1156      000020      PAT=      BIT4      ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1157      000040      DMD=      BITS      ;DIAGNOSTIC MODE
1158      000100      MSP=      BIT6      ;MAINTENANCE SECTOR PULSE
1159      000200      MIND=      BIT7      ;MAINTENANCE INDEX
1160      000400      MCLK=      BIT8      ;MAINTENANCE CLOCK
1161      001000      MERD=      BIT9      ;MAINTENANCE ENCODED READ DATA
1162      002000      MEWD=      BIT10     ;MAINTENANCE ENCODED WRITE DATA
1163      004000      PCA=      BIT11     ;PRECOMPENSATION ADVANCE
1164      010000      PCD=      BIT12     ;PRECOMPENSATION DELAY
1165      020000      ECCW=      BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1166      040000      WRTGAT= BIT14     ;WRITE GATE
1167      100000      ROGATE= BIT15     ;READ GATE
1168
1169      .SBTTL  TRANSMITTED MESSAGE A
1170
1171      000020      S. SEEK= BIT4      ;SEEK COMMAND
1172      000040      S. RECL= BITS      ;RECALIBRATE COMMAND
1173      000100      S. STSP= BIT6      ;START SPINDLE COMMAND
1174      000200      S. RTC=  BIT7      ;DRIVE RETURN TO CENTERLINE COMMAND
1175      000400      S. CLB=  BIT8      ;CLEAR ERROR AND DSC
1176      001000      S. FAT=  BIT9      ;FORMAT
1177      002000      S. UNLD= BIT10     ;UNLOAD
1178      004000      S. PACK= BIT11     ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1179
1180      .SBTTL  TRAP CATCHER
1181
1182      .=0
1183      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1184      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1185      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1186      000174      000000      .=174
1187      000176      000000      DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
1188      .SBTTL  STARTING ADDRESS(ES)
1189      000200      000137      003662      JMP      JSTART ;; JUMP TO STARTING ADDRESS OF PROGRAM
1190      000204      000137      003652      JMP      RESTRT  ;; JUMP TO RESTART ROUTINE
1191
1192      000214      000137      003642      .=214
1193      JMP      PARM      ;JUMP TO OPERATOR ASSIGNED PARMETERS
1194
1195      .SBTTL  ACT11 HOOKS
1196
1197      ;*****
1198      ;HOOKS REQUIRED BY ACT11
1199      $SVPC=.      ;SAVE PC
1200      .=46
1201      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1202      .=52
1203      .WORD 0      ;;2)SET LOC.52 TO ZERO
1204      .=$SVPC      ;; RESTORE PC

```

1203
1204 000114
1205 000116
1206
1207
1208
1209
1210
1211
1212 001000
1213 000024
1214 000024 000200
1215 000044
1216 000044 001000
1217 001000
1218
1219
1220
1221
1222 001000
1223 001000 000030
1224 001002 001214
1225 001004 000000
1226 001006 000000
1227 001010 000000
1228 001012 000032

```

.=MEMVEC
MEMERR
PR7
.=1000
.SBTTL APT PARAMETER BLOCK

;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.SX=      ;SAVE CURRENT LOCATION
.=24     ;SET POWER FAIL TO POINT TO START OF PROGRAM
200      ;FOR APT START UP
.=44     ;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR  ;POINT TO APT HEADER BLOCK
.=.SX    ;RESET LOCATION COUNTER

;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

$APTHD:
$HIBTS: .WORD 0      ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBAADR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT: .WORD        ;RUN TIM OF LONGEST TEST
$PASTM: .WORD        ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD        ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD SETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

1229
1230
1231
1232
1233
1234
1235
1236 001100
1237 001100 000000
1238 001102 000
1239 001103 000
1240 001104 000000
1241 001106 000000
1242 001110 000000
1243 001112 000000
1244 001114 000
1245 001115 001
1246 001116 000000
1247 001120 000000
1248 001122 000000
1249 001124 000000
1250 001126 000000
1251 001130 000000
1252 001132 000000
1253 001134 000
1254 001135 000
1255 001136 000000
1256 001140 177570
1257 001142 177570
1258 001144 177560
1259 001146 177562
1260 001150 177564
1261 001152 177566
1262 001154 000
1263 001155 002
1264 001156 012
1265 001157 000
1266 001160 000000
1267 001162 000000
1268 001164 000000
1269 001166 000000
1270 001170 000000
1271 001172 000000
1272 001174 000000
1273 001176 000000
1274 001200 000000
1275 001202 000000
1276 001204 177607 000377
1277 001210 077
1278 001211 015
1279 001212 000012
1280
1281
1282
1283
1284

.SBTTL COMMON TAGS

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

SCMTAG: . =1100

; START OF COMMON TAGS

.WORD 0
\$TSTNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0

CONTAINS THE TEST NUMBER
CONTAINS ERROR FLAG
CONTAINS SUBTEST ITERATION COUNT
CONTAINS SCOPE LOOP ADDRESS
CONTAINS SCOPE RETURN FOR ERRORS
CONTAINS TOTAL ERRORS DETECTED
CONTAINS ITEM CONTROL BYTE
CONTAINS MAX. ERRORS PER TEST
CONTAINS PC OF LAST ERROR INSTRUCTION
CONTAINS ADDRESS OF 'GOOD' DATA
CONTAINS ADDRESS OF 'BAD' DATA
CONTAINS 'GOOD' DATA
CONTAINS 'BAD' DATA
RESERVED--NOT TO BE USED

\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0

AUTOMATIC MODE INDICATOR
INTERRUPT MODE INDICATOR

\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP

ADDRESS OF SWITCH REGISTER
ADDRESS OF DISPLAY REGISTER

\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$STPFLG: .BYTE 0
\$TMP0: .WORD 0
\$TMP1: .WORD 0
\$TMP2: .WORD 0
\$TMP3: .WORD 0
\$TMP4: .WORD 0
\$TMP5: .WORD 0
\$TMP6: .WORD 0
\$TMP7: .WORD 0

TTY KBD STATUS
TTY KBD BUFFER
TTY PRINTER STATUS REG. ADDRESS
TTY PRINTER BUFFER REG. ADDRESS
CONTAINS NULL CHARACTER FOR FILLS
CONTAINS # OF FILLER CHARACTERS REQUIRED
INSERT FILL CHARS. AFTER A "LINE FEED"
"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)

\$TIMES: 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED

; SBTTL APT MAILBOX-ETABLE

; EVEN

1285	001214		\$MAIL:		:: APT MAILBOX
1286	001214	000000	\$MSGTY:	.WORD	AMSGTY :: MESSAGE TYPE CODE
1287	001216	000000	\$FATAL:	.WORD	AFATAL :: FATAL ERROR NUMBER
1288	001220	000000	\$TESTN:	.WORD	ATESTN :: TEST NUMBER
1289	001222	000000	\$PASS:	.WORD	APASS :: PASS COUNT
1290	001224	000000	\$DEVCT:	.WORD	ADEVCT :: DEVICE COUNT
1291	001226	000000	\$UNIT:	.WORD	AUNIT :: I/O UNIT NUMBER
1292	001230	000000	\$MSGAD:	.WORD	AMSGAD :: MESSAGE ADDRESS
1293	001232	000000	\$MSGLG:	.WORD	AMSGLG :: MESSAGE LENGTH
1294	001234		\$ETABLE:		:: APT ENVIRONMENT TABLE
1295	001234	000	\$ENV:	.BYTE	AENV :: ENVIRONMENT BYTE
1296	001235	000	\$ENVM:	.BYTE	AENVM :: ENVIRONMENT MODE BITS
1297	001236	000000	\$SWREG:	.WORD	ASWREG :: APT SWITCH REGISTER
1298	001240	000000	\$USWR:	.WORD	AUSWR :: USER SWITCHES
1299	001242	000000	\$CPUOP:	.WORD	ACPUOP :: CPU TYPE, OPTIONS
1300			*		BITS 15-11=CPU TYPE
1301			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1302			*		11/70=06, P00=07, Q=10
1303			*		BIT 10=REAL TIME CLOCK
1304			*		BIT 9=FLOATING POINT PROCESSOR
1305			*		BIT 8=MEMORY MANAGEMENT
1306	001244	000	\$MAMS1:	.BYTE	AMAMS1 :: HIGH ADDRESS, M.S. BYTE
1307	001245	000	\$MTYP1:	.BYTE	AMTYP1 :: MEM. TYPE, BLK#1
1308			*		MEM. TYPE BYTE -- (HIGH BYTE)
1309			*		900 NSEC CORE=001
1310			*		300 NSEC BIPOLAR=002
1311			*		500 NSEC MOS=003
1312	001246	000000	\$MAOR1:	.WORD	AMAOR1 :: HIGH ADDRESS, BLK#1
1313			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1314	001250	000	\$MAMS2:	.BYTE	AMAMS2 :: HIGH ADDRESS, M.S. BYTE
1315	001251	000	\$MTYP2:	.BYTE	AMTYP2 :: MEM. TYPE, BLK#2
1316	001252	000000	\$MAOR2:	.WORD	AMAOR2 :: MEM. LAST ADDRESS, BLK#2
1317	001254	000	\$MAMS3:	.BYTE	AMAMS3 :: HIGH ADDRESS, M.S. BYTE
1318	001255	000	\$MTYP3:	.BYTE	AMTYP3 :: MEM. TYPE, BLK#3
1319	001256	000000	\$MAOR3:	.WORD	AMAOR3 :: MEM. LAST ADDRESS, BLK#3
1320	001260	000	\$MAMS4:	.BYTE	AMAMS4 :: HIGH ADDRESS, M.S. BYTE
1321	001261	000	\$MTYP4:	.BYTE	AMTYP4 :: MEM. TYPE, BLK#4
1322	001262	000000	\$MAOR4:	.WORD	AMAOR4 :: MEM. LAST ADDRESS, BLK#4
1323	001264	120210	\$VECT1:	.WORD	AVECT1 :: INTERRUPT VECTOR#1, BUS PRIORITY#1
1324	001266	000000	\$VECT2:	.WORD	AVECT2 :: INTERRUPT VECTOR#2, BUS PRIORITY#2
1325	001270	177440	\$BASE:	.WORD	ABASE :: BASE ADDRESS OF EQUIPMENT UNDER TEST
1326	001272	000000	\$DEVN:	.WORD	ADEVN :: DEVICE MAP
1327	001274	000000	\$CDW1:	.WORD	ACDW1 :: CONTROLLER DESCRIPTION WORD#1
1328	001276	000000	\$CDW2:	.WORD	ACDW2 :: CONTROLLER DESCRIPTION WORD#2
1329	001300		\$ETEND:		
1330			.MEXIT		

1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345 001300
1346
1347
1348 001300 057746
1349 001302 064125
1350 001304 053206
1351 001306 053650
1352
1353
1354 001310 057746
1355 001312 064170
1356 001314 053206
1357 001316 053650
1358
1359
1360 001320 057746
1361 001322 064214
1362 001324 053206
1363 001326 053650
1364
1365
1366 001330 060030
1367 001332 064125
1368 001334 053206
1369 001336 053650
1370
1371
1372 001340 060030
1373 001342 064170
1374 001344 053206
1375 001346 053650
1376
1377
1378 001350 060030
1379 001352 064214
1380 001354 053206
1381 001356 053650
1382
1383
1384 001360 060113
1385 001362 064125
1386 001364 053206

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF SITEMB 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: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
CSI INCORRECT.
EM200
EM3000
DT001
DF001
ERROR 2: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
MESSAGE A INCORRECT.
EM200
EM3001
DT001
DF001
ERROR 3: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
MESSAGE B INCORRECT.
EM200
EM3002
DT001
DF001
ERROR 4: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
CSI INCORRECT.
EM201
EM3000
DT001
DF001
ERROR 5: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
MESSAGE A INCORRECT.
EM201
EM3001
DT001
DF001
ERROR 6: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
MESSAGE IS INCORRECT.
EM201
EM3002
DT001
DF001
ERROR 7: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
CSI INCORRECT
EM202
EM3000
DT001

1387	001366	053650	DF001	
1388			ERROR 10:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1389				MESSAGE A INCORRECT.
1390	001370	060113	EM202	
1391	001372	064170	EM3001	
1392	001374	053206	DT001	
1393	001376	053650	DF001	
1394			ERROR 11:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1395				MESSAGE B INCORRECT.
1396	001400	060113	EM202	
1397	001402	064214	EM3002	
1398	001404	053206	DT001	
1399	001406	053650	DF001	
1400			ERROR 12:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1401				CSI INCORRECT.
1402	001410	060204	EM203	
1403	001412	064125	EM3000	
1404	001414	053206	DT001	
1405	001416	053650	DF001	
1406			ERROR 13:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1407				MESSAGE A INCORRECT.
1408	001420	060204	EM203	
1409	001422	064170	EM3001	
1410	001424	053206	DT001	
1411	001426	053650	DF001	
1412			ERROR 14:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1413				MESSAGE B INCORRECT.
1414	001430	060204	EM203	
1415	001432	064214	EM3002	
1416	001434	053206	DT001	
1417	001436	053650	DF001	
1418			ERROR 15:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1419				CSI INCORRECT AFTER SENDING DRIVE CLEAR.
1420	001440	060276	EM204	
1421	001442	064240	EM3003	
1422	001444	053226	DT015	
1423	001446	053674	DF015	
1424			ERROR 16:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1425				CSI INCORRECT AFTER DATE SIMULATION.
1426	001450	060276	EM204	
1427	001452	064310	EM3004	
1428	001454	053226	DT015	
1429	001456	053674	DF015	
1430			ERROR 17:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1431				MAINT REG. 1 INCORRECT DURING DATA SIMULATION (SECTOR PULSE)
1432	001460	060276	EM204	
1433	001462	064362	EM3005	
1434	001464	053236	DT017	
1435	001466	053720	DF017	
1436			ERROR 20:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1437				MAINT REG. 1 INCORRECT DURING DATA SIMULATION (INDEX PULSE)
1438	001470	060276	EM204	
1439	001472	064464	EM3006	
1440	001474	053236	DT017	
1441	001476	053720	DF017	
1442			ERROR 21:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT

1443			:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)
1444	001500	060276	:	EM204
1445	001502	064565	:	EM3007
1446	001504	053236	:	DT017
1447	001506	053720	:	DF017
1448			:	ERROR 22: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1449			:	CSI INCORRECT AFTER SENDING DRIVE CLEAR
1450	001510	060364	:	EM205
1451	001512	064240	:	EM3003
1452	001514	053254	:	DT022
1453	001516	053744	:	DF022
1454			:	ERROR 23: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1455			:	BUS ADD INCORRECT AFTER SENDING DRIVE CLEAR.
1456	001520	060364	:	EM205
1457	001522	064706	:	EM3008
1458	001524	053254	:	DT022
1459	001526	053744	:	DF022
1460			:	ERROR 24: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1461			:	WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR
1462	001530	060364	:	EM205
1463	001532	064766	:	EM3009
1464	001534	053254	:	DT022
1465	001536	053744	:	DF022
1466			:	ERROR 25: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1467			:	MAINT REG 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)
1468	001540	060364	:	EM205
1469	001542	065264	:	EM3014
1470	001544	053274	:	DT025
1471	001546	053770	:	DF025
1472			:	ERROR 26: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1473			:	CSI CHANGED DURING COMMAND EXECUTION
1474	001550	060364	:	EM205
1475	001552	065045	:	EM3010
1476	001554	053254	:	DT022
1477	001556	053744	:	DF022
1478			:	ERROR 27: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1479			:	BUS ADDRESS CHANGED BEFORE INDEX PULSE
1480	001560	060364	:	EM205
1481	001562	065112	:	EM3011
1482	001564	053254	:	DT022
1483	001566	053744	:	DF022
1484			:	ERROR 30: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1485			:	WORD COUNT CHANGED BEFORE INDEX PULSE
1486	001570	060364	:	EM205
1487	001572	065161	:	EM3012
1488	001574	053254	:	DT022
1489	001576	053744	:	DF022
1490			:	ERROR 31: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1491			:	CSI CHANGED AFTER INDEX PULSE
1492	001600	060364	:	EM205
1493	001602	065227	:	EM3013
1494	001604	053306	:	DT031
1495	001606	054014	:	DF031
1496			:	ERROR 32: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1497			:	BUS ADDRESS CHANGED AFTER INDEX PULSE.
1498	001610	060364	:	EM205

H03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 32
 DZR6CB.P11 28-JAN-77 14:04 ERROR POINTER TABLE

SEQ 0032

1499	001612	065335	EM3015
1500	001614	053306	DT031
1501	001616	054014	DF031
1502	:	:	ERROR 33: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1503	:	:	WORD COUNT CHANGED AFTER INDEX PULSE
1504	001620	060364	EM205
1505	001622	065403	EM3016
1506	001624	053306	DT031
1507	001626	054014	DF031
1508	:	:	ERROR 34: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1509	:	:	MAINT REG 1 INCORRECT AFTER SECTOR PULSE
1510	001630	060364	EM205
1511	001632	065763	EM3025
1512	001634	053306	DT031
1513	001636	054014	DF031
1514	:	:	ERROR 35: ATTEMPTING AN NPR READ OF ONE WORD
1515	:	:	CS1 INCORRECT
1516	001640	060452	EM206
1517	001642	064125	EM3000
1518	001644	053332	DT035
1519	001646	054040	DF035
1520	:	:	ERROR 36: ATTEMPTING AN NPR READ OF ONE WORD
1521	:	:	CS2 INCORRECT
1522	001650	060452	EM206
1523	001652	065450	EM3018
1524	001654	053332	DT035
1525	001656	054040	DF035
1526	:	:	ERROR 37: ATTEMPTING AN NPR READ OF ONE WORD
1527	:	:	BUS ADDRESS INCORRECT
1528	001660	060452	EM206
1529	001662	065514	EM3019
1530	001664	053332	DT035
1531	001666	054040	DF035
1532	:	:	ERROR 40: ATTEMPTING AN NPR READ OF ONE WORD
1533	:	:	WORD COUNT REG INCORRECT
1534	001670	060452	EM206
1535	001672	065542	EM3020
1536	001674	053332	DT035
1537	001676	054040	DF035
1538	:	:	ERROR 41: ATTEMPTING AN NPR READ OF ONE WORD
1539	:	:	WORD READ INCORRECT
1540	001700	060452	EM206
1541	001702	065573	EM3021
1542	001704	053356	DT041
1543	001706	054064	DF041
1544	:	:	ERROR 42: ATTEMPTING AN NPR READ OF ONE WORD
1545	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1546	001710	060452	EM206
1547	001712	065617	EM3022
1548	001714	053366	DT042
1549	001716	054110	DF042
1550	:	:	ERROR 43: ATTEMPTING AN NPR READ OF ONE WORD
1551	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1552	001720	060452	EM206
1553	001722	065667	EM3023
1554	001724	053366	DT042

1555	001726	054110	DF042	
1556			ERROR 44:	ATTEMPTING AN NPR READ OF ONE WORD
1557				CS1 INCORRECT
1558	001730	060515	EM207	
1559	001732	064125	EM3000	
1560	001734	053332	DT035	
1561	001736	054040	DF035	
1562			ERROR 45:	ATTEMPTING AN NPR READ
1563				CS2 INCORRECT
1564	001740	060515	EM207	
1565	001742	065450	EM3018	
1566	001744	053332	DT035	
1567	001746	054040	DF035	
1568			ERROR 46:	ATTEMPTING AN NPR READ
1569				BUS ADDRESS INCORRECT
1570	001750	060515	EM207	
1571	001752	065514	EM3019	
1572	001754	053332	DT035	
1573	001756	054040	DF035	
1574			ERROR 47:	ATTEMPTING AN NPR READ
1575				WORD COUNT INCORRECT
1576	001760	060515	EM207	
1577	001762	065542	EM3020	
1578	001764	053332	DT035	
1579	001766	054040	DF035	
1580			ERROR 50:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1581				CS1 INCORRECT
1582	001770	060544	EM208	
1583	001772	064125	EM3000	
1584	001774	053332	DT035	
1585	001776	054040	DF035	
1586			ERROR 51:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1587				CS2 INCORRECT
1588	002000	060544	EM208	
1589	002002	065450	EM3018	
1590	002004	053332	DT035	
1591	002006	054040	DF035	
1592			ERROR 52:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1593				BUS ADDRESS INCORRECT
1594	002010	060544	EM208	
1595	002012	065514	EM3019	
1596	002014	053332	DT035	
1597	002016	054040	DF035	
1598			ERROR 53:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1599				WORD COUNT INCORRECT
1600	002020	060544	EM208	
1601	002022	065542	EM3020	
1602	002024	053332	DT035	
1603	002026	054040	DF035	
1604			ERROR 54:	ATTEMPTING NPR READ
1605				DATA BUFFER INCORRECT
1606	002030	060515	EM207	
1607	002032	065573	EM3021	
1608	002034	053402	DT054	
1609	002036	054134	DF054	
1610			ERROR 55:	ATTEMPTING NPR READ

J03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 34
 DZR6CB.P11 28-JAN-77 14:04 ERROR POINTER TABLE

SEQ 0034

1611			:		CS1 INCORRECT AFTER READING DATA BUFFER
1612	002040	060515	:	EM207	
1613	002042	065617	:	EM3022	
1614	002044	053414	:	DT055	
1615	002046	054160	:	DF055	
1616			:	ERROR 56:	ATTEMPTING NPR READ
1617			:		CS2 INCORRECT AFTER READING DATA BUFFER
1618	002050	060515	:	EM207	
1619	002052	065667	:	EM3023	
1620	002054	053414	:	DT055	
1621	002056	054160	:	DF055	
1622			:	ERROR 57:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1623			:		INHIBIT CS1 INCORRECT
1624	002060	060615	:	EM209	
1625	002062	064125	:	EM3000	
1626	002064	053332	:	DT035	
1627	002066	054040	:	DF035	
1628			:	ERROR 60:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1629			:		INHIBIT CS2 INCORRECT
1630	002070	060615	:	EM209	
1631	002072	065450	:	EM3018	
1632	002074	053332	:	DT035	
1633	002076	054040	:	DF035	
1634			:	ERROR 61:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1635			:		INHIBIT BUS ADDRESS INCORRECT
1636	002100	060615	:	EM209	
1637	002102	065514	:	EM3019	
1638	002104	053332	:	DT035	
1639	002106	054040	:	DF035	
1640			:	ERROR 62:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1641			:		INHIBIT WORD COUNT INCORRECT
1642	002110	060615	:	EM209	
1643	002112	065542	:	EM3020	
1644	002114	053332	:	DT035	
1645	002116	054040	:	DF035	
1646			:	ERROR 63:	ATTEMPTING NPR READ WITH IBA TO CHECK ZERO
1647			:		DETECT-CS1 INCORRECT
1648	002120	060704	:	EM210	
1649	002122	064125	:	EM3000	
1650	002124	053332	:	DT035	
1651	002126	054040	:	DF035	
1652			:	ERROR 64:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1653			:		DETECT-CS2 INCORRECT
1654	002130	060704	:	EM210	
1655	002132	065450	:	EM3018	
1656	002134	053332	:	DT035	
1657	002136	054040	:	DF035	
1658			:	ERROR 65:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1659			:		DETECT-BUS ADDRESS INCORRECT
1660	002140	060704	:	EM210	
1661	002142	065514	:	EM3019	
1662	002144	053332	:	DT035	
1663	002146	054040	:	DF035	
1664			:	ERROR 66:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1665			:		DETECT-WORD COUNT INCORRECT
1666	002150	060704	:	EM210	

K03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 35
 DZR6CB.P11 28-JAN-77 14:04 ERROR POINTER TABLE

SEQ 0C35

1667	002152	065542	EM3020	
1668	002154	053332	DT035	
1669	002156	054040	DF035	
1670	:	:	ERROR 67:	ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1671	:	:		INCREMENT-DATA BUFFER INCORRECT
1672	002160	060615	EM209	
1673	002162	065573	EM3021	
1674	002164	053402	DT054	
1675	002166	054134	DF054	
1676	:	:	ERROR 70:	ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1677	:	:		INCREMENT-CS1 INCORRECT AFTER READING DATA BUFFER
1678	002170	060615	EM209	
1679	002172	065617	EM3022	
1680	002174	053414	DT055	
1681	002176	054160	DF055	
1682	:	:	ERROR 71:	ATTEMPTING NPR READ WITH ADDRESS BUFFER INHIBIT
1683	:	:		INCREMENT-CS2 INCORRECT AFTER READING DATA BUFFER
1684	002200	060615	EM209	
1685	002202	065667	EM3023	
1686	002204	053414	DT055	
1687	002206	054160	DF055	
1688	:	:	ERROR 72:	ATTEMPTING TO FORCE NON-EXISTANT MEMORY
1689	:	:		CS1 INCORRECT
1690	002210	061020	EM211	
1691	002212	064125	EM3000	
1692	002214	053432	DT072	
1693	002216	054204	DF072	
1694	:	:	ERROR 73:	ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1695	:	:		CS2 INCORRECT
1696	002220	061020	EM211	
1697	002222	065450	EM3018	
1698	002224	053432	DT072	
1699	002226	054204	DF072	
1700	:	:	ERROR 74:	ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1701	:	:		ERROR REG INCORRECT
1702	002230	061020	EM211	
1703	002232	065737	EM3024	
1704	002234	053432	DT072	
1705	002236	054204	DF072	
1706	:	:	ERROR 75:	ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1707	:	:		BUS ADDRESS INCORRECT
1708	002240	061020	EM211	
1709	002242	065514	EM3019	
1710	002244	053432	DT072	
1711	002246	054204	DF072	
1712	:	:	ERROR 76:	ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1713	:	:		WORD COUNT INCORRECT
1714	002250	061020	EM211	
1715	002252	065542	EM3020	
1716	002254	053432	DT072	
1717	002256	054204	DF072	
1718	:	:	ERROR 77:	ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1719	:	:		CS1 INCORRECT
1720	002260	061070	EM212	
1721	002262	064125	EM3000	
1722	002264	053462	DT077	

1723	002266	054240	DF077
1724			ERROR 100: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1725			CS2 INCORRECT
1726	002270	061070	EM212
1727	002272	065450	EM3018
1728	002274	053462	DT077
1729	002276	054240	DF077
1730			ERROR 101: TESTING EXTENDED MEMORY ADDRESSING BITS
1731			CS1 INCORRECT
1732	002300	061140	EM213
1733	002302	064125	EM3000
1734	002304	053332	DT035
1735	002306	054040	DF035
1736			ERROR 102: TESTING EXTENDED MEMORY ADDRESSING BITS
1737			CS2 INCORRECT
1738	002310	061140	EM213
1739	002312	065450	EM3018
1740	002314	053332	DT035
1741	002316	054040	DF035
1742			ERROR 103: TESTING EXTENDED MEMORY ADDRESSING BITS
1743			BUS ADDRESS INCORRECT
1744	002320	061140	EM213
1745	002322	065514	EM3019
1746	002324	053332	DT035
1747	002326	054040	DF035
1748			ERROR 104: TESTING EXTEND MEMORY ADDRESSING BITS
1749			WORD COUNT INCORRECT
1750	002330	061140	EM213
1751	002332	065542	EM3020
1752	002334	053332	DT035
1753	002336	054040	DF035
1754			ERROR 105: TESTING EXTENDED MEMORY ADDRESSING BITS
1755			DATA BUFFER INCORRECT
1756	002340	061140	EM213
1757	002342	065573	EM3021
1758	002344	053356	DT041
1759	002346	054064	DF041
1760			ERROR 106: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1761			CS1 INCORRECT
1762	002350	061210	EM214
1763	002352	064125	EM3000
1764	002354	053432	DT072
1765	002356	054204	DF072
1766			ERROR 107: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1767			CS2 INCORRECT
1768	002360	061210	EM214
1769	002362	065450	EM3018
1770	002364	053432	DT072
1771	002366	054204	DF072
1772			ERROR 110: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1773			BUS ADDRESS INCORRECT
1774	002370	061210	EM214
1775	002372	065514	EM3019
1776	002374	053432	DT072
1777	002376	054204	DF072
1778			ERROR 111: ATTEMPTING TO FORCE UNIBUS PARITY ERROR

1779			:	WORD COUNT INCORRECT
1780	002400	061210	:	EM214
1781	002402	065542	:	EM3020
1782	002404	053432	:	DT072
1783	002406	054204	:	DF072
1784			:	ERROR 112: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1785			:	CS1 INCORRECT
1786	002410	061260	:	EM215
1787	002412	064125	:	EM3000
1788	002414	053332	:	DT035
1789	002416	054040	:	DF035
1790			:	ERROR 113: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1791			:	CS2 INCORRECT
1792	002420	061260	:	EM215
1793	002422	065450	:	EM3018
1794	002424	053332	:	DT035
1795	002426	054040	:	DF035
1796			:	ERROR 114: ATTEMPTING NPR READ OR LOCATION PRIOR TO BAD PARITY
1797			:	BUS ADDRESS INCORRECT
1798	002430	061260	:	EM215
1799	002432	065514	:	EM3019
1800	002434	053332	:	DT035
1801	002436	054040	:	DF035
1802			:	ERROR 115: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1803			:	WORD COUNT INCORRECT
1804	002440	061260	:	EM215
1805	002442	065450	:	EM3018
1806	002444	053332	:	DT035
1807	002446	054040	:	DF035
1808			:	ERROR 116: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1809			:	ERROR REG INCORRECT
1810	002450	061210	:	EM214
1811	002452	065737	:	EM3024
1812	002454	053432	:	DT072
1813	002456	054204	:	DF072
1814			:	ERROR 117: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1815			:	CS1 INCORRECT
1816	002460	061344	:	EM216
1817	002462	064125	:	EM3000
1818	002464	053462	:	DT077
1819	002466	054240	:	DF077
1820			:	ERROR 120: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1821			:	CS2 INCORRECT
1822	002470	061344	:	EM216
1823	002472	065450	:	EM3018
1824	002474	053462	:	DT077
1825	002476	054240	:	DF077
1826			:	ERROR 121: ATTEMPTING 18 BIT NPR READ
1827			:	CS1 INCORRECT
1828	002500	061414	:	EM217
1829	002502	064125	:	EM3000
1830	002504	053332	:	DT035
1831	002506	054040	:	DF035
1832			:	ERROR 122: ATTEMPTING 18 BIT NPR READ
1833			:	CS2 INCORRECT
1834	002510	061414	:	EM217

1835	002512	065450	EM3018
1836	002514	053332	DT035
1837	002516	054040	DF035
1838	:	:	ERROR 123: ATTEMPTING 18 BIT NPR READ
1839	:	:	BUS ADDRESS INCORRECT
1840	002520	061414	EM217
1841	002522	065514	EM3019
1842	002524	053332	DT035
1843	002526	054040	DF035
1844	:	:	ERROR 124: ATTEMPTING 18 BIT NPR READ
1845	:	:	WORD COUNT INCORRECT
1846	002530	061414	EM217
1847	002532	065542	EM3020
1848	002534	053332	DT035
1849	002536	054040	DF035
1850	:	:	ERROR 125: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1851	:	:	CS1 INCORRECT
1852	002540	061447	EM218
1853	002542	064125	EM3000
1854	002544	053332	DT035
1855	002546	054040	DF035
1856	:	:	ERROR 126: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1857	:	:	CS2 INCORRECT
1858	002550	061447	EM218
1859	002552	065450	EM3018
1860	002554	053332	DT035
1861	002556	054040	DF035
1862	:	:	ERROR 127: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1863	:	:	BUS ADDRESS INCORRECT
1864	002560	061447	EM218
1865	002562	065514	EM3019
1866	002564	053332	DT035
1867	002566	054040	DF035
1868	:	:	ERROR 130: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1869	:	:	WORD COUNT INCORRECT
1870	002570	061447	EM218
1871	002572	065542	EM3020
1872	002574	053332	DT035
1873	002576	054040	DF035
1874	:	:	ERROR 131: ATTEMPTING 18 BIT NPR READ
1875	:	:	DATA BUFFER INCORRECT
1876	002600	061414	EM217
1877	002602	065573	EM3021
1878	002604	053366	DT042
1879	002606	054110	DF042
1880	:	:	ERROR 132: ATTEMPTING 18 BIT NPR READ
1881	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1882	002610	061414	EM217
1883	002612	065617	EM3022
1884	002614	053414	DT055
1885	002616	054160	DF055
1886	:	:	ERROR 133: ATTEMPTING 18 BIT NPR READ
1887	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1888	002620	061414	EM217
1889	002622	065667	EM3023
1890	002624	053414	DT055

1891	002626	054160	DF055
1892			ERROR 134: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1893			CSI INCORRECT
1894	002630	061527	EM219
1895	002632	064125	EM3000
1896	002634	053332	DT035
1897	002636	054040	DF035
1898			ERROR 135: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1899			CS2 INCORRECT
1900	002640	061527	EM219
1901	002642	065450	EM3018
1902	002644	053332	DT035
1903	002646	054040	DF035
1904			ERROR 136: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1905			BUS ADDRESS INCORRECT
1906	002650	061527	EM219
1907	002652	065514	EM3019
1908	002654	053332	DT035
1909	002656	054040	DF035
1910			ERROR 137: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1911			WORD COUNT INCORRECT
1912	002660	061527	EM219
1913	002662	065542	EM3020
1914	002664	053332	DT035
1915	002666	054040	DF035
1916			ERROR 140: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1917			CHECKING ZERO DETECT
1918			CSI INCORRECT
1919	002670	061607	EM220
1920	002672	064125	EM3000
1921	002674	053332	DT035
1922	002676	054040	DF035
1923			ERROR 141: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1924			CHECKING ZERO DETECT
1925			CS2 INCORRECT
1926	002700	061607	EM220
1927	002702	065450	EM3018
1928	002704	053332	DT035
1929	002706	054040	DF035
1930			ERROR 142: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1931			CHECKING ZERO DETECT
1932			BUS ADDRESS INCORRECT
1933	002710	061607	EM220
1934	002712	065514	EM3019
1935	002714	053332	DT035
1936	002716	054040	DF035
1937			ERROR 143: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1938			CHECKING ZERO DETECT
1939			WORD COUNT INCORRECT
1940	002720	061607	EM220
1941	002722	065542	EM3020
1942	002724	053332	DT035
1943	002726	054040	DF035
1944			ERROR 144: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1945			DATA BUFFER INCORRECT
1946	002730	061527	EM219

1947	002732	065573	EM3021
1948	002734	053356	DT041
1949	002736	054064	DF041
1950			ERROR 145: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1951			CS1 INCORRECT AFTER READING DATA BUFFER
1952	002740	061527	EM219
1953	002742	065617	EM3022
1954	002744	053462	DT077
1955	002746	054240	DF077
1956			ERROR 146: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1957			CS2 INCORRECT AFTER READING DATA BUFFER
1958	002750	061527	EM219
1959	002752	065667	EM3023
1960	002754	053462	DT077
1961	002756	054240	DF077
1962			ERROR 147: UNEXPECTED MEMORY PARITY ENABLE TRAP
1963	002760	057701	EM000
1964	002762	055061	DM000C
1965	002764	053202	DT000
1966	002766	053644	DF000
1967			ERROR 150: ATTEMPTING SIMULATION OF DATA IN READ HEADER
1968			CS1 INCORRECT
1969	002770	061666	EM221
1970	002772	064125	EM3000
1971	002774	053476	DT150
1972	002776	054264	DF150
1973			ERROR 151: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1974			CS1 INCORRECT AFTER COMPLETION OF COMMAND
1975	003000	061743	EM222
1976	003002	066033	EM3026
1977	003004	053366	DT042
1978	003006	054110	DF042
1979			ERROR 152: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1980			CS2 INCORRECT AFTER COMPLETION OF COMMAND
1981	003010	061743	EM222
1982	003012	066102	EM3027
1983	003014	053366	DT042
1984	003016	054110	DF042
1985			ERROR 153: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1986			CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1987	003020	062016	EM223
1988	003022	065617	EM3022
1989	003024	053414	DT055
1990	003026	054160	DF055
1991			ERROR 154: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1992			CS2 INCORRECT AFTER UNLOADING DATA
1993	003030	062016	EM223
1994	003032	065667	EM3023
1995	003034	053414	DT055
1996	003036	054160	DF055
1997			ERROR 155: ATTEMPTING DATA DUFFER READ AFTER READ HEADER
1998			DATA READ INCORRECT
1999	003040	062016	EM223
2000	003042	065573	EM3021
2001	003044	053402	DT054
2002	003046	054134	DF054

2003	:	ERROR 156: ATTEMPTING SIMULATION OF DATA IN READ HEADER (20 BIT FORMAT)
2004	:	CS1 INCORRECT
2005	003050	062074
2006	003052	064125
2007	003054	053476
2008	003056	054264
2009	:	ERROR 157: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE/
2010	:	CS1 INCORRECT AFTER COMMAND COMPLETION
2011	003060	062171
2012	003062	066033
2013	003064	053366
2014	003066	054110
2015	:	ERROR 160: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE
2016	:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
2017	003070	062171
2018	003072	066102
2019	003074	053366
2020	003076	054110
2021	:	ERROR 161: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2022	:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
2023	003100	062256
2024	003102	065617
2025	003104	053414
2026	003106	054160
2027	:	ERROR 162: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2028	:	CS2 INCORRECT AFTER UNLOADING DATA BUFFER
2029	003110	062256
2030	003112	065667
2031	003114	053414
2032	003116	054160
2033	:	ERROR 163: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2034	:	DATA BUFFER INCORRECT
2035	003120	062256
2036	003122	065573
2037	003124	053402
2038	003126	054134
2039	:	ERROR 164: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2040	:	CS1 INCORRECT
2041	003130	062356
2042	003132	064125
2043	003134	053512
2044	003136	054310
2045	:	ERROR 165: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2046	:	CS2 INCORRECT
2047	003140	062356
2048	003142	065450
2049	003144	053512
2050	003146	054310
2051	:	ERROR 166: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2052	:	CS1 INCORRECT READING EMPTY SILO
2053	003150	062356
2054	003152	066151
2055	003154	053512
2056	003156	054310
2057	:	ERROR 167: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2058	:	CS1 INCORRECT READING EMPTY SILO

E04

2059	003160	062356	EM227
2060	003162	066220	EM3029
2061	003164	053512	DT164
2062	003166	054310	DF164
2063			ERROR 170: WRITE BIT ERRORS
2064	003170	000000	0
2065	003172	000000	0
2066	003174	053530	DT170
2067	003176	054334	DF170
2068			ERROR 171: WRITE GATE NOT RESET WITH SECTOR PULSE
2069	003200	062572	EM231
2070	003202	066267	EM3030
2071	003204	053554	DT171
2072	003206	054360	DF171
2073			ERROR 172: WRITE GATE NOT SET WITH SECTOR PULSE RESET
2074	003210	062714	EM232
2075	003212	066267	EM3030
2076	003214	053554	DT171
2077	003216	054360	DF171
2078			ERROR 173: WRITE GATE NOT RESET WITH SECOND INDEX PULSE
2079	003220	063142	EM235
2080	003222	066267	EM3030
2081	003224	053554	DT171
2082	003226	054360	DF171
2083			ERROR 174: CSI INCORRECT AT END OF WRITE HEADER
2084	003230	063252	EM236
2085	003232	064125	EM3000
2086	003234	053226	DT015
2087	003236	053674	DF015
2088			ERROR 175: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2089			CSI INCORRECT
2090	003240	063622	EM241
2091	003242	064125	EM3000
2092	003244	053564	DT175
2093	003246	054404	DF175
2094			ERROR 176: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2095			CS2 INCORRECT
2096	003250	063622	EM241
2097	003252	065450	EM3018
2098	003254	053564	DT175
2099	003256	054404	DF175
2100			ERROR 177: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2101			DRIVE STATUS REG INCORRECT
2102	003260	063622	EM241
2103	003262	066315	EM3031
2104	003264	053564	DT175
2105	003266	054404	DF175
2106			ERROR 200: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2107			ERROR REG INCORRECT
2108	003270	063622	EM241
2109	003272	066350	EM3032
2110	003274	053564	DT175
2111	003276	054404	DF175
2112			ERROR 201: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2113			CSI INCORRECT
2114	003300	063706	EM242

2115	003302	064125	EM3000
2116	003304	053564	DT175
2117	003306	054404	DF175
2118	:	:	ERROR 202: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2119	:	:	CS2 INCORRECT
2120	003310	063706	EM242
2121	003312	065450	EM3018
2122	003314	053564	DT175
2123	003316	054404	DF175
2124	:	:	ERROR 203: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2125	:	:	DRIVE STATUS REG INCORRECT
2126	003320	063706	EM242
2127	003322	066315	EM3031
2128	003324	053564	DT175
2129	003326	054404	DF175
2130	:	:	ERROR 204: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2131	:	:	ERROR REGISTER INCORRECT
2132	003330	063706	EM242
2133	003332	066350	EM3032
2134	003334	053564	DT175
2135	003336	054404	DF175
2136	:	:	ERROR 205: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2137	:	:	CS1 INCORRECT
2138	003340	063772	EM243
2139	003342	064125	EM3000
2140	003344	053564	DT175
2141	003346	054404	DF175
2142	:	:	ERROR 206: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2143	:	:	CS2 INCORRECT
2144	003350	063772	EM243
2145	003352	065450	EM3018
2146	003354	053564	DT175
2147	003356	054404	DF175
2148	:	:	ERROR 207: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2149	:	:	DRIVE STATUS REG INCORRECT
2150	003360	063772	EM243
2151	003362	066315	EM3031
2152	003364	053564	DT175
2153	003366	054404	DF175
2154	:	:	ERROR 210: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2155	:	:	ERROR REG INCORRECT
2156	003370	063772	EM243
2157	003372	066350	EM3032
2158	003374	053564	DT175
2159	003376	054404	DF175
2160	:	:	ERROR 211: ATTEMPTING TO CLEAR CONTROLLER ERROR
2161	:	:	CS1 INCORRECT
2162	003400	064073	EM244
2163	003402	064125	EM3000
2164	003404	053610	DT211
2165	003406	054430	DF211
2166	:	:	ERROR 212: ATTEMPTING TO CLEAR CONTROLLER ERROR
2167	:	:	CS2 INCORRECT
2168	003410	064073	EM244
2169	003412	065450	EM3018
2170	003414	053610	DT211

G04

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 44
DZR6CB.P11 28-JAN-77 14:04 ERROR POINTER TABLE

SEQ 0044

2171	003416	054430	DF211
2172			ERROR 213: ATTEMPTING TO CLEAR CONTROLLER
2173			DRIVE STATUS REG INCORRECT
2174	003420	064073	EM244
2175	003422	066315	EM3031
2176	003424	053610	DT211
2177	003426	054430	DF211
2178			ERROR 214: ATTEMPTING TO CLEAR CONTROLLER
2179			ERROR REG INCORRECT
2180	003430	064073	EM244
2181	003432	066350	EM3032
2182	003434	053610	DT211
2183	003436	054430	DF211

Address	Hex	Dec	Register Name	Value	Description
2184			.SBTTL TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER		
2185			T.CS1:	.WORD 0	:CONTROL AND STATUS REGISTER 1
2186	003440	000000	T.WC:	.WORD 0	:WORD COUNT REGISTER
2187	003442	000000	T.BA:	.WORD 0	:BUS ADDRESS REGISTER
2188	003444	000000	T.DA:	.WORD 0	:DESIRED TRACK SECTOR REGISTER
2189	003446	000000	T.CS2:	.WORD 0	:CONTROL AND STATUS REGISTER 2
2190	003450	000000	T.DS:	.WORD 0	:DRIVE STATUS REGISTER
2191	003452	000000	T.ER:	.WORD 0	:ERROR REGISTER
2192	003454	000000	T.ASOF:	.WORD 0	:ATTENTION SUMMARY AND OFFSET REGISTER
2193	003456	000000	T.DCYL:	.WORD 0	:DESIRED CYLINDER REGISTER
2194	003460	000000	T.DB:	.WORD 0	:DATA BUFFER
2195	003462	000000	T.MR1:	.WORD 0	:MAINTENANCE REGISTER 1
2196	003464	000000	T.MR2:	.WORD 0	:MAINTENANCE REGISTER 2
2197	003466	000000	T.MR3:	.WORD 0	:MAINTENANCE REGISTER 3
2198	003470	000000	T.ECPS:	.WORD 0	:ECC POSITION INFORMATION
2199	003472	000000	T.ECPT:	.WORD 0	:ECC PATTERN INFORMATION
2200	003474	000000	T.SPAR:	.WORD 0	:SPARE REGISTER
2201	003476	000000			
2202			.SBTTL EXPECTED RK611 CONTROLLER REGISTERS		
2203			E.CS1:	.WORD 0	:CONTROL AND STATUS REGISTER 1
2204			E.WC:	.WORD 0	:WORD COUNT REGISTER
2205	003500	000000	E.BA:	.WORD 0	:BUS ADDRESS REGISTER
2206	003502	000000	E.DA:	.WORD 0	:DESIRED TRACK SECTOR REGISTER
2207	003504	000000	E.CS2:	.WORD 0	:CONTROL AND STATUS REGISTER 2
2208	003506	000000	E.DS:	.WORD 0	:DRIVE STATUS REGISTER
2209	003510	000000	E.ER:	.WORD 0	:ERROR REGISTER
2210	003512	000000	E.ASOF:	.WORD 0	:ATTENTION SUMMARY AND OFFSET REGISTER
2211	003514	000000	E.DCYL:	.WORD 0	:DESIRED CYLINDER REGISTER
2212	003516	000000	E.DB:	.WORD 0	:DATA BUFFER
2213	003520	000000	E.MR1:	.WORD 0	:MAINTENANCE REGISTER 1
2214	003522	000000	E.MR2:	.WORD 0	:MAINTENANCE REGISTER 2
2215	003524	000000	E.MR3:	.WORD 0	:MAINTENANCE REGISTER 3
2216	003526	000000	E.ECPS:	.WORD 0	:ECC POSITION INFORMATION
2217	003530	000000	E.ECPT:	.WORD 0	:ECC PATTERN INFORMATION
2218	003532	000000	E.SPAR:	.WORD 0	:SPARE REGISTER
2219	003534	000000			
2220	003536	000000			
2221			.SBTTL PREVIOUS RK611 CONTROLLER REGISTERS		
2222			P.CS1:	.WORD 0	:CONTROL AND STATUS REGISTER 1
2223			P.WC:	.WORD 0	:WORD COUNT REGISTER
2224	003540	000000	P.BA:	.WORD 0	:BUS ADDRESS REGISTER
2225	003542	000000	P.DA:	.WORD 0	:DESIRED TRACK SECTOR REGISTER
2226	003544	000000	P.CS2:	.WORD 0	:CONTROL AND STATUS REGISTER 2
2227	003546	000000	P.DS:	.WORD 0	:DRIVE STATUS REGISTER
2228	003550	000000	P.ER:	.WORD 0	:ERROR REGISTER
2229	003552	000000	P.ASOF:	.WORD 0	:ATTENTION SUMMARY AND OFFSET REGISTER
2230	003554	000000	P.DCYL:	.WORD 0	:DESIRED CYLINDER REGISTER
2231	003556	000000	P.DB:	.WORD 0	:DATA BUFFER
2232	003560	000000	P.MR1:	.WORD 0	:MAINTENANCE REGISTER 1
2233	003562	000000	P.MR2:	.WORD 0	:MAINTENANCE REGISTER 2
2234	003564	000000	P.MR3:	.WORD 0	:MAINTENANCE REGISTER 3
2235	003566	000000	P.ECPS:	.WORD 0	:ECC POSITION INFORMATION
2236	003570	000000	P.ECPT:	.WORD 0	:ECC PATTERN INFORMATION
2237	003572	000000	P.SPAR:	.WORD 0	:SPARE REGISTER
2238	003574	000000			
2239	003576	000000			

```

2240          .SBTTL PROGRAM DEFINED VARIABLES
2241
2242 003600 000210      RKVEC:  .WORD  210      ;RK611 VECTOR
2243 003602 000240      RKPRI:  .WORD  PR5      ;RK611 PRIORITY
2244 003604 000000      TRAPPC: .WORD  0        ;PC FOR MEMORY CHECK ENABLE TRAP
2245 003606 000000      SRTFLG: .WORD  0        ;START FLAG
2246                                     ; 0 = 200
2247                                     ; 1 = 214
2248                                     ; -1 = 204
2249 003610 000000      ERRCNT: .WORD  0        ;ERROR COUNT FOR SWITCH 12 ABORT
2250 003612 000000      P1.BIT: .WORD  0        ;NEXT BIT IN DATA SIMULATION
2251 003614 000000      PR.BIT: .WORD  0        ;PRESENT BIT IN DATA SIMULATION
2252 003616 000000      M1.BIT: .WORD  0        ;PREVIOUS BIT IN DATA SIMULATION
2253 003620 000000      M2.BIT: .WORD  0        ;BIT BEFORE PREVIOUS BIT
2254 003622 000000      BITCNT: .WORD  0        ;BIT POSITION
2255 003624 000000      WRDCNT: .WORD  0        ;WORD COUNT FOR NPR TRANSFER
2256 003626 000000      SECCNT: .WORD  0        ;SECTOR COUNT
2257 003630 000000      MEMPAR: .WORD  0        ;MEMORY ENABLE ON FIRST 24K
2258 003632 000015      WAITIM: .WORD  15      ;WAIT TIME FOR CONTROLLER READY
2259 003634 000000      SAVSWR: .WORD  0        ;STORAGE FOR SWITCH REGISTER
2260 003636 000000      PARPRE: .WORD  0        ;PARITY OPTION PRESENT FOR
2261                                     ; LOCATION USED
2262 003640 000000      CSRPTR: .WORD  0        ;POINTER TO CSR TO BE USED
    
```

J04

```

2263 .SBTTL PROGRAM SETUP
2264
2265 003642 012737 000001 003606 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2266 003650 000406 BR START1
2267
2268 003652 012737 177777 003606 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2269 003660 000402 BR START1
2270
2271 003662 005037 003606 START: CLR SRTFLG ;CLEAR START FLAG
2272 003666 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2273 003670 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2274 003674 004737 051274 JSR PC,STKINT ;INIT KEYBOARD
2275 003700 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2276 003704 012746 003712 MOV #15,-(SP) ;LOAD START OF PROGRAM
2277 003710 000002 RTI ;LOAD PSW
2278
2279 003712
2280
2281 15:
2282 .SBTTL INITIALIZE THE COMMON TAGS
2283 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2284 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2285 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2286 CMP #SWR,R6 ;;DONE?
2287 BNE -6 ;;LOOP BACK IF NO
2288 MOV #STACK,SP ;;SETUP THE STACK POINTER
2289 ;;INITIALIZE A FEW VECTORS
2290 MOV #SCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2291 MOV #340,#IOTVEC+2 ;;LEVEL 7
2292 MOV #ERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2293 MOV #340,#EMTVEC+2 ;;LEVEL 7
2294 MOV #TRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2295 MOV #340,#TRAPVEC+2 ;;LEVEL 7
2296 MOV #SPWRON,#PWAVEC ;;POWER FAILURE VECTOR
2297 MOV #340,#PWAVEC+2 ;;LEVEL 7
2298 MOV $ENDCT,$SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
2299 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
2300 CLR $ESCAPE ;;CLEAR THE ESCAPE OR ERROR ADDRESS
2301 MOV #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
2302 MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2303 MOV #,$SLPERR ;;SETUP THE ERROR LOOP ADDRESS
2304 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2305 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2306 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
2307 MOV #64,$ERRVEC ;;SET UP ERROR VECTOR
2308 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
2309 MOV #DISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2310 CMP #-1,$SWR ;;TRY TO REFERENCE HARDWARE SWR
2311 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2312 BR 65$ ;;AND THE HARDWARE SWR IS NOT = -1
2313 BR 65$ ;;BRANCH IF NO TIMEOUT
2314 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
2315 RTI
2316 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
2317 MOV #DISPREG,DISPLAY ;;AND A SOFTWARE DISPLAY REGISTER
2318 66$: MOV (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
2319 CLR $PASS ;;CLEAR PASS COUNT

```

```

2319 004144 132737 000200 001235 BITB #APTSIZE,SENVN ;;TEST USER SIZE UNDER APT
2320 004152 001403 BEQ 67$ ;;YES,USE NON-APT SWITCH
2321 004154 012737 001236 001140 MOV #SSWREG,SWR ;;NO,USE APT SWITCH REGISTER
2322 004162 67$: CLR ERRCNT ;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
2323 004162 005037 003610
2324 .SBTTL TYPE PROGRAM NAME
2325 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2326 004166 005227 177777 INC #-1 ;;FIRST TIME?
2327 004172 001063 BNE 68$ ;;BRANCH IF NO
2328 004174 022737 044666 000042 CMP #SENDAD,2#42 ;;ACT-11?
2329 004202 001457 BEQ 68$ ;;BRANCH IF YES
2330 004204 104401 004252 TYPE 69$ ;;TYPE ASCIZ STRING
2331 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2332 004210 005737 000042 TST 2#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
2333 004214 001012 BNE 70$ ;;BRANCH IF YES
2334 004216 123727 001234 000001 CMPB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
2335 004224 001406 BEQ 70$ ;;BRANCH IF YES
2336 004226 023727 001140 000176 CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
2337 004234 001005 BNE 71$ ;;BRANCH IF NO
2338 004236 104406 GTSWR ;;GET SOFT-SWR SETTINGS
2339 004240 000403 BR 71$
2340 004242 112737 000001 001134 70$: MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
2341 004250 71$: BR 68$
2342 004250 000434 ;;GET OVER THE ASCIZ
2343 ;;69$: .ASCIZ <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 3 MAINDEC-11-DZR6C-B/<CRLF>
2344 68$:
2345 004342 005227 177777 INC #-1 ;TEST IF FIRST PASS
2346 004346 001002 BNE 6$ ;NO - SKIP
2347 004350 104401 054576 TYPE ,OPR006 ;TYPE RUN TIME MESSAGE
2348 004354 022737 000001 003606 6$: CMP #1,SRTFLG ;CHECK IF PARAMETER START
2349 004362 001122 BNE 15$ ;NO,CONTINUE SETUP
2350 004364 104401 054464 5$: TYPE ,OPR001 ;TYPE "RK611 BUS ADDRESS ( ) ="
2351 004370 013746 001270 MOV $BASE,-(SP) ;SAVE $BASE FOR TYPEOUT
2352 004374 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2353 004376 104401 054516 TYPE ,OPR002
2354 004402 104412 RDOCT ;GET VALUE
2355 004404 012637 001160 MOV (SP)+,$TMPD
2356 004410 001407 BEQ 7$ ;CHECK IF <CR>
2357 004412 022737 160000 001160 CMP #160000,$TMPD ;CHECK IF IN I/O PAGE
2358 004420 101361 BHI 5$
2359 004422 013737 001160 001270 MOV $TMPD,$BASE ;LOAD NEW BUS ADDRESS
2360 004430 104401 054524 7$: TYPE ,OPR003 ;TYPE "RK611 VECTOR ADDRESS ( ) ="
2361 004434 013746 001264 MOV $VECT1,-(SP) ;TYPE OUT VECTOR ADDRESS
2362 004440 042716 160000 BIC #160000,(SP)
2363 004444 104402 TYPOC
2364 004446 104401 054516 TYPE ,OPR002
2365 004452 104412 RDOCT ;GET VALUE
2366 004454 012637 001160 MOV (SP)+,$TMPD
2367 004460 001412 BEQ 10$ ;CHECK IF <CR>
2368 004462 022737 001000 001160 CMP #1000,$TMPD ;CHECK IF LEGAL
2369 004470 101757 BLOS 7$
2370 004472 042737 017777 001264 BIC #17777,$VECT1 ;LOAD NEW VECTOR ADDRESS
2371 004500 053737 001160 001264 BIS $TMPD,$VECT1
2372 004506 104401 054554 10$: TYPE ,OPR004 ;TYPE "RK611 PRIORITY ( ) ="
2373 004512 005046 CLR -(SP)
2374 004514 113716 001265 MOVB $VECT1+1,(SP)
    
```

L04

2375	004520	006216			ASR	(SP)		;SHIFT 5 BITS RIGHT
2376	004522	006216			ASR	(SP)		
2377	004524	006216			ASR	(SP)		
2378	004526	006216			ASR	(SP)		
2379	004530	006216			ASR	(SP)		
2380	004532	104402			TYPOC			
2381	004534	104401	054516		TYPE	,OPR002		
2382	004540	104412			RDOCT			;GET VALUE
2383	004542	012637	001160		MOV	(SP)+,STMP0		
2384	004546	001430			BEQ	15\$;CHECK FOR DEFAULT
2385	004550	022737	000007	001160	CMP	#7,STMP0		;CHECK IF LEGAL
2386	004556	103753			BLO	10\$		
2387	004550	022737	000004	001160	CMP	#4,STMP0		
2388	004556	101347			BHI	10\$		
2389	004570	006337	001160		ASL	STMP0		;SHIFT 5 BITS LEFT
2390	004574	006337	001160		ASL	STMP0		
2391	004600	006337	001160		ASL	STMP0		
2392	004604	006337	001160		ASL	STMP0		
2393	004610	006337	001160		ASL	STMP0		
2394	004614	042737	160000	001264	BIC	#160000,\$VECT1		;STORE NEW PRIORITY
2395	004622	153737	001160	001265	BISB	STMP0,\$VECT1+1		
2396	004630	013737	001264	003600	MOV	\$VECT1,RKVEC		;STORE RK611 VECTOR
2397	004636	042737	160000	003600	BIC	#160000,RKVEC		
2398	004644	113737	001265	003602	MOV#B	\$VECT1+1,RKPRI		;STORE PRIORITY
2399	004652	004737	046330		JSR	PC,\$SIZE		;SIZE MEMORY
2400	004656	013702	001270		MOV	\$BASE,R2		;SET RK611 BASE
2401	004662	005037	001202		CLR	\$ESCAPE		;CLEAR ESCAPE
2402								
2403	004666	004737	044732		NEWPAS: JSR	PC,PARCHK		;CHECK OF MEMORY CHECK ENABLE
2404	004672	012746	000000		MOV	#PRO,-(SP)		;ALLOW ALL INTERRUPTS
2405	004676	012746	004704		MOV	#TST1,-(SP)		
2406	004702	000002			RTI			

.SBTTL **DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

*TEST 1 READ HEADER SEEK MESSAGE

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT THE RK611 CONTROLLER IN DIAGNOSTIC MODE. ISSUE
* A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
* HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER.
* VERIFY THAT A SEEK IS LOADED WITH THE PROPER BITS IN
* MESSAGE SET. REPEAT FOR A READ HEADER WITH CDT SET
* IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.

```
TST1: SCOPE
MOV #100.,STIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;;CLEAR RK611
MOV #DMD,RKMR1(R2) ;;PUT RK611 IN DIAGNOSTIC MODE
MOV #1777,RKDCYL(R2) ;;LOAD CYLINDER ADDRESS REG.
MOV #3400,RKDA(R2) ;;LOAD TRACK
MOV #7,RKCS2(R2) ;;LOAD DRIVE NUM.
MOV #CDT!CFMT!RDHEAD,RKCS1(R2) ;;ISSUE RDHEAD WITH CDT SET IN
;; 24 SECTOR FORMAT
MOV #3*4+2,R0 ;;CLOCK IN DRIVE MESSAGE
15: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 15
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!RDHEAD,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 25 ;;YES, CHECK MESSAGE A
ERPOR 1 ;;CS1 INCORRECT
MOV #CCLR,RKCS1(R2) ;;CLEAR RK611
BR TST2 ;;GO TO NEXT TEST
25: CMP E.MR2,T.MR2 ;;CHECK MESS A CORRECT
BEQ 35 ;;YES, CHECK MESSAGE B
ERROR 2 ;;MESS A INCORRECT
35: CMP E.MR3,T.MR3 ;;CHECK MESS B CORRECT
BEQ TST2 ;;YES, GO ON TO NEXT TEST
ERROR ;;MESS B INCORRECT
```

*TEST 2 WRITE HEADER SEEK MESSAGE

* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY
* THAT A SEEK IS LOADED WITH THE RTC BIT SET. REPEAT
* FOR A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT,

2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562

* CYLINDER 1777, HEAD 7, DRIVE 7.

2463
2464
2465
2466 005122 000004
2467 005124 012737 000144 001200
2468 005132 013702 001270
2469 005136 012762 100000 000000
2470 005144 012762 000040 000026
2471 005152 012762 001777 000020
2472 005160 012762 003400 000006
2473 005166 012762 000007 000010
2474 005174 012762 012027 000000
2475
2476 005202 012700 000016
2477 005206 012762 000440 000026
2478 005214 012762 000040 000026
2479 005222 005300
2480 005224 001370
2481 005226 016237 000000 003440
2482 005234 016237 000034 003466
2483 005242 016237 000036 003470
2484 005250 012737 012027 003500
2485 005256 012737 071227 003526
2486 005264 012737 037760 003530
2487 005272 023737 003500 003440
2488 005300 001405
2489 005302 104004
2490 005304 012762 100000 000000
2491 005312 000412
2492 005314 023737 003526 003466
2493 005322 001401
2494 005324 104005
2495 005326 023737 003530 003470
2496 005334 001401
2497 005336 104006

TST2: SCOPE
MOV #100, \$TIMES ; DO 100. ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #1777, RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ; LOAD TRACK
MOV #7, RKCS2(R2) ; LOAD DRIVE NUM.
MOV #CDT!CFMT!WRHEAD, RKCS1(R2) ; ISSUE WRHEAD WITH CDT SET IN
; 24 SECTOR FORMAT
; CLOCK IN DRIVE MESSAGE
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!WRHEAD, E.CS1 ; LOAD EXPECTED CS1
MOV #S.SEEK!S.RTC!S.FMT!70007, E.MR2 ; LOAD EXPECTED MR2
MOV #37760, E.MR3 ; LOAD EXPECTED MR3
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2\$; YES, CHECK MESSAGE A
ERROR 4 ; 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 5 ; MESS A INCORRECT
CMP E.MR3, T.MR3 ; CHECK MESS B CORRECT
BEQ TST3 ; YES, GO ON TO NEXT TEST
ERROR 6 ; MESS B INCORRECT

TST3 READ HEADER DRIVE CLEAR MESSAGE

* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER WITH CDT SET
* IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
* CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS
* GENERATED, AND THE PROPER BITS ARE SET.

2509 005340 000004
2510 005342 012737 000144 001200
2511 005350 013702 001270
2512 005354 012762 100000 000000
2513 005352 012762 000040 000026
2514 005370 012762 001777 000020
2515 005376 012762 003400 000006
2516 005404 012762 000007 000010
2517 005412 012762 012025 000000
2518

TST3: SCOPE
MOV #100, \$TIMES ; DO 100. ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #1777, RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ; LOAD TRACK
MOV #7, RKCS2(R2) ; LOAD DRIVE NUMBER
MOV #CDT!CFMT!RDHEAD, RKCS1(R2) ; ISSUE COMMAND WITH CDT SET IN
; 24 SECTOR FORMAT

2519	005420	012700	000156			MOV	#27, #4+2, R0	:LOAD COUNT TO LOAD DRIVE CLEAR
2520	005424	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)	
2521	005432	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
2522	005440	005300				DEC	R0	
2523	005442	001370				BNE	1\$	
2524	005444	016237	000000	003440		MOV	RKCS1(R2), T.CS1	:STORE COMMAND AND STATUS REG. 1
2525	005452	016237	000034	003466		MOV	RKMR2(R2), T.MR2	:STORE MAINT. REG. 2 (MESS A)
2526	005460	016237	000036	003470		MOV	RKMR3(R2), T.MR3	:STORE MAINT. REG. 3 (MESS B)
2527	005466	012737	012025	003500		MOV	#CDT!CFMT!RDHEAD, E.CS1	:LOAD EXPECTED CS1
2528	005474	012737	071407	003526		MOV	#S.CLR!S.FMT!70007, E.MR2	:LOAD EXPECTED MAINT REG. 2
2529	005502	005037	003530			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2530	005506	023737	003500	003440		CMP	E.CS1, T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
2531	005514	001405				BEQ	2\$:YES, CHECK CS2
2532	005516	104007				ERROR	7	:CS1 INCORRECT
2533	005520	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:CLEAR RK611
2534	005526	000412				BR	TST4	:GO TO NEXT TEST
2535	005530	023737	003526	003466	2\$:	CMP	E.MR2, T.MR2	:CHECK MESS A CORRECT
2536	005536	001401				BEQ	3\$:YES, CHECK MESS B
2537	005540	104010				ERROR	10	:MESS A INCORRECT
2538	005542	023737	003530	003470	3\$:	CMP	E.MR3, T.MR3	:CHECK MESS B CORRECT
2539	005550	001401				BEQ	TST4	:YES, GO ON TO NEXT TEST
2540	005552	104011				ERROR	11	:MESS B INCORRECT

 *TEST 4 WRITE HEADER DRIVE CLEAR MESSAGE

* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER WITH CDT SET
 * IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
 * CLOCK SEEK MESSAGE AND LOAD GENERATED DRIVE CLEAR
 * INTO SHIFT REGISTER. MAKE SURE THE DRIVE CLEAR IS
 * GENERATED AND THE PROPER BITS ARE SET.

2553	005554	000004			TST4:	SCOPE		
2554	005556	012737	000144	001200		MOV	#100, STIMES	:DO 100. ITERATIONS
2555	005564	013702	001270			MOV	\$BASE, R2	:LOAD RK611 BASE
2556	005570	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:CLEAR RK611
2557	005576	012762	000040	000026		MOV	#DMD, RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
2558	005604	012762	001777	000020		MOV	#1777, RKDCYL(R2)	:LOAD CYLINDER ADDRESS REG.
2559	005612	012762	003400	000006		MOV	#3400, RKDA(R2)	:LOAD TRACK
2560	005620	012762	000007	000010		MOV	#7, RKCS2(R2)	:LOAD DRIVE NUMBER
2561	005626	012762	012027	000000		MOV	#CDT!CFMT!WRHEAD, RKCS1(R2)	:ISSUE COMMAND WITH CDT SET IN : 24 SECTOR FORMAT
2562								
2563	005634	012700	000156			MOV	#27, #4+2, R0	:LOAD COUNT TO LOAD DRIVE CLEAR
2564	005640	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)	
2565	005646	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
2566	005654	005300				DEC	R0	
2567	005656	001370				BNE	1\$	
2568	005660	016237	000000	003440		MOV	RKCS1(R2), T.CS1	:STORE COMMAND AND STATUS REG. 1
2569	005666	016237	000034	003466		MOV	RKMR2(R2), T.MR2	:STORE MAINT. REG. 2 (MESS A)
2570	005674	016237	000036	003470		MOV	RKMR3(R2), T.MR3	:STORE MAINT. REG. 3 (MESS B)
2571	005702	012737	012027	003500		MOV	#CDT!CFMT!WRHEAD, E.CS1	:LOAD EXPECTED CS1
2572	005710	012737	071407	003526		MOV	#S.CLR!S.FMT!70007, E.MR2	:LOAD EXPECTED MAINT REG. 2
2573	005716	005037	003530			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2574	005722	023737	003500	003440		CMP	E.CS1, T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT

2575	005730	001405				BEQ	2\$:YES, CHECK CS2
2576	005732	104012				ERROR	12	:CSI INCORRECT
2577	005734	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2578	005742	000412				BR	TST5	:GO TO NEXT TEST
2579	005744	023737	003526	003466	2\$:	CMP	E.MR2,T.MR2	:CHECK MESS A CORRECT
2580	005752	001401				BEQ	3\$:YES, CHECK MESS B
2581	005754	104013				ERROR	13	:MESS A INCORRECT
2582	005756	023737	003530	003470	3\$:	CMP	E.MR3,T.MR3	:CHECK MESS B CORRECT
2583	005764	001401				BEQ	TST5	:YES, GO ON TO NEXT TEST
2584	005766	104014				ERROR	14	:MESS B INCORRECT

.SBTTL **INDEX AND SECTOR PULSE DETECT ON

```

*****
:TEST 5          SECTOR PULSE DETECT IN READ HEADER (PART 1)

```

```

:
:   CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
:   IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN RK06
:   IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
:   CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
:   SIMULATE SECTOR PULSE, 255 ZEROES AND A ONE.

```

: MAKE SURE READ GATE DOES SET.

```

*****
TST5:  SCOPE

```

2601	005770	000004				MOV	#10,STIMES	:DO 10. ITERATIONS
2602	005772	012737	000012	001200		MOV	\$BASE,R2	:LOAD RK611 BASE
2603	006000	013702	001270			MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2604	006004	012762	100000	000000		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
2605	006012	012762	000040	000026		MOV	#DMD!MSP,RKMR1(R2)	:INITIALIZE ROM ADDRESS
2606	006020	012762	000140	000026		MOV	#DMD,RKMR1(R2)	
2607	006026	012762	000040	000026		MOV	#RDHEAD,RKCS1(R2)	:ISSUE READ HEADER
2608	006034	012762	000025	000000		MOV	#50,#4+2,R0	:CLOCK UNTIL READY FOR SECTOR PULSE
2609	006042	012700	000312			MOV	#DMD!MCLK,RKMR1(R2)	
2610	006046	012762	000440	000026	1\$:	MOV	#DMD,RKMR1(R2)	
2611	006054	012762	000040	000026		DEC	R0	
2612	006062	005300				BNE	1\$	
2613	006064	001370				MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2614	006066	016237	000000	003440		MOV	#RDHEAD,E.CS1	:LOAD EXPECTED CS1
2615	006074	012737	000025	003500		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
2616	006102	023737	003500	003440		BEQ	2\$:YES, CLOCK ZEROES
2617	006110	001405				ERROR	15	:CSI INCORRECT
2618	006112	104015				MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2619	006114	012762	100000	000000		BR	TST6	:GO ON TO NEXT TEST
2620	006122	000553						
2621								
2622	006124	012762	000140	000026	2\$:	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
2623	006132	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2624	006140	012737	022040	003524		MOV	#DMD!MEWD!ECCW,E.MR1	:LOAD EXPECT MAINT REG. 1
2625	006146	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
2626	006152	005037	003614			CLR	PR.BIT	:INITIALIZE PRESENT AND PREVIOUS
2627	006156	005037	003616			CLR	M1.BIT	:BITS TO GENERATE ZEROES
2628	006162	012700	000200			MOV	#128,R0	:GENERATE 128 ZEROS UNTIL READ GATE
2629	006166	004737	046156		5\$:	JSR	PC,R0BIT	:READ A ZERO
2630	006172	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1

```

2631 006200 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK MAINTENANCE REG. 1 CORRECT
2632 006206 001405                    BEQ      65             ;YES, SIMULATE NEXT BIT
2633 006210 104017                    ERROR   17             ;MAINT REG. 1 INCORRECT
2634 006212 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2635 006220 000514                    BR       T56           ;GO ON TO NEXT TEST
2636
2637 006222 005237 003622          65:     INC      BITCNT      ;INCREMENT BIT COUNT
2638 006226 005300                    DEC      R0            ;CHECK READY OF READ GATE
2639 006230 001356                    B'VE    55            ;NO, CONTINUE
2640 006232 012737 122040 003524      MOV      #DMD!MEWD!ECCW!R0GATE,E.MR1 ;LOAD EXPECTED MR1
2641 006240 012700 000177                    MOV      #127,R0       ;GENERATE 127 ZEROS
2642 006244 004737 046156          105:    JSR      PC,R0BIT      ;READ A ZERO
2643 006250 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
2644 006256 023737 003524 003464      CMP      E.MR1,T.MR1   ;CHECK MAINT REG. 1 CORRECT
2645 006264 001405                    BEQ      115          ;YES, SIMULATE NEXT BIT
2646 006266 104017                    ERROR   17             ;MAINT REG. 1 INCORRECT
2647 006270 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2648 006276 000465                    BR       T56           ;GO ON TO NEXT TEST
2649
2650 006300 005237 003622          115:    INC      BITCNT      ;INCREMENT BIT COUNT
2651 006304 005300                    DEC      R0            ;CHECK IF ALL ZEROS ISSUED
2652 006306 001356                    BNE     105           ;NO, CONTINUE
2653 006310 012737 000001 003614      MOV      #1,PR.BIT    ;LOAD ONE FOR READING 1
2654 006316 004737 046156          JSR      PC,R0BIT      ;READ A ONE
2655 006322 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.
2656 006330 023737 003524 003464      CMP      E.MR1,T.MR1   ;CHECK MAINT REG. 1
2657 006336 001405                    BEQ      125          ;YES, CONTINUE
2658 006340 104017                    ERROR   17             ;MAINTENANCE REG. 1 INCORRECT
2659 006342 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2660 006350 000440                    BR       T56           ;GO ON TO NEXT TEST
2661 006352 005237 003622          125:    INC      BITCNT      ;INCREMENT BIT COUNT
2662 006356 013737 003614 003616      MOV      PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2663 006364 005037 003614                    CLR      PR.BIT
2664 006370 004737 046156          JSR      PC,R0BIT      ;SIMULATE ZERO
2665 006374 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2666 006402 023737 003524 003464      CMP      E.MR1,T.MR1   ;CHECK MAINT REG. 1 CORRECT
2667 006410 001405                    BEQ      135          ;CHECK CSI CORRECT
2668 006412 104017                    ERROR   17             ;MAINTENANCE REG. 1 INCORRECT
2669 006414 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2670 006422 000413                    BR       T56           ;GO TO NEXT TEST
2671
2672 006424 016237 000000 003440      135:    MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2673 006432 012737 000025 003500      MOV      #RDHEAD,E.CS1 ;LOAD EXPECTED CSI
2674 006440 023737 003500 003440      CMP      E.CS1,T.CS1   ;CHECK CSI CORRECT
2675 006446 001401                    BEQ      T56          ;YES, GO TO NEXT TEST
2676 006450 104016                    ERROR   16             ;CSI INCORRECT

```

```

2677
2678 *****
2679 *TEST 6 SECTOR PULSE DETECT IN READ HEADER (PART 2)
2680 *
2681 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2682 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
2683 * IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
2684 * CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
2685 * SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.
2686 *

```

E05

```

2687          : * MAKE SURE READ GATE DOES NOT SET.
2688          : *
2689          : *****
2690 006452 000004          TST6: SCOPE
2691 006454 012737 000012 001200      MOV      #10, STIMES          ;; DO 10. ITERATIONS
2692 006462 013702 001270          MOV      $BASE, R2          ;; LOAD RK611 BASE
2693 006466 012762 100000 000000      MOV      #CCLR, RKCS1(R2)  ;; CLEAR RK611
2694 006474 012762 000040 000000      MOV      #DMD, RKCS1(R2)  ;; PUT RK611 IN DIAGNOSTIC MODE
2695 006502 012762 000140 000026      MOV      #DMD!MSP, RKMRI(R2) ;; INITIALIZE ROM ADDRESS
2696 006510 012762 000040 000026      MOV      #DMD, RKMRI(R2)
2697 006516 012762 000025 000000      MOV      #RDHEAD, RKCS1(R2) ;; ISSUE READ HEADER
2698 006524 012700 000312          MOV      #SO, #4+2, RO     ;; CLOCK UNTIL READY FOR SECTOR PULSE
2699 006530 012762 000440 000026 1$:   MOV      #DMD!MCLK, RKMRI(R2)
2700 006536 012762 000040 000026      MOV      #DMD, RKMRI(R2)
2701 006544 005300          DEC      RO
2702 006546 001370          BNE     1$
2703 006550 016237 000000 003440      MOV      RKCS1(R2), T.CS1  ;; STORE COMMAND AND STATUS REG. 1
2704 006556 012737 000025 003500      MOV      #RDHEAD, E.CS1   ;; LOAD EXPECTED CS1
2705 006564 023737 003500 003440      CMP      E.CS1, T.CS1     ;; CHECK COMMAND AND STATUS REG. 1 CORRECT
2706 006572 001405          BEQ     2$
2707 006574 104015          ERROR  1$
2708 006576 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ;; CLEAR RK611
2709 006604 000535          BR     TST7              ;; GO ON TO NEXT TEST
2710 006606          2$:
2711 006606 012762 000240 000026      MOV      #DMD!MIND, RKMRI(R2) ;; SIMULATE INDX PULSE
2712 006614 012700 000004          MOV      #4, RO
2713 006620 012762 000640 000026 3$:   MOV      #DMD!MIND!MCLK, RKMRI(R2)
2714 006626 012762 000240 000026      MOV      #DMD!MIND, RKMRI(R2)
2715 006634 005300          DEC      RO
2716 006636 001370          BNE     3$
2717 006640 012762 000040 000026      MOV      #DMD, RKMRI(R2)
2718 006646 005037 003622          CLR     BITCNT           ;; INITIALIZE BIT COUNT
2719 006652 012737 022040 003524      MOV      #DMD!MEND!ECCW, E.MRI ;; LOAD EXPECTED MAINTENANCE REG. 1
2720 006660 005037 003614          CLR     PR.BIT          ;; INITIALIZE PRESENT AND PREVIOUS
2721 006664 005037 003616          CLR     M1.BIT         ;; BITS TO GENERATE ZEROS
2722 006670 012700 000377          MOV      #255, RO       ;; GENERATE 255 ZEROES
2723 006674 004737 046156          JSR     PC, RDBIT       ;; READ A ZERO
2724 006700 016237 000026 003464 5$:   MOV      RKMRI(R2), T.MRI  ;; STORE MAINTENANCE REG. 1
2725 006706 023737 003524 003464      CMP      E.MRI, T.MRI    ;; CHECK READ GATE NOT SET
2726 006714 001405          BEQ     6$
2727 006716 104020          ERROR  2$
2728 006720 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ;; ISSUE CONTROLLER CLEAR
2729 006726 000464          BR     TST7              ;; GO ON TO NEXT TEST
2730
2731 006730 005237 003622          6$:   INC     BITCNT           ;; INCREMENT BIT COUNT
2732 006734 005300          DEC     RO              ;; CHECK IF ALL ZEROES ISSUED
2733 006736 001356          BNE     5$              ;; NO, CONTINUE
2734 006740 012737 000001 003614      MOV      #1, PR.BIT     ;; LOAD ONE FOR READING 1
2735 006746 004737 046156          JSR     PC, RDBIT       ;; READ A ONE
2736 006752 016237 000026 003464      MOV      RKMRI(R2), T.MRI ;; STORE MAINTENANCE REG. 1
2737 006760 023737 003524 003464      CMP      E.MRI, T.MRI    ;; CHECK READ GATE NOT SET
2738 006766 001405          BEQ     7$
2739 006770 104020          ERROR  2$
2740 006772 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ;; ISSUE CONTROLLER CLEAR
2741 007000 000437          BR     TST7              ;; GO ON TO NEXT TEST
2742

```

F05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 56
 DZR6CB.P11 28-JAN-77 14:04 T6 SECTOR PULSE DETECT IN READ HEADER (PART 2)

SEQ 0056

```

2743 007002 005237 003622 7$: INC BITCNT ;INCREMENT BIT COUNT
2744 007006 013737 003614 003616 MOV PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2745 007014 005037 003614 CLR PR.BIT
2746 007020 004737 046156 JSR PC,ROBIT ;SIMULATE ZERO
2747 007024 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2748 007032 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MAINTENANCE REG. 1 CORRECT
2749 007040 001404 BEQ 9$ ;CHECK CSI CORRECT
2750 007042 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2751 007050 000413 BR TST7 ;GO TO NEXT TEST
2752
2753 007052 016237 000000 003440 9$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2754 007060 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CSI
2755 007066 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CSI CORRECT
2756 007074 001401 BEQ TST7 ;YES, GO TO NEXT TEST
2757 007076 104016 ERROR 16 ;CSI INCORRECT
2758
2759
2760 *****
2761 *TEST 7 SECTOR PULSE DETECT IN READ HEADER (PART 3)
2762 *
2763 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2764 * DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
2765 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
2766 * CLOCK BOTH SEEK AND DRIVE CHECK MESSAGES.
2767 * SIMULATE 255 ZEROES AND A ONE.
2768 *
2769 * MAKE SURE READ GATE DOES NOT SET.
2770 *****
2771 007100 000004 TST7: SCOPE
2772 007102 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
2773 007110 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2774 007114 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2775 007122 012762 000040 000000 MOV #DMD,RKCS1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2776 007130 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2777 007136 012762 000040 000026 MOV #DMD,RKMR1(R2)
2778 007144 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2779 007152 012700 000312 MOV #50,*4+2,RO ;CLOCK UNTIL READY FOR SECTOR PULSE
2780 007156 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2781 007164 012762 000040 000026 MOV #DMD,RKMR1(R2)
2782 007172 005300 DEC RO
2783 007174 001370 BNE 1$
2784 007176 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2785 007204 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CSI
2786 007212 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2787 007220 001405 BEQ 2$ ;YES, CLOCK IN ZEROS
2788 007222 104015 ERROR 15
2789 007224 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2790 007232 000515 BR TST10 ;GO ON TO NEXT TEST
2791
2792 007234 005037 003622 2$: CLR BITCNT ;INITIALIZE BIT COUNT
2793 007240 012737 022040 003524 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MAINTENANCE REG. 1
2794 007246 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2795 007252 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROS
2796 007256 012700 000377 MOV #255,RO ;GENERATE 255 ZEROES
2797 007262 004737 046156 JSR PC,ROBIT ;READ A ZERO
2798 007266 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
  
```

G05

```

2799 007274 023737 003524 003464    CMP    E.MR1,T.MR1    ;CHECK READ GATE NOT SET
2800 007302 001405                    BEQ    6$            ;READ GATE NOT SET SIMULATE NEXT BIT
2801 007304 104021                    ERROR  21            ;MAINT REG. 1 INCORRECT
2802 007306 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2803 007314 000464                    BR     TST10        ;GO ON TO NEXT TEST
2804
2805 007316 005237 003622          6$:    INC    BITCNT      ;INCREMENT BIT COUNT
2806 007322 005300                    DEC    R0            ;CHECK IF ALL ZEROES ISSUED
2807 007324 001356                    BNE   5$            ;NO, CONTINUE
2808 007326 012737 000001 003614    MOV    #1,PR.BIT    ;LOAD ONE FOR READING 1
2809 007334 004737 046156                    JSR   PC,R0BIT      ;READ A ONE
2810 007340 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2811 007346 023737 003524 003464    CMP    E.MR1,T.MR1    ;CHECK READ GATE NOT SET
2812 007354 001405                    BEQ    7$            ;YES, CONTINUE
2813 007356 104021                    ERROR  21            ;MAINT REG. 1 INCORRECT
2814 007360 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2815 007366 000437                    BR     TST10        ;GO ON TO NEXT TEST
2816
2817 007370 005237 003622          7$:    INC    BITCNT      ;INCREMENT BIT COUNT
2818 007374 013737 003614 003616    MOV    PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2819 007402 005037 003614                    CLR   PR.BIT
2820 007406 004737 046156                    JSR   PC,R0BIT      ;SIMULATE ZERO
2821 007412 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2822 007420 023737 003524 003464    CMP    E.MR1,T.MR1    ;CHECK MAINTENANCE REG. 1 CORRECT
2823 007426 001404                    BEQ    9$            ;CHECK CSI CORRECT
2824 007430 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2825 007436 000413                    BR     TST10        ;GO TO NEXT TEST
2826
2827 007440 016237 000000 003440          9$:    MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2828 007446 012737 000025 003500    MOV    #ROHEAD,E.CS1 ;LOAD EXPECTED CSI
2829 007454 023737 003500 003440    CMP    E.CS1,T.CS1    ;CHECK CSI CORRECT
2830 007462 001401                    BEQ    TST10        ;YES, GO TO NEXT TEST
2831 007464 104016                    ERROR  16            ;CSI INCORRECT
2832
2833 *****
2834 *TEST 10      INDEX PULSE DETECTION IN WRITE HEADER
2835 *
2836 *          CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
2837 *          THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER
2838 *          TO AN RK06, IN 26 SECTOR FORMAT, TO CYLINDER 0, HEAD 0,
2839 *          DRIVE 0, WITH A ONE WORD TRANSFER.  CLOCK THROUGH THE
2840 *          SEEK AND THE DRIVE CLEAR MESSAGES.  ISSUE 200 CONTROLLER
2841 *          CLOCKS AND MAKE SURE WRITE GATE DOES NOT SET.  SIMULATE
2842 *          SECTOR PULSE AND 200 CONTROLLER CLOCKS MAKING SURE WRITE
2843 *          GATE DOES NOT SET.  SIMULATE INDEX PULSE AND MAKE SURE
2844 *          WRITE GATE SETS.
2845 *
2846 *****
2847 *          TST10:  SCOPE
2848 *          MOV    #10,$TIMES    ;DO 10. ITERATIONS
2849 *          MOV    $BASE,R2      ;LOAD RK611 BASE
2850 *          MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2851 *          MOV    #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2852 *          MOV    #DMD,$MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2853 *          MOV    #DMD,RKMR1(R2)
2854 *          MOV    #WRBUFF,RKBA(R2) ;LOAD BUS ADDRESS
    
```


2911	010072	023737	003502	003442	11\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT REG. CORRECT
2912	010100	001403				BEQ	12\$:YES, CONTINUE
2913	010102	104030				ERROR	30	:WORD COUNT INCORRECT
2914	010104	000137	010604			JMP	60\$:CLEAR RK611
2915								
2916	010110	012700	000004		12\$:	MOV	#4,RO	:SIMULATE SECTOR PULSE
2917	010114	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	
2918	010122	012762	000540	000026	13\$:	MOV	#DMD!MSP!MCLK,RKMR1(R2)	
2919	010130	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	
2920	010136	005300				DEC	RO	
2921	010140	001370				BNE	13\$	
2922	010142	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2923	010150	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2924	010156	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS AND REG.
2925	010164	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
2926	010172	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 INCORRECT
2927	010200	001402				BEQ	15\$:YES, CONTINUE
2928	010202	104026				ERROR	26	:CS1 INCORRECT
2929	010204	000577				BR	60\$:CLEAR RK611
2930								
2931	010206	023737	003504	003444	15\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
2932	010214	001402				BEQ	16\$:YES, CONTINUE
2933	010216	104027				ERROR	27	:BUS ADDRESS INCORRECT
2934	010220	000571				BR	60\$:CLEAR RK611
2935								
2936	010222	023737	003502	003442	16\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
2937	010230	001402				BEQ	20\$:YES, CONTINUE
2938	010232	104030				ERROR	30	:WORD COUNT INCORRECT
2939	010234	000563				BR	60\$:CLEAR RK611
2940								
2941	010236	005037	003622		20\$:	CLR	BITCNT	:INITIALIZE BIT COUNT
2942	010242	012700	000310			MOV	#200,RO	:ISSUE 200 MAINT BITS
2943	010246	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2944	010254	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2945	010262	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	
2946	010270	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2947	010276	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1
2948	010304	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINT REG. 1 CORRECT
2949	010312	001402				BEQ	22\$:YES, CONTINUE
2950	010314	104025				ERROR	25	:MAINT REG. 1 INCORRECT
2951	010316	000532				BR	60\$:CLEAR RK611
2952								
2953	010320	005300			22\$:	DEC	RO	:CHECK IF READY FOR INDEX PULSE
2954	010322	001351				BNE	21\$:NO, GET NEXT BIT
2955	010324	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2956	010332	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS
2957	010340	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
2958	010346	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
2959	010354	001402				BEQ	23\$:YES, CONTINUE
2960	010356	104026				ERROR	26	:CS1 INCORRECT
2961	010360	000511				BR	60\$:CLEAR RK611
2962								
2963	010362	023737	003504	003444	23\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
2964	010370	001402				BEQ	24\$:YES, CONTINUE
2965	010372	104027				ERROR	27	:BUS ADDRESS INCORRECT
2966	010374	000503				BR	60\$:CLEAR RK611

J05

```

2967
2968 010376 023737 003502 003442 24$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
2969 010404 001402 BEQ 25$ ;YES, CONTINUE
2970 010406 104030 ERROR 30 ;WORD COUNT INCORRECT
2971 010410 000475 BR 60$ ;CLEAR RK611
2972
2973 010412 012762 000240 000026 25$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE PULSE
2974 010420 012700 000004 MOV #4,R0
2975 010424 012762 000640 000026 26$: MOV #DMD!MIND!MCLK,RKMR1(R2)
2976 010432 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
2977 010440 005300 DEC R0
2978 010442 001370 BNE 26$
2979 010444 012762 000040 000026 MOV #DMD,RKMR1(R2)
2980 010452 012700 000002 MOV #2,R0 ;SIMULATE TWO CLOCK PULSES FOR WRITE
2981 ; GATE TO COME UP
2982 010456 012762 000440 000026 27$: MOV #DMD!MCLK,RKMR1(R2)
2983 010464 012762 000040 000026 MOV #DMD,RKMR1(R2)
2984 010472 005300 DEC R0
2985 010474 001370 BNE 27$
2986 010476 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.
2987 010504 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
2988 010512 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
2989 010520 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
2990 010526 012737 062040 003524 MOV #WRTGAT!MEND!E.MR1!DMD,E.MR1 ;LOAD EXPECTED MAINT REG. 1
2991 010534 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
2992 010542 001401 BEQ 28$ ;YES, CONTINUE
2993 010544 104031 ERROR 31 ;CS1 INCORRECT
2994 010546 023737 003504 003444 28$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
2995 010554 001401 BEQ 29$ ;YES, CONTINUE
2996 010556 104032 ERROR 32 ;BUS ADDRESS INCORRECT
2997 010560 023737 003502 003442 29$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
2998 010566 001401 BEQ 30$ ;YES, CONTINUE
2999 010570 104033 ERROR 33 ;WORD COUNT INCORRECT
3000 010572 023737 003524 003464 30$: CMP E.MR1,T.MR1 ;CHECK MAINT REG 1 CORRECT
3001 010600 001401 BEQ 60$ ;YES, CLEAR RK611
3002 010602 104034 ERROR 34 ;MAINT REG. 1 INCORRECT
3003 010604 012762 100000 000000 60$: MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

.SBTTL **NPR READING OF MEMORY

```

3004
3005
3006
3007
3008 :*****
3009 :*TEST 11 NPR OUTPUT DATA TRANSFER
3010 :*
3011 :* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
3012 :* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
3013 :* IN 26 SECTOR FORMAT, CYLINDER 777, HEAD 7, DRIVE 7.
3014 :* SPECIFY A ONE WORD DATA TRANSFER. CLOCK BOTH SEEK
3015 :* AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE.
3016 :* CLOCK IN FIRST WORD OF NPR TRANSFER. CHECK INPUT READY,
3017 :* OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF
3018 :* THE SILO. REPEAT FOR 8 DIFFERENT DATA PATTERNS.
3019 :*****
3020 :*ST11: SCOPE
3021 010612 000004 MOV #100,$TIMES ;DO 100. ITERATIONS
3022 010614 012737 000144 001200 MOV $BASE,R2 ;LOAD RK611 BASE
3023 010622 013702 001270

```

K05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 61
 DZR6CB.P11 28-JAN-77 14:04 T11 NPR OUTPUT DATA TRANSFER

SEQ 0061

3023	010626	012703	066734		MOV	#PATTERN,R3	:LOAD PATTERN ADDRESS
3024	010632	012704	000010		MOV	#8,R4	:LOAD PATTERN COUNT
3025	010636	012737	067256	003504	MOV	#WRBUFF+2,E.BA	:LOAD EXPECTED BUS ADDRESS
3026	010644	012737	000027	003500	MOV	#WHEAD,E.CS1	:LOAD EXPECTED CS1
3027	010652	005037	003502		CLR	E.WC	:LOAD EXPECTED WORD COUNT
3028	010656	012737	010664	001110	MOV	#15,\$LPERR	:LOAD LOOP ON ERROR LOCATION FOR
3029							:SUBTEST LOOP
3030							
3031	010664				15:		
3032	010664	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3033	010672	011337	067254		MOV	(R3),WRBUFF	:LOAD BUFFER FOR WRITE HEADER
3034	010676	012762	000040	000026	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN MAINT MODE
3035	010704	012762	000777	000020	MOV	#777,RKDCYL(R2)	:LOAD CYLINDER ADDRESS
3036	010712	012762	003400	000006	MOV	#3400,RKDA(R2)	:LOAD DISK ADDRESS
3037	010720	012762	067254	000004	MOV	#WRBUFF,RKBA(R2)	:LOAD BUS ADDRESS
3038	010726	012762	000007	000010	MOV	#7,RKCS2(R2)	:LOAD OTHER NUMBER
3039	010734	012762	177777	000002	MOV	#-1,RKWC(R2)	:LOAD WORD COUNT
3040	010742	012762	000027	000000	MOV	#WHEAD,RKCS1(R2)	:ISSUE WRITE HEADER
3041	010750	012700	000312		MOV	#50.#4+2,R0	:ISSUE CLOCKS UNTIL READY FOR
3042							:INDEX PULSE
3043	010754	012762	000440	000026	25:	MOV	#DMD:MCLK,RKMR1(R2)
3044	010762	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3045	010770	005300			DEC	R0	
3046	010772	001370			BNE	25	
3047	010774	012700	000004		MOV	#4,R0	:SIMULATE INDEX PULSE
3048	011000	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	
3049	011006	012762	000640	000026	35:	MOV	#DMD:MIND:MCLK,RKMR1(R2)
3050	011014	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	
3051	011022	005300			DEC	R0	
3052	011024	001370			BNE	35	
3053	011026	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3054	011034	012700	000045		MOV	#37,R0	:SIMULATE 1 NPR TRANSFER
3055	011040	012762	000440	000026	45:	MOV	#DMD:MCLK,RKMR1(R2)
3056	011046	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3057	011054	005300			DEC	R0	
3058	011056	001370			BNE	45	
3059	011060	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3060	011066	016237	000004	003444	MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG
3061	011074	016237	000002	003442	MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3062	011102	012700	000024		MOV	#20.,R0	:WAIT FOR OUTPUT READY
3063	011106	005300			55:	DEC	R0
3064	011110	001376			BNE	55	
3065	011112	016237	000010	003450	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3066	011120	012737	000307	003510	MOV	#OR!R!7,E.CS2	:LOAD EXPECTED CS2
3067	011126	023737	003500	003440	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
3068	011134	001401			BEQ	65	:YES, CHECK CS2
3069	011136	104035			ERROR	35	:CS1 INCORRECT
3070	011140	023737	003510	003450	65:	CMP	E.CS2,T.CS2
3071	011146	001401			BEQ	75	:YES, CONTINUE
3072	011150	104036			ERROR	36	:CS2 INCORRECT
3073	011152	023737	003504	003444	75:	CMP	E.BA,T.BA
3074	011160	001401			BEQ	85	:YES, CONTINUE
3075	011162	104037			ERROR	37	:BUS ADDRESS INCORRECT
3076	011164	023737	003502	003442	85:	CMP	E.WC,T.WC
3077	011172	001401			BEQ	95	:YES, CONTINUE
3078	011174	104040			ERROR	40	:WORD COUNT INCORRECT

```

3079 011176 016237 000024 003462 95:  MOV  RKDB(R2),T.DB ;READ DATA BUFFER
3080 011204 011337 003522          MOV  (R3),E.DB ;LOAD EXPECTED DATA BUFFER
3081 011210 023737 003522 003462  CMP  E.DB,T.DB ;CHECK IF DATA CORRECT
3082 011216 001401          BEQ  15$ ;YES, CONTINUE
3083 011220 104041          ERROR 41 ;DATA BUFFER INCORRECT
3084 011222 016237 000000 003440 15$:  MOV  RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3085 011230 016237 000010 003450  MOV  RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3086 011236 012737 000107 003510  MOV  #IR!7,E.CS2 ;LOAD EXPECTED CS2
3087 011244 023737 003500 003440  CMP  E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
3088 011252 001401          BEQ  17$ ;YES, CONTINUE
3089 011254 104042          ERROR 42 ;CS1 INCORRECT
3090 011256 023737 003510 003450 17$:  CMP  E.CS2,T.CS2 ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3091 011264 001401          BEQ  20$ ;YES, CONTINUE
3092 011266 104043          ERROR 43 ;CS2 INCORRECT
3093 011270 104415          20$: SCOPI ;CHECK IF LOOP ON ERROR
3094 011272 005723          TST  (R3)+ ;GENERATE ADDRESS OF NEXT CONFIG
3095 011274 005304          DEC  R4 ;CHECK IF ALL 8 CONFIGS TRIED
3096 011276 001000          BNE  TST12 ;;NO, TRY NEXT DATA PATTERN

```

```

*****
*TEST 12 PARTIAL SILO FILLING
*****

```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A ONE WORD DATA TRANSFER. CLOCK IN ALL
* SPECIFIED WORDS INTO THE SILO. CHECK WORD COUNT,
* BUS ADDRESS, INPUT READY, AND OUTPUT READY. MAKE
* SURE NO MORE THAN SPECIFIED DATA LENGTH IS CLOCKED
* INTO THE SILO. CHECK THE SILO FOR CORRECT DATA.
* REPEAT FOR WORD COUNTS 2-65.
*

```

```

3111
3112 011300 000004          TST12: SCOPE
3113 011302 012737 000144 001200  MOV  #100, $TIMES ;DO 100. ITERATIONS
3114 011310 013702 001270          MOV  $BASE,R2 ;LOAD RK611 BASE
3115 011314 012737 000001 001160  MOV  #1, $TMP0 ;LOAD NUMBER OF WORDS FOR DATA TRANSFER
3116 011322 012704 000065          MOV  #65, R4 ;LOAD ITERATION COUNT
3117 011326 012737 000027 003500  MOV  #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
3118 011334 012737 011342 001110  MOV  #15, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
3119                                     ; SUBTEST LOOP
3120
3121 011342          15:
3122 011342 012762 100000 000000  MOV  #CLR,RKCS1(R2) ;CLEAR RK611
3123 011350 012762 000040 000026  MOV  #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3124 011356 012762 067032 000004  MOV  #NPRBUF,RKBA(R2) ;LOAD BUS ADDRESS
3125 011364 012737 067032 003504  MOV  #NPRBUF,E.BA
3126 011372 013737 001160 003502  MOV  $TMP0,E.WC ;LOAD WORD COUNT
3127 011400 005437 003502          NEG  E.WC
3128 011404 013762 003502 000002  MOV  E.WC,RKWC(R2)
3129 011412 012762 000027 000000  MOV  #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3130 011420 012700 000312          MOV  #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
3131                                     ; INDEX PULSE
3132 011424 012762 000440 000026  25:  MOV  #DMD!MCLK,RKMR1(R2)
3133 011432 012762 000040 000026  MOV  #DMD,RKMR1(R2)
3134 011440 005300          DEC  R0

```

M05

3135	011442	001370				BNE	25	
3136	011444	012700	000004			MOV	#4, RO	: SIMULATE INDEX PULSE
3137	011450	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3138	011456	012762	000640	000026	35:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
3139	011464	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3140	011472	005300				DEC	RO	
3141	011474	001370				BNE	35	
3142	011476	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3143	011504	012737	000300	003510		MOV	#IR!OR, E.CS2	: LOAD EXPECTED CS2
3144	011512	012700	000045		45:	MOV	#37, RO	: SIMULATE 1 NPR TRANSFER
3145	011516	012762	000440	000026	55:	MOV	#DMD!MCLK, RKMR1(R2)	
3146	011524	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3147	011532	005300				DEC	RO	
3148	011534	001370				BNE	55	
3149	011536	016237	000000	003440		MOV	RKCS1(R2), T.CS1	: STORE COMMAND AND STATUS REG. 1
3150	011544	016237	000004	003444		MOV	RKBA(R2), T.BA	: STORE BUS ADDRESS
3151	011552	016237	000002	003442		MOV	RKWC(R2), T.WC	: STORE WORD COUNT
3152	011560	022737	067032	003504		CMP	#NPRBUF, E.BA	: CHECK IF FIRST WORD
3153	011566	001004				BNE	75	: NO, STORE CS2
3154	011570	012700	000024			MOV	#20., RO	: WAIT FOR OUTPUT READY
3155	011574	005300			65:	DEC	RO	
3156	011576	001376				BNE	65	
3157	011600	016237	000010	003450	75:	MOV	RKCS2(R2), T.CS2	: STORE COMMAND AND STATUS REG. 2
3158	011606	062737	000002	003504		ADD	#2, F.BA	: INCREMENT WORD COUNT AND BUS ADD
3159	011614	005237	003502			INC	E.W	
3160	011620	023737	003500	003440		CMP	E.CS1, T.CS1	: CHECK COMMAND STATUS REG. 1 CORRECT
3161	011626	001401				BEQ	85	: YES, CHECK CS2
3162	011630	104044				ERROR	44	: CS1 INCORRECT
3163	011632	023737	003510	003450	85:	CMP	E.CS2, T.CS2	: CHECK COMMAND STATUS REG. 2 CORRECT
3164	011640	001401				BEQ	95	: YES, CHECK BUSS ADDRESS REG.
3165	011642	104045				ERROR	45	: CS2 INCORRECT
3166	011644	023737	003504	003444	95:	CMP	E.BA, T.BA	: CHECK BUS ADDRESS CORRECT
3167	011652	001401				BEQ	105	: YES, CHECK WORD COUNT
3168	011654	104046				ERROR	46	: BUS ADDRESS INCORRECT
3169	011656	023737	003502	003442	105:	CMP	E.WC, T.WC	: CHECK WORD COUNT CORRECT
3170	011664	001401				BEQ	115	: YES, CHECK IF ALL WORDS TRANSFERRED
3171	011666	104047				ERROR	47	: WORD COUNT INCORRECT
3172	011670	005737	003502		115:	TST	E.WC	: CHECK IF FINISHED
3173	011674	001306				BNE	45	: NO, TRANSFER NEXT WORD
3174	011676	012700	000112			MOV	#2*37., RO	: ISSUE ENOUGH CLOCKS FOR
3175	011702	012762	000440	000026	155:	MOV	#DMD!MCLK, RKMR1(R2)	: 2 NPR TRANSFERS
3176	011710	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3177	011716	005300				DEC	RO	
3178	011720	001370				BNE	155	
3179	011722	016237	000000	003440		MOV	RKCS1(R2), T.CS1	: STORE COMMAND AND STATUS REG. 1
3180	011730	016237	000010	003450		MOV	RKCS2(R2), T.CS2	: STORE COMMAND AND STATUS REG. 2
3181	011736	016237	000004	003444		MOV	RKBA(R2), T.BA	: STORE BUS ADDRESS REG.
3182	011744	016237	000002	003442		MOV	RKWC(R2), T.WC	: STORE WORD COUNT
3183	011752	023737	003500	003440		CMP	E.CS1, T.CS1	: CHECK COMMAND STATUS REG. 1 CORRECT
3184	011760	001401				BEQ	165	: YES, CHECK CS2
3185	011762	104050				ERROR	50	: CS1 INCORRECT
3186	011764	023737	003510	003450	165:	CMP	E.CS2, T.CS2	: CHECK COMMAND STATUS REG. 2 CORRECT
3187	011772	001401				BEQ	175	: YES, CHECK BUS ADDRESS
3188	011774	104051				ERROR	51	: CS2 INCORRECT
3189	011776	023737	003504	003444	175:	CMP	E.BA, T.BA	: CHECK BUS ADDRESS CORRECT
3190	012004	001401				BEQ	185	: YES, CHECK WORD COUNT

B06

3247	012264	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	
3248	012272	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3249	012300	005300				DEC	RO	
3250	012302	001370				BNE	15	
3251	012304	012700	000004			MOV	#4,RO	;SIMULATE INDEX PULSE
3252	012310	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3253	012316	012762	000640	000026	25:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
3254	012324	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3255	012332	005300				DEC	RO	
3256	012334	001370				BNE	25	
3257	012336	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3258	012344	012737	000300	003510		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
3259	012352	012700	000045		45:	MOV	#37,RO	;SIMULATE 1 NPR TRANSFER
3260	012356	012762	000440	000026	55:	MOV	#DMD!MCLK,RKMR1(R2)	
3261	012364	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3262	012372	005300				DEC	RO	
3263	012374	001370				BNE	55	
3264	012376	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3265	012404	016237	000004	003444		MOV	RKBA(R2),↑.BA	;STORE BUS ADDRESS
3266	012412	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
3267	012420	022737	067034	003504		CMP	#NPRBUF+2,E.BA	;CHECK IF FIRST WORD
3268	012426	001004				BNE	75	;NO, STORE CS2
3269	012430	012700	000024			MOV	#20.,RO	;WAIT FOR OUTPUT READY
3270	012434	005300			65:	DEC	RO	
3271	012436	001376				BNE	65	
3272	012440	016237	000010	003450	75:	MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3273	012446	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK COMMAND STATUS REG. 1 CORRECT
3274	012454	001401				BEQ	85	;YES, CHECK CS2
3275	012456	104044				ERROR	44	;CS1 INCORRECT
3276	012460	023737	003510	003450	85:	CMP	E.CS2,T.CS2	;CHECK COMMAND STATUS REG. 2 CORRECT
3277	012466	001401				BEQ	95	;YES, CHECK BUS ADDRESS
3278	012470	104045				ERROR	45	;CS2 INCORRECT
3279	012472	023737	003504	003444	95:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
3280	012500	001401				BEQ	105	;YES, CHECK WORD COUNT
3281	012502	104046				ERROR	46	;BUS ADDRESS INCORRECT
3282	012504	023737	003502	003442	105:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
3283	012512	001401				BEQ	115	;YES, CHECK IF ALL WORDS TRANSFERRED
3284	012514	104047				ERROR	47	;WORD COUNT INCORRECT
3285	012516	062737	000002	003504	115:	ADD	#2,E.BA	;INCREMENT WORD COUNT AND BUS ADDRESS
3286	012524	005237	003502			INC	E.WC	
3287	012530	100710				BMI	45	;CHECK IF FINISHED (NO, BRANCH)
3288	012532	001004				BNE	125	;CHECK IF LAST WORD
3289	012534	012737	000200	003510		MOV	#OR,E.CS2	;LOAD EXPECTED CS2
3290	012542	000703				BR	45	;PROCESS LAST WORD
3291								
3292	012544	005037	003502		125:	CLR	E.WC	;ADJUST EXPECTED WORD COUNT
3293	012550	162737	000002	003504		SUB	#2,E.BA	;AND BUS ADDRESS
3294	012556	012700	000112			MOV	#2*37,RO	;ISSUE ENOUGH CLOCKS FOR
3295	012562	012762	000440	000026	155:	MOV	#DMD!MCLK,RKMR1(R2)	; 2 NPR TRANSFERS
3296	012570	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3297	012576	005300				DEC	RO	
3298	012600	001370				BNE	155	
3299	012602	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3300	012610	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3301	012616	016237	000004	003444		MOV	RKBA(R2),↑.BA	;STORE BUS ADDRESS REG
3302	012624	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT

3303	012632	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3304	012640	001401				BEQ	16\$:YES, CHECK CS2
3305	012642	104050				ERROR	50	:CS1 INCORRECT
3306	012644	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3307	012652	001401				BEQ	17\$:YES, CHECK BUS ADDRESS
3308	012654	104051				ERROR	51	:CS2 INCORRECT
3309	012656	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3310	012664	001401				BEQ	18\$:YES, CHECK WORD COUNT
3311	012666	104052				ERROR	52	:BUS ADDRESS INCORRECT
3312	012670	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3313	012676	001401				BEQ	19\$:YES, CHECK DATA
3314	012700	104053				ERROR	53	:WORD COUNT INCORRECT
3315	012702	012701	000102		19\$:	MOV	#66.,R1	:LOAD NUMBER OF WORDS LOADED
3316	012706	012703	067032			MOV	#NPRBUF,R3	:LOAD START ADDRESS
3317	012712	005037	003624			CLR	WROCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3318	012716	012737	000300	003510		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3319	012724	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3320	012732	012337	003522			MOV	(R3)+,E.DB	:LOAD EXPECTED DATA BUFFER
3321	012736	023737	003522	003462		CMP	E.DB,↑.DB	:CHECK IF DATA CORRECT
3322	012744	001401				BEQ	26\$:YES, CONTINUE
3323	012746	104054				ERROR	54	:DATA BUFFER INCORRECT
3324	012750	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
3325	012756	005737	003624			TST	WROCNT	:CHECK IF FIRST WORD
3326	012762	001015				BNE	28\$:NO, GET CS2
3327	012764	012700	000024			MOV	#20.,R0	:WAIT FOR INPUT READY TO SET
3328	012770	005300			27\$:	DEC	R0	
3329	012772	001376				BNE	27\$	
3330	012774	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3331	013002	022701	000001			CMP	#1,R1	:CHECK IF LAST WORD IN SILO
3332	013006	001003				BNE	28\$:NO, CONTINUE
3333	013010	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3334	013016	023737	003500	003440	28\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3335	013024	001401				BEQ	29\$:YES, CHECK CS2
3336	013026	104055				ERROR	55	:CS1 INCORRECT
3337	013030	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND AND STATUS REG. 2 CORRECT
3338	013036	001401				BEQ	30\$:YES, GET NEXT WORD
3339	013040	104056				ERROR	56	:CS2 INCORRECT
3340	013042	005237	003624		30\$:	INC	WROCNT	:INCREMENT WORD COUNT
3341	013046	005301				DEC	R1	:DECREMENT WORDS READ
3342	013050	001325				BNE	25\$:CHECK IF ALL WORDS READ

3343
 3344
 3345 :*****
 3346 :*TEST 14 SILO CAPACITY WITH NPR TRANSFERS
 3347 :*
 3348 :* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
 3349 :* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
 3350 :* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
 3351 :* SPECIFY A 66 WORD DATA TRANSFER. CLOCK IN 66
 3352 :* WORDS INTO THE SILO. MAKE SURE THAT SILO WILL STOP
 3353 :* FILLING AT 66 WORDS. TAKE ONE WORD FROM SILO AND
 3354 :* CHECK IT. CLOCK IN NEXT WORD. MAKE SURE NO MORE
 3355 :* THAN ONE WORD IS CLOCKED IN THE SILO. TAKE ONE
 3356 :* WORD FROM SILO AND CHECK IT. CLOCK IN NEXT WORD.
 3357 :* CLOCK IN NEXT WORD. MAKE SURE NO MORE THAN ONE WORD IS
 3358 :* CLOCKED IN THE SILO. TAKE ONE WORD FROM SILO AND
 :* CHECK IT. ATTEMPT TO CLOCK IN NEXT WORD AND

```

3359      * MAKE SURE NO WORDS ARE CLOCK INTO SILO. UNLOAD THE
3360      * SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.
3361      *
3362      * *****
3363      * ST14: SCOPE
3364      * MOV      #100, $TIMES      ; DO 100. ITERATIONS
3365      * MOV      $BASE, R2        ; LOAD RK611 BASE
3366      * MOV      $RHEAD, E.CS1    ; LOAD EXPECTED CS1
3367      * MOV      $CCLR, RKCS1(R2) ; CLEAR RK611
3368      * MOV      $DMD, RKMR1(R2)  ; PUT RK611 IN MAINT MODE
3369      * MOV      $NPRBUF+2, E.BA  ; LOAD BUS ADDRESS
3370      * MOV      $NPRBUF, RKBA(R2)
3371      * MOV      #-67, E.WC      ; LOAD WORD COUNT
3372      * MOV      #-68, RKWC(R2)
3373      * MOV      $RHEAD, RKCS1(R2) ; ISSUE WRITE HEADER
3374      * MOV      $50.*4+2, R0     ; ISSUE CLOCKS UNTIL READY FOR
3375      *                               ; INDEX PULSE
3376      * 1$: MOV      $DMD!MCLK, RKMR1(R2)
3377      * MOV      $DMD, RKMR1(R2)
3378      * DEC      R0
3379      * BNE     1$
3380      * MOV      #4, R0           ; SIMULATE INDEX PULSE
3381      * MOV      $DMD!MIND, RKMR1(R2)
3382      * 2$: MOV      $DMD!MIND!MCLK, RKMR1(R2)
3383      * MOV      $DMD!MIND, RKMR1(R2)
3384      * DEC      R0
3385      * BNE     2$
3386      * MOV      $DMD, RKMR1
3387      * MOV      $IR!OR, E.CS2    ; LOAD EXPECTED CS2
3388      * 4$: MOV      #37, R0     ; SIMULATE NPR TRANSFER
3389      * 5$: MOV      $DMD!MCLK, RKMR1(R2)
3390      * MOV      $DMD, RKMR1(R2)
3391      * DEC      R0
3392      * BNE     5$
3393      * MOV      RKCS1(R2), T.CS1  ; STORE COMMAND AND REG. 1
3394      * MOV      RKBA(R2), T.BA    ; STORE BUS ADDRESS
3395      * MOV      RKWC(R2), T.WC   ; STORE WORD COUNT
3396      * CMP      $NPRBUF+2, E.BA  ; CHECK IF FIRST WORD
3397      * BNE     7$                ; NO, STORE CS2
3398      * MOV      #20., R0        ; WAIT FOR OUTPUT READY
3399      * 6$: DEC      R0
3400      * BNE     6$
3401      * 7$: MOV      RKCS2(R2), T.CS2 ; STORE COMMAND AND STATUS REG. 2
3402      * CMP      E.CS1, T.CS1    ; CHECK COMMAND STATUS REG. 1 CORRECT
3403      * BEQ     8$                ; YES, CHECK CS2
3404      * ERROR   44                ; CS1 INCORRECT
3405      * 8$: CMP      E.CS2, T.CS2 ; CHECK COMMAND STATUS REG. 2 CORRECT
3406      * BEQ     9$                ; YES, CHECK BUS ADDRESS
3407      * ERROR   45                ; CS2 INCORRECT
3408      * 9$: CMP      E.BA, T.BA  ; CHECK BUS ADDRESS CORRECT
3409      * BEQ     10$               ; YES, CHECK WORD COUNT
3410      * ERROR   46                ; BUS ADDRESS INCORRECT
3411      * 10$: CMP     E.WC, T.WC  ; CHECK WORD COUNT CORRECT
3412      * BEQ     11$               ; YES, CHECK IF 1ST 65 WORDS TRANSFERRED
3413      * ERROR   47                ; WORD COUNT INCORRECT
3414      * 11$: ADD     #2, E.BA    ; INCREMENT BUS ADD AND WORD COUNT

```

E06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 68
 DZR6CB.P11 28-JAN-77 14:04 T14 SILO CAPACITY WITH NPR TRANSFERS

SEQ 0068

3415	013412	005237	003502			INC	E.WC	
3416	013416	022737	177776	003502		CMP	#-2,E.WC	;CHECK IF 65 WORDS IN SILO
3417	013424	101305				BHI	4\$;NO, GET NEXT WORD
3418	013426	001004				BNE	12\$;CHECK IF ALL 65 WORDS IN SILO
3419	013430	012737	000200	003510		MOV	#OR,E.CS2	;LOAD EXPECTED CS2
3420	013436	000700				BR	4\$;PROCESS 66TH WORD
3421								
3422	013440	005337	003502		12\$:	DEC	E.WC	;ADJUST WORD COUNT AND
3423	013444	162737	000002	003504		SUB	#2,E.BA	; BUS ADDRESS
3424	013452	012701	000003			MOV	#3,R1	
3425	013456	005037	003624			CLR	WROCNT	
3426	013462	012703	067032			MOV	#NPRBUF,R3	;LOAD START ADDRESS
3427	013466	012700	000112		13\$:	MOV	#2*37,R0	;ISSUE ENOUGH CLOCKS FOR
3428	013472	012762	000440	000026	15\$:	MOV	#DMD!MCLK,RKMR1(R2)	; 2 NPR TRANSFERS
3429	013500	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3430	013506	005300				DEC	R0	
3431	013510	001370				BNE	15\$	
3432	013512	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3433	013520	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3434	013526	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS REG
3435	013534	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
3436	013542	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK COMMAND STATUS REG. 1
3437	013550	001401				BEQ	16\$;YES, CHECK CS2
3438	013552	104050				ERROR	50	;CS1 INCORRECT
3439	013554	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	;CHECK COMMAND STATUS REG. 2 CORRECT
3440	013562	001401				BEQ	17\$;YES, CHECK BUS ADDRESS
3441	013564	104051				ERROR	51	;CS2 INCORRECT
3442	013566	023737	003504	003444	17\$:	CMP	E.BA,T.BA	;CHECK BUS ADD CORRECT
3443	013574	001401				BEQ	18\$;YES, CHECK WORD COUNT
3444	013576	104052				ERROR	52	;BUS ADDRESS INCORRECT
3445	013600	023737	003502	003442	18\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
3446	013606	001401				BEQ	19\$;YES, READ 1 WORD FROM SILO
3447	013610	104053				ERROR	53	;WORD COUNT INCORRECT
3448	013612	012737	000300	003510	19\$:	MOV	#IR!OR,E.CS2	;LOAD EXPECT CS2
3449	013620	016237	000024	003462		MOV	RKDB(R2),T.DB	;STORE DATA BUFFER
3450	013626	012337	003522			MOV	(R3)+,E.DB	;LOAD EXPECTED DATA BUFFER
3451	013632	023737	003522	003462		CMP	E.DB,T.DB	;CHECK IF DATA CORRECT
3452	013640	001401				BEQ	25\$;YES, CONTINUE
3453	013642	104054				ERROR	54	;DATA BUFFER INCORRECT
3454	013644	016237	000000	003440	25\$:	MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3455	013652	012700	000024			MOV	#20.,R0	;WAIT FOR OUTPUT READY
3456	013656	005300			26\$:	DEC	R0	
3457	013660	001376				BNE	26\$	
3458	013662	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3459	013670	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK IF COMMAND STATUS REG. 1 CORRECT
3460	013676	001401				BEQ	27\$;YES, CHECK CS2
3461	013700	104055				ERROR	55	;CS1 INCORRECT
3462	013702	023737	003510	003450	27\$:	CMP	E.CS2,T.CS2	;CHECK IF COMMAND STATUS REG. 2 CORRECT
3463	013710	001401				BEQ	30\$;YES, DO NPR TRANSFER
3464	013712	104056				ERROR	56	;CS2 INCORRECT
3465	013714	012700	000045		30\$:	MOV	#37,R0	;CLOCK IN ONE WORD (NPR TRANSFER)
3466	013720	012762	000440	000026	31\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3467	013726	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3468	013734	005300				DEC	R0	
3469	013736	001370				BNE	31\$	
3470	013740	022701	000001			CMP	#1.R1	;CHECK IF 68TH WORD READ

F06

```

3471 013744 001410          BEQ      32$          ;YES, NO NPR WILL TAKE PLACE
3472 013746 062737 000002 003504      ADD      #2, E.BA      ;INCREMENT BUS ADD AND WORD COUNT
3473 013754 005237 003502          INC      E.WC
3474 013760 012737 000200 003510      MOV      #0R, E.CS2    ;LOAD EXPECTED CS2
3475 013766 016237 000000 003440 32$:      MOV      RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
3476 013774 016237 000010 003450      MOV      RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
3477 014002 016237 000004 003444      MOV      RKBA(R2), ↑.BA  ;STORE BUS ADDRESS
3478 014010 016237 000002 003442      MOV      RKWC(R2), T.WC  ;STORE WORD COUNT
3479 014016 023737 003500 003440      CMP      E.CS1, T.CS1    ;CHECK COMMAND STATUS REG. 1 CORRECT
3480 014024 001401          BEQ      33$          ;YES, CHECK CS2
3481 014026 104044          ERROR    44          ;CS1 INCORRECT
3482 014030 023737 003510 003450 33$:      CMP      E.CS2, T.CS2    ;CHECK COMMAND STATUS REG. 2 CORRECT
3483 014036 001401          BEQ      34$          ;YES, CHECK BUS ADD
3484 014040 104045          ERROR    45          ;CS2 INCORRECT
3485 014042 023737 003504 003444 34$:      CMP      E.BA, T.BA      ;CHECK BUS ADD CORRECT
3486 014050 001401          BEQ      35$          ;YES, CHECK WORD COUNT
3487 014052 104046          ERROR    46          ;BUS ADD INCORRECT
3488 014054 023737 003502 003442 35$:      CMP      E.WC, T.WC      ;CHECK WORD COUNT CORRECT
3489 014062 001401          BEQ      36$          ;YES, CONTINUE
3490 014064 104047          ERROR    47          ;WORD COUNT INCORRECT
3491 014066 005237 003624          36$:      INC      WRDCNT
3492 014072 005301          DEC      R1            ;CHECK IF READ TO UNLOAD SILO
3493 014074 001402          BEQ      39$          ;YES, READ DATA; BUFFER
3494 014076 000137 013466          JMP      13$          ;NO, INPUT NEXT WORD
3495
3496 014102 162737 000002 003504 39$:      SUB      #2, E.BA      ;ADJUST WORD COUNT AND BUS ADDRESS
3497 014110 005037 003502          CLR      E.WC
3498 014114 012737 000300 003510      MOV      #IR!OR, E.CS2  ;LOAD EXPECTED CS2
3499 014122 012701 000101          MOV      #65, R1        ;LOAD NUMBER OF WORDS LEFT
3500 014126 016237 000024 003462 40$:      MOV      RKDB(R2), T.DB  ;READ DATA BUFFER
3501 014134 012337 003522          MOV      (R3)+, E.DB    ;LOAD EXPECTED DATA BUFFER
3502 014140 023737 003522 003462      CMP      E.DB, ↑.DB     ;CHECK IF DATA CORRECT
3503 014146 001401          BEQ      41$          ;YES, CHECK CS1
3504 014150 104054          ERROR    54          ;DATA BUFFER INCORRECT
3505 014152 016237 000000 003440 41$:      MOV      RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
3506 014160 022737 000002 003624      CMP      #2, WRDCNT     ;CHECK IF FIRST WORDS
3507 014166 001004          BNE      43$          ;NO, DO NOT WAIT FOR INPUT READY
3508 014170 012700 000024          MOV      #20., R0      ;WAIT FOR INPUT READY
3509 014174 005300          42$:      DEC      R0
3510 014176 001376          BNE      42$
3511 014200 016237 000010 003450 43$:      MOV      RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
3512 014206 022701 000001          CMP      #1, R1        ;CHECK IF LAST WORD
3513 014212 001003          BNE      44$          ;NO, CONTINUE
3514 014214 012737 000100 003510      MOV      #IR, E.CS2    ;LOAD EXPECTED CS2
3515 014222 023737 003500 003440 44$:      CMP      E.CS1, T.CS1    ;CHECK CS1 CORRECT
3516 014230 001401          BEQ      45$          ;YES, CHECK CS2
3517 014232 104055          ERROR    55          ;CS1 INCORRECT
3518 014234 023737 003510 003450 45$:      CMP      E.CS2, T.CS2    ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3519 014242 001401          BEQ      46$          ;YES, READ NEXT WORD ON SILO
3520 014244 104056          ERROR    56          ;CS2 INCORRECT
3521 014246 005237 003624          46$:      INC      WRDCNT        ;INCREMENT WORD COUNT
3522 014252 005301          DEC      R1            ;CHECK IF ALL WORDS READ
3523 014254 001324          BNE      40$          ;NO, READ NEXT WORD
3524
3525
3526
;*****
; *TEST 15      BUS ADDRESS INHIBIT

```

```

3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537 014256 000004
3538 014260 012737 000144 001200
3539 014266 013702 001270
3540 014272 012737 000027 003500
3541 014300 012762 100000 000000
3542 014306 012762 000040 000026
3543 014314 012737 067032 003504
3544 014322 012762 067032 000004
3545 014330 012737 177677 003502
3546 014336 012762 177676 000002
3547 014344 012762 000020 000010
3548 014352 012762 000027 000000
3549 014360 012700 000312
3550
3551 014364 012762 000440 000026 15:
3552 014372 012762 000040 000026
3553 014400 005300
3554 014402 001370
3555 014404 012700 000004
3556 014410 012762 000240 000026
3557 014416 012762 000640 000026 25:
3558 014424 012762 000240 000026
3559 014432 005300
3560 014434 001370
3561 014436 012762 000040 000026
3562 014444 012737 000320 003510
3563 014452 012700 000045 45:
3564 014456 012762 000440 000026 55:
3565 014464 012762 000040 000026
3566 014472 005300
3567 014474 001370
3568 014476 016237 000000 003440
3569 014504 016237 000004 003444
3570 014512 016237 000002 003442
3571 014520 022737 000101 003502
3572 014526 001004
3573 014530 012700 000024
3574 014534 005300 65:
3575 014536 001376
3576 014540 016237 000010 003450 75:
3577 014546 023737 003500 003440
3578 014554 001401
3579 014556 104057
3580 014560 023737 003510 003450 85:
3581 014566 001401
3582 014570 104060

```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN
* RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A 66 WORD DATA TRANSFER WITH BUS ADDRESS
* INHIBIT INCREMENT. CHECK WORD COUNT, BUS ADDRESS,
* INPUT READY, OUTPUT READY, AND MAKE SURE ALL THE
* WORDS IN THE SILO ARE THE CORRECT SAME WORD.
*
*****
*ST15: SCOPE
MOV #100, $TIMES ; DO 100 ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV $WHEAD, E.CS1 ; LOAD EXPECTED CS1
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #CMD, RKMR1(R2) ; PUT RK611 IN MAINT MODE
MOV $NPRBUF, E.BA ; LOAD BUS ADDRESS
MOV #65, E.WC ; LOAD WORD COUNT
MOV #66, RKWC(R2)
MOV $BA1, RKCS2(R2) ; SET BUS ADDRESS INCREMENT INHIBIT
MOV $WHEAD, RKCS1(R2) ; ISSUE WRITE HEADER
MOV #50, #4+2, R0 ; ISSUE CLOCKS UNTIL READY FOR
; INDEX PULSE
15: MOV #CMD, MCLK, RKMR1(R2)
MOV #CMD, RKMR1(R2)
DEC R0
BNE 15
MOV #4, R0 ; SIMULATE INDEX PULSE
MOV #CMD, MIND, RKMR1(R2)
25: MOV #CMD, MIND, MCLK, RKMR1(R2)
MOV #CMD, MIND, RKMR1(R2)
DEC R0
BNE 25
MOV #CMD, RKMR1(R2)
MOV #IR, OR, BAI, E.CS2 ; LOAD EXPECTED CS2
45: MOV #37, R0 ; SIMULATE 1 NPR TRANSFER
55: MOV #CMD, MCLK, RKMR1(R2)
MOV #CMD, RKMR1(R2)
DEC R0
BNE 55
MOV RKCS1(R2), T.CS1 ; STORE COMMAND AND STATUS REG. 1
MOV RKBA(R2), T.BA ; STORE BUS ADDRESS
MOV RKWC(R2), T.WC ; STORE WORD COUNT
CMP #65, E.WC ; CHECK IF FIRST WORD
BNE 75 ; NO, STORE CS2
MOV #20, R0 ; WAIT FOR OUTPUT READY
65: DEC R0
BNE 65
MOV RKCS2(R2), T.CS2 ; STORE COMMAND AND STATUS REG. 2
CMP E.CS1, T.CS1 ; CHECK COMMAND STATUS REG. 1 CORRECT
BEQ 85 ; YES, CHECK CS2
ERROR 57 ; CS1 INCORRECT
85: CMP E.CS2, T.CS2 ; CHECK COMMAND STATUS REG. 2 CORRECT
BEQ 95 ; YES, CHECK BUS ADDRESS
ERROR 60 ; CS2 INCORRECT

```

3583	014572	023737	003504	003444	95:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS
3584	014600	001401				BEQ	10\$:YES, CHECK WORD COUNT
3585	014602	104061				ERROR	61	:BUS ADDRESS INCORRECT
3586	014604	023737	003502	003442	105:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3587	014612	001401				BEQ	11\$:YES, CHECK IF ALL WORDS TRANSFERRED
3588	014614	104062				ERROR	62	:WORD COUNT INCORRECT
3589	014616	005237	003502		115:	INC	E.WC	:INCREMENT WORD COUNT
3590	014622	100713				BMI	4\$:CHECK IF FINISHED (NO, BRANCH)
3591	014624	001004				BNE	12\$:CHECK IF LAST WORD
3592	014626	012737	000220	003510		MOV	#OR!BAI,E.CS2	:LOAD EXPECTED COMMAND STATUS REG. 2
3593	014634	000706				BR	4\$:PROCESS THE LAST WORD
3594								
3595	014636	005037	003502		125:	CLR	E.WC	:ADJUST WORD COUNT
3596	014642	012700	000112			MOV	#2*37,RO	:ISSUE ENOUGH CLOCKS FOR
3597	014646	012762	000440	000026	155:	MOV	#CMD!MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3598	014654	012762	000040	000026		MOV	#CMD,RKMR1(R2)	
3599	014662	005300				DEC	RO	
3600	014664	001370				BNE	15\$	
3601	014666	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3602	014674	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3603	014702	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3604	014710	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
3605	014716	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3606	014724	001401				BEQ	16\$:YES, CHECK CS2
3607	014726	104063				ERROR	63	:CS1 INCORRECT
3608	014730	023737	003510	003450	165:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3609	014736	001401				BEQ	17\$:YES, CHECK BUS ADDRESS
3610	014740	104064				ERROR	64	:CS2 INCORRECT
3611	014742	023737	003504	003444	175:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3612	014750	001401				BEQ	18\$:YES, CHECK WORD COUNT
3613	014752	104065				ERROR	65	:BUS ADDRESS INCORRECT
3614	014754	023737	003502	003442	185:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3615	014762	001401				BEQ	19\$:YES, CHECK DATA
3616	014764	104066				ERROR	66	:WORD COUNT INCORRECT
3617	014766	012701	000102		195:	MOV	#66,R1	:LOAD NUMBERS OF WORDS LOADED
3618	014772	013737	067032	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3619	015000	005037	003624			CLR	WRDCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3620	015004	016237	000024	003462	255:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3621	015012	012737	000320	003510		MOV	#IR!OR!BAI,E.CS2	:LOAD EXPECTED CS2
3622	015020	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3623	015026	001401				BEQ	26\$:YES, CONTINUE
3624	015030	104067				ERROR	67	:DATA BUFFER INCORRECT
3625	015032	016237	000000	003440	265:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3626	015040	005737	003624			TST	WRDCNT	:CHECK IF FIRST WORD
3627	015044	001015				BNE	28\$:NO, GET CS2
3628	015046	012700	000024			MOV	#20.,RO	:WAIT FOR INPUT READY TO SET
3629	015052	005300			275:	DEC	RO	
3630	015054	001376				BNE	27\$	
3631	015056	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3632	015064	022701	000001			CMP	#1,R1	:CHECK IF LAST WORD
3633	015070	001003				BNE	28\$:NO, CONTINUE
3634	015072	012737	000120	003510		MOV	#IR!BAI,E.CS2	:LOAD EXPECTED CS2
3635	015100	023737	003500	003440	285:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3636	015106	001401				BEQ	29\$:YES, CHECK CS2
3637	015110	104070				ERROR	70	:CS1 INCORRECT
3638	015112	023737	003510	003450	295:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG 2 CORRECT

```

3639 015120 001401          BEQ      30$      ;YES, GET NEXT WORD
3640 015122 104071          ERROR    71      ;CS2 INCORRECT
3641 015124 005237 003624 30$: INC      WDCNT    ;INCREMENT WORD COUNT
3642 015130 005301          DEC      R1      ;DECREMENT WORDS READ
3643 015132 001324          BNE      25$      ;CHECK IF ALL WORDS READ
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656 015134 000004          *****
3657 015136 012737 000144 001200 *TEST 16      NON-EXISTENT MEMORY
3658 015144 013702 001270          *
3659 015150 012762 100000 000000          * CLEAR RK611 WITH A CONTROLLER CLEAR, PUT CONTROLLER
3660 015156 012762 000040 000026          * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
3661 015164 012737 160002 003504          * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
3662 015172 012762 160000 000004          * SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS
3663 015200 012737 177677 003502          * (760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR
3664 015206 012762 177676 000002          * OCCURS IN THE RK611 CONTROLLER.
3665 015214 012737 101626 003500          *****
3666 015222 012762 001427 000000          *ST16: SCOPE
3667 015230 012700 000312          MOV      #100, $TIMES ;DO 100 ITERATIONS
3668
3669 015234 012762 000440 000026 1$: MOV     #BASE, R2 ;LOAD RK611 BASE
3670 015242 012762 000040 000026          MOV     #CCLR, RKCS1(R2) ;CLEAR RK611
3671 015250 005300          MOV     #DMD, RKMRI(R2) ;PUT RK611 IN MAINT MODE
3672 015252 001370          MOV     #160002, E.BA ;LOAD BUS ADDRESS
3673 015254 012700 000004          MOV     #160000, RKBA(R2)
3674 015260 012762 000240 000026          MOV     #-65, E.WC ;LOAD WORD COUNT
3675 015266 012762 000640 000026 2$: MOV     #-66, RKWC(R2)
3676 015274 012762 000240 000026          MOV     #CERR!ROY!BA16!BA17!WRHEAD<+C<GO>>, E.CS1
3677 015302 005300          MOV     #BA17!BA16!WRHEAD, RKCS1(R2) ;ISSUE WRITE HEADER
3678 015304 001370          MOV     #50, *4+2, RO ;ISSUE CLOCKS UNTIL READY FOR
3679 015306 012762 000040 000026          ; INDEX PULSE
3680 015314 012700 000045          MOV     #DMD!MCLK, RKMRI(R2)
3681 015320 012762 000440 000026 3$: MOV     #DMD, RKMRI(R2)
3682 015326 012762 000040 000026          DEC     RO
3683 015334 005300          BNE     1$
3684 015336 001370          MOV     #4, RO ;SIMULATE INDEX PULSE
3685 015340 016237 000000 003440          MOV     #DMD!MIND, RKMRI(R2)
3686 015346 016237 000010 003450 2$: MOV     #DMD!MIND!MCLK, RKMRI(R2)
3687 015354 016237 000004 003444          MOV     #DMD!MIND, RKMRI(R2)
3688 015362 016237 000002 003442          DEC     RO
3689 015370 016237 000014 003454          BNE     2$
3690 015376 005037 003514          MOV     #DMD, RKMRI(R2)
3691 015402 012737 004100 003510          DEC     RO
3692 015410 032737 000200 003450          MOV     #DMD!MCLK, RKMRI(R2)
3693 015416 001403          MOV     #DMD, RKMRI(R2)
3694 015420 052737 000200 003510          DEC     RO
          BNE     5$
          MOV     RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
          MOV     RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
          MOV     RKBA(R2), T.BA ;STORE BUS ADDRESS
          MOV     RKWC(R2), T.WC ;STORE WORD COUNT
          MOV     RKER(R2), T.ER ;STORE ERROR REG.
          CLR     E.ER ;LOAD EXPECTED ERROR REG.
          MOV     #IR!NEM, E.CS2 ;LOAD EXPECTED CS2
          BIT     #OR, T.CS2
          BEQ     7$
          BIS     #OR, E.CS2

```

```

3695 015426 023737 003500 003440 75:  CMP      E.CS1,T.CS1  ;CHECK COMMAND STATUS REG. 1 CORRECT
3696 015434 001401          BEQ      85      ;YES, CHECK CS2
3697 015436 104072          ERROR    72      ;CS1 INCORRECT
3698 015440 023737 003510 003450 85:  CMP      E.CS2,T.CS2  ;CHECK COMMAND STATUS REG. 2 CORRECT
3699 015446 001401          BEQ      95      ;YES, CHECK ERROR REG.
3700 015450 104073          ERROR    73      ;CS2 INCORRECT
3701 015452 023737 003514 003454 95:  CMP      E.ER,T.ER    ;CHECK ERROR REG CORRECT
3702 015460 001401          BEQ     105      ;CHECK BUS ADDRESS
3703 015462 104074          ERROR    74      ;ERROR REG INCORRECT
3704 015464 023737 003504 003444 105: CMP      E.BA,T.BA    ;CHECK BUS ADDRESS CORRECT
3705 015472 001401          BEQ     115      ;YES, CHECK WORD COUNT
3706 015474 104075          ERROR    75      ;BUS ADDRESS INCORRECT
3707 015476 023737 003502 003442 115: CMP      E.WC,T.WC    ;CHECK WORD COUNT REG.
3708 015504 001401          BEQ     125      ;YES, CLEAR RK611
3709 015505 104076          ERROR    76      ;WORD COUNT INCORRECT
3710 015510 012762 100000 000000 125: MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3711 015516 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3712 015524 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3713 015532 012737 000200 003500      MOV      #RDY,E.CS1     ;LOAD EXPECTED CS1
3714 015540 012737 000100 003510      MOV      #IR,E.CS2     ;LOAD EXPECTED CS2
3715 015546 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG. 1
3716 015554 001401          BEQ     155      ;YES, CHECK CS2
3717 015556 104077          ERROR    77      ;CS1 INCORRECT
3718 015560 023737 003510 003450 155: CMP      E.CS2,T.CS2    ;CHECK IF MEM CLEARED
3719 015566 001401          BEQ     TST17     ;YES, GO ON TO NEXT TEST
3720 015570 104100          ERROR    100     ;CS2 INCORRECT

```

```

*****
*TEST 17      BUS ADDRESS BIT 16

```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A ONE WORD DATA TRANSFER FROM 200000.
* READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
* REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.
* CHECK BUS ADDRESS AND WORD COUNT.

```

```

* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
* OF MEMORY IS ON THE SYSTEM.

```

```

*****

```

```

3737 015572 000004          TST17: SCOPE
3738 015574 012737 000144 001200      MOV      #100,$TIMES   ;DO 100. ITERATIONS
3739 015602 005737 046366          TST      $K11         ;CHECK FOR MEMORY MANAGEMENT
3740 015606 100004          BPL      15          ;NO, BYPASS TEST
3741 015610 022737 002000 046634      CMP      #2000,$LSTBK ;CHECK IF ENOUGH MEMORY
3742 015616 103417          BLO      25          ;YES, DO TEST
3743 015620          ;
3744 015620 012737 000001 001200 15:  MOV      #1,$TIMES    ;FORCE INTERATION COUNT TO 1
3745 015626 005227 177777          INC      #-1         ;ONLY DO ONCE
3746 015632 001007          BNE      64$        ;NO, GO TO NEXT TEST
3747 015634 104401 055004          TYPE    TSTBY1     ;TYPE TEST N BYPASSED
3748 015640 013746 001220          MOV      $TESTN,-(SP) ;SAVE $TESTN FOR TYPEOUT
3749 015644 104402          TYPOC          ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3750 015646 104401 055014          TYPE    ,TSTBY2

```

K06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 74
 DZR6CB.P11 28-JAN-77 14:04 T17 BUS ADDRESS BIT 16

SEQ 0C74

```

3751 015652 000137 016706      64$: JMP      TST20          ;GO TO NEXT TEST
3752
3753 015656 013702 001270      2$:  MOV      $BASE,R2          ;LOAD K611 BASE
3754 015662 012737 002000 172354  MOV      #2000,KIPAR6        ;LOAD PAGE ADDRESS 6 FOR DATA
3755 015670 005237 177572          INC      SRO                  ;TURN ON MEMORY MANAGEMENT
3756 015674 013737 067032 140000  MOV      NPRBUF,140000       ;LOAD WORD IN MEMORY
3757 015702 005037 177572          CLR      SRO                  ;TURN OFF MEMORY MANAGEMENT
3758 015706 012737 015714 001110  MOV      #3$, $LPERR         ;LOAD LOOP ON ERROR LOCATION FOR
3759                                     ; SUBTEST LOOP
3760
3761 015714                                     3$:
3762 015714 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3763 015722 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3764 015730 012762 177777 000002  MOV      #-1,RKWC(R2)
3765 015736 012737 000427 003500  MOV      #BA16!WRHEAD,E.CS1 ;LOAD COMMAND
3766 015744 012762 000427 000000  MOV      #BA16!WRHEAD,RKCS1(R2)
3767 015752 012700 000312          MOV      #50,#4+2,R0        ;ISSUE ENOUGH CLOCKS UNTIL
3768 015756 012762 000440 000026  5$:  MOV      #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3769 015764 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3770 015772 005300          DEC      R0
3771 015774 001370          BNE     5$
3772 015776 012700 000          MOV      #4,R0              ;SIMULATE INDEX PULSE
3773 016002 012762 000040 000026  MOV      #DMD!MIND,RKMR1(R2)
3774 016010 012762 000640 000026  6$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
3775 016016 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
3776 016024 005300          DEC      R0
3777 016026 001370          BNE     6$
3778 016030 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3779 016036 012700 000045          MOV      #37,R0            ;ISSUE 1 NPR TRANSFER
3780 016042 012762 000440 000026  7$:  MOV      #DMD!MCLK,RKMR1(R2)
3781 016050 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3782 016056 005300          DEC      R0
3783 016060 001370          BNE     7$
3784 016062 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3785 016070 016237 000004 003444  MOV      RKBA(R2),T.BA ;STORE BUS ADDRESS
3786 016076 016237 000002 003442  MOV      RKWC(R2),T.WC ;STORE WOR
3787 016104 012700 000024          MOV      #20.,R0          ;WAIT FOR OUTPUT READY
3788 016110 005300      8$:  DEC      R0
3789 016112 001376          BNE     8$
3790 016114 016237 000010 003450  MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3791 016122 012737 000300 003510  MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3792 016130 012737 000002 003504  MOV      #2,E.BA ;LOAD EXPECTED BUS ADDRESS
3793 016136 005037 003502          CLR      E.WC ;LOAD EXPECTED WORD COUNT
3794 016142 023737 003500 003440  CMP      E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3795 016150 001401          BEQ     10$ ;YES, CHECK CS2
3796 016152 104101          ERROR  101 ;CS1 INCORRECT
3797 016154 023737 003510 003450  10$: CMP      E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
3798 016162 001401          BEQ     11$ ;YES, CHECK BUS ADDRESS CORRECT
3799 016164 104102          ERROR  102 ;CS2 INCORRECT
3800 016166 023737 003504 003444  11$: CMP      E.BA,T.BA ;CHECK IF BUS ADD INCORRECT
3801 016174 001401          BEQ     12$ ;YES, CHECK WORD COUNT CORRECT
3802 016176 104103          ERROR  103 ;BUS ADDRESS INCORRECT
3803 016200 023737 003502 003442  12$: CMP      E.WC,T.WC ;CHECK IF WORD COUNT CORRECT
3804 016206 001401          BEQ     13$ ;YES, CHECK DATA BUFFER
3805 016210 104104          ERROR  104 ;WORD COUNT INCORRECT
3806 016212 016237 000024 003462  13$: MOV      RKDB(R2),T.DB ;READ DATA BUFFER
    
```

3807	016220	013737	067032	003522		MOV	NPRBUF, E.DB	; LOAD EXPECTED DATA BUFFER
3808	016226	023737	003522	003462		CMP	E.DB, T.DB	; CHECK IF DATA CORRECT
3809	016234	001401				BEQ	15\$; YES, CHECK IF LOOP ON ERROR
3810	016236	104105				ERROR	105	; DATA INCORRECT
3811	016240	104415			15\$:	SCOP1		; CHECK IF LOOP ON ERROR
3812	016242	012737	001777	172354		MOV	#2000-1, KIPAR6	; LOAD PAGE ADDRESS 6 FOR DATA
3813	016250	005237	177572			INC	SRO	; TURN ON MEMORY MANAGEMENT
3814	016254	013737	067032	140076		MOV	NPRBUF, 140076	; LOAD WORDS IN MEMORY
3815	016262	013737	067034	140100		MOV	NPRBUF+2, 140100	
3816	016270	005037	177572			CLR	SRO	; TURN OFF MEMORY MANAGEMENT
3817	016274	012737	016302	001110		MOV	#20\$, \$LPERR	; LOAD LOOP ON ERROR LOCATION FOR
3818								; SUBTEST LOOP
3819								
3820	016302				20\$:			
3821	016302	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	; CLEAR RK611
3822	016310	012762	000040	000026		MOV	#DMD, RKMR1(R2)	; PUT RK611 IN DIAGNOSTIC MODE
3823	016316	012737	177777	003502		MOV	#-1, E.WC	; LOAD WORD COUNT REG.
3824	016324	012762	177776	000002		MOV	#-2, RKWC(R2)	
3825	016332	005037	003504			CLR	E.BA	; LOAD BUS ADDRESS
3826	016336	012762	177776	000004		MOV	#177776, RKBA(R2)	
3827	016344	012737	000427	003500		MOV	#BA16!WHEAD, E.CS1	; LOAD COMMAND
3828	016352	012762	000027	000000		MOV	#WHEAD, RKCS1(R2)	
3829	016360	012700	000312			MOV	#50, #4+2, RO	; ISSUE ENOUGH CLOCKS UNTIL
3830	016364	012762	000440	000026	21\$:	MOV	#DMD!MCLK, RKMR1(R2)	; INDEX PULSE
3831	016372	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3832	016400	005300				DEC	RO	
3833	016402	001370				BNE	21\$	
3834	016404	012700	000004			MOV	#4, RO	; SIMULATE INDEX PULSE
3835	016410	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3836	016416	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
3837	016424	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3838	016432	005300				DEC	RO	
3839	016434	001370				BNE	22\$	
3840	016436	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3841	016444	012701	000002			MOV	#2, R1	; ISSUE 2 NPR TRANSFERS
3842	016450	012700	000045		23\$:	MOV	#37, RO	
3843	016454	012762	000440	000026	24\$:	MOV	#DMD!MCLK, RKMR1(R2)	
3844	016462	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3845	016470	005300				DEC	RO	
3846	016472	001370				BNE	24\$	
3847	016474	016237	000000	003440		MOV	RKCS1(R2), T.CS1	; STORE COMMAND AND STATUS REG. 1
3848	016502	016237	000004	003444		MOV	RKBA(R2), T.BA	; STORE BUS ADDRESS REG.
3849	016510	016237	000002	003442		MOV	RKWC(R2), T.WC	; STORE WORD COUNT
3850	016516	005737	003502			TST	E.WC	; CHECK IF FIRST WORD
3851	016522	001404				BEQ	26\$; NO, GET CS2
3852	016524	012700	000024			MOV	#20, RO	; WAIT FOR OUTPUT READY
3853	016530	005300			25\$:	DEC	RO	
3854	016532	001376				BNE	25\$	
3855	016534	016237	000010	003450	26\$:	MOV	RKCS2(R2), T.CS2	; STORE COMMAND AND STATUS REG 2
3856	016542	012737	000300	003510		MOV	#R!OR, E.CS2	; LOAD EXPECTED CS2
3857	016550	023737	003500	003440		CMP	E.CS1, T.CS1	; CHECK COMMAND STATUS REG 1 CORRECT
3858	016556	001401				BEQ	28\$; YES, CHECK CS2
3859	016560	104101				ERROR	101	; CS1 INCORRECT
3860	016562	023737	003510	003450	28\$:	CMP	E.CS2, T.CS2	; CHECK COMMAND STATUS REG 2 CORRECT
3861	016570	001401				BEQ	29\$; YES, CHECK BUS ADDRESS
3862	016572	104102				ERROR	102	; CS2 INCORRECT

M06

```

3863 016574 023737 003504 003444 29$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3864 016602 001401 BEQ 30$ ;YES, CHECK WORD COUNT
3865 016604 104103 ERROR 103 ;BUS ADDRESS INCORRECT
3866 016606 023737 003502 003442 30$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3867 016614 001401 BEQ 31$ ;YES, GET TEXT WORD
3868 016616 104104 ERROR 104 ;WORD COUNT INCORRECT
3869 016620 062737 000002 003504 31$: ADD #2,E.BA ;INCREMENT BUS ADDRESS AND
3870 016626 005237 003502 INC E.WC ;WORD COUNT
3871 016632 005301 DEC R1 ;CHECK IF FINISHED
3872 016634 001305 BNE 23$ ;NO, GET SECOND WORD
3873 016636 012700 000002 MOV #2,R0 ;LOAD COUNT AND ADDRESS WORD
3874 016642 012703 067032 MOV #NPRBUF,R3 ;DATA COMPARE
3875 016646 016237 000024 003462 35$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
3876 016654 012337 003522 MOV (R3)+,E.DB ;GET EXPECTED DATA
3877 016660 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF CORRECT
3878 016666 001401 BEQ 36$ ;YES, CHECK IF FINISHED
3879 016670 104105 ERROR 105 ;DATA BUFFER INCORRECT
3880 016672 005300 36$: DEC R0 ;CHECK IF FINISHED
3881 016674 001364 BNE 35$ ;NO READ SECOND WORD
3882 016676 104415 SCOP1 ;CHECK IF LOOP ON ERROR
3883 016700 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

```

*****
*TEST 20 BUS ADDRESS BIT 17
*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0.
* SPECIFY A ONE WORD DATA TRANSFER FROM 40000.
* READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
* REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 377776.
* CHECK BUS ADDRESS AND WORD COUNT.
*
* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
* OF MEMORY IS ON THE SYSTEM.
*****

```

```

3899 *****
3900 016706 000004 TST20: SCOPE
3901 016710 012737 000144 001200 MOV #100, $TIMES ;DO 100. ITERATIONS
3902 016716 005737 046366 TST $KT11 ;CHECK FOR MEMORY MANAGEMENT
3903 016722 100004 BPL 1$ ;NO, BYPASS TEST
3904 016724 022737 004000 046634 CMP #4000, $LSTBK ;CHECK IF ENOUGH MEMORY
3905 016732 103417 BLO 2$ ;YES, DO TEST
3906 016734 1$:
3907 016734 012737 000001 001200 MOV #1, $TIMES ;FORCE INTERATION COUNT TO 1
3908 016742 005227 177777 INC #-1 ;ONLY DO ONCE
3909 016746 001007 BNE 64$ ;NO, GO TO NEXT TEST
3910 016750 104401 055004 TYPE TSTBY1 ;TYPE TEST N BYPASSED
3911 016754 013746 001220 MOV $TESTN, -(SP) ;SAVE $TESTN FOR TYPEOUT
3912 016760 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3913 016762 104401 055014 TYPE TSTBY2
3914 016766 000137 020022 64$: JMP TST21 ;GO TO NEXT TEST
3915
3916 016772 013702 001270 2$: MOV $BASE, R2 ;LOAD K611 BASE
3917 016776 012737 004000 172354 MOV #4000, KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
3918 017004 005237 177572 INC SRO ;TURN ON MEMORY MANAGEMENT

```

N06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 77
 DZR6CB.P11 28-JAN-77 14:04 T20 BUS ADDRESS BIT 17

SEQ 0077

3919	017010	013737	067032	140000		MOV	NPRBUF,140000	:LOAD WORD IN MEMORY
3920	017016	005037	177572			CLR	SRO	:TURN OFF MEMORY MANAGEMENT
3921	017022	012737	017030	001110		MOV	#3\$,SLPERR	:LOAD LOOP ON ERROR LOCATION FOR
3922								:SUBTEST LOOP
3923								
3924	017030				3\$:			
3925	017030	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
3926	017036	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3927	017044	012762	177777	000002		MOV	#-1,RKWC(R2)	
3928	017052	012737	001027	003500		MOV	#BA17!,RHEAD,E.CS1	:LOAD COMMAND
3929	017060	012762	001027	000000		MOV	#BA17!,RHEAD,RKCS1(R2)	
3930	017066	012700	000312			MOV	#50,#4+2,RJ	:ISSUE ENOUGH CLOCKS UNTIL
3931	017072	012762	000440	000026	5\$:	MOV	#DMD!MCLK,RKMR1(R2)	:INDEX PULSE
3932	017100	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3933	017106	005300				DEC	RO	
3934	017110	001370				BNE	5\$	
3935	017112	012700	000004			MOV	#4,RO	:SIMULATE INDEX PULSE
3936	017116	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3937	017124	012762	000640	000026	6\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
3938	017132	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3939	017140	005300				DEC	RO	
3940	017142	001370				BNE	6\$	
3941	017144	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3942	017152	012700	000045			MOV	#37,RO	:ISSUE 1 NPR TRANSFER
3943	017156	012762	000440	000026	7\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3944	017164	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3945	017172	005300				DEC	RO	
3946	017174	001370				BNE	7\$	
3947	017176	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3948	017204	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS
3949	017212	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WOR
3950	017220	012700	000024			MOV	#20.,RO	:WAIT FOR OUTPUT READY
3951	017224	005300			8\$:	DEC	RO	
3952	017226	001376				BNE	8\$	
3953	017230	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3954	017236	012737	000300	003510		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3955	017244	012737	000002	003504		MOV	#2,E.BA	:LOAD EXPECTED BUS ADDRESS
3956	017252	005037	003502			CLR	E.WC	:LOAD EXPECTED WORD COUNT
3957	017256	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3958	017264	001401				BEQ	10\$:YES, CHECK CS2
3959	017266	104101				ERROR	101	:CS1 INCORRECT
3960	017270	023737	003510	003450	10\$:	CMP	E.CS2,T.CS2	:CHECK IF CS2 CORRECT
3961	017276	001401				BEQ	11\$:YES, CHECK BUS ADDRESS CORRECT
3962	017300	104102				ERROR	102	:CS2 INCORRECT
3963	017302	023737	003504	003444	11\$:	CMP	E.BA,T.BA	:CHECK IF BUS ADDRESS INCORRECT
3964	017310	001401				BEQ	12\$:YES, CHECK WORD COUNT CORRECT
3965	017312	104103				ERROR	103	:BUS ADDRESS INCORRECT
3966	017314	023737	003502	003442	12\$:	CMP	E.WC,T.WC	:CHECK IF WORD COUNT CORRECT
3967	017322	001401				BEQ	13\$:YES, CHECK DATA BUFFER
3968	017324	104104				ERROR	104	:WORD COUNT INCORRECT
3969	017326	016237	000024	003462	13\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3970	017334	013737	067032	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3971	017342	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3972	017350	001401				BEQ	15\$:YES, CHECK IF LOOP ON ERROR
3973	017352	104105				ERROR	105	:DATA INCORRECT
3974	017354	104415			15\$:	SCOPI		:CHECK IF LOOP ON ERROR

3975	017356	012737	003777	172354		MOV	#4000-1, KIPAR6	: LOAD PAGE ADDRESS 6 FOR DATA
3976	017364	005237	177572			INC	SRO	: TURN ON MEMORY MANAGEMENT
3977	017370	013737	067032	140076		MOV	NPRBUF, 140076	: LOAD WORDS IN MEMORY
3978	017376	013737	067034	140100		MOV	NPRBUF+2, 140100	
3979	017404	005037	177572			CLR	SRO	: TURN OFF MEMORY MANAGEMENT
3980	017410	012737	017416	001110		MOV	#20\$, SLPERR	: LOAD LOOP ON ERROR LOCATION FOR
3981								: SUBTEST LOOP
3982								
3983	017416				20\$:			
3984	017416	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	: CLEAR RK611
3985	017424	012762	000040	000026		MOV	#DMD, RKMR1(R2)	: PUT RK611 IN DIAGNOSTIC MODE
3986	017432	012737	177777	003502		MOV	#-1, E.WC	: LOAD WORD COUNT REG.
3987	017440	012762	177776	000002		MOV	#-2, RKWC(R2)	
3988	017446	005037	003504			CLR	E.BA	: LOAD BUS ADDRESS
3989	017452	012762	177776	000004		MOV	#177776, RKBA(R2)	
3990	017460	012737	001027	003500		MOV	#BA17!WRHEAD, E.CS1	: LOAD COMMAND
3991	017466	012762	000427	000000		MOV	#BA16!WRHEAD, RKCS1(R2)	
3992	017474	012700	000312			MOV	#50, #4+2, RO	: ISSUE ENOUGH CLOCKS UNTIL
3993	017500	012762	000440	000026	21\$:	MOV	#DMD!MCLK, RKMR1(R2)	: INDEX PULSE
3994	017506	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3995	017514	005300				DEC	RO	
3996	017516	001370				BNE	21\$	
3997	017520	012700	000004			MOV	#4, RO	: SIMULATE INDEX PULSE
3998	017524	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3999	017532	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
4000	017540	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
4001	017546	005300				DEC	RO	
4002	017550	001370				BNE	22\$	
4003	017552	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4004	017560	012701	000002			MOV	#2, R1	: ISSUE 2 NPR TRANSFERS
4005	017564	012700	000045		23\$:	MOV	#37, RO	
4006	017570	012762	000440	000026	24\$:	MOV	#DMD!MCLK, RKMR1(R2)	
4007	017576	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4008	017604	005300				DEC	RO	
4009	017606	001370				BNE	24\$	
4010	017610	016237	000000	003440		MOV	RKCS1(R2), T.CS1	: STORE COMMAND AND STATUS REG. 1
4011	017616	016237	000004	003444		MOV	RKBA(R2), T.BA	: STORE BUS ADDRESS REG.
4012	017624	016237	000002	003442		MOV	RKWC(R2), T.WC	: STORE WORD COUNT
4013	017632	005737	003502			TST	E.WC	: CHECK IF FIRST WORD
4014	017636	001404				BEQ	26\$: NO, GET CS2
4015	017640	012700	000024			MOV	#20., RO	: WAIT FOR OUTPUT READY
4016	017644	005300			25\$:	DEC	RO	
4017	017646	001376				BNE	25\$	
4018	017650	016237	000010	003450	26\$:	MOV	RKCS2(R2), T.CS2	: STORE COMMAND AND STATUS REG 2
4019	017656	012737	000300	003510		MOV	#IR!OR, E.CS2	: LOAD EXPECTED CS2
4020	017664	023737	003500	003440		CMP	E.CS1, T.CS1	: CHECK COMMAND STATUS REG 1 CORRECT
4021	017672	001401				BEQ	28\$: YES, CHECK CS2
4022	017674	104101				ERROR	101	: CS1 INCORRECT
4023	017676	023737	003510	003450	28\$:	CMP	E.CS2, T.CS2	: CHECK COMMAND STATUS REG 2 CORRECT
4024	017704	001401				BEQ	29\$: YES, CHECK BUS ADDRESS
4025	017706	104102				ERROR	102	: CS2 INCORRECT
4026	017710	023737	003504	003444	29\$:	CMP	E.BA, T.BA	: CHECK BUS ADDRESS CORRECT
4027	017716	001401				BEQ	30\$: YES, CHECK WORD COUNT
4028	017720	104103				ERROR	103	: BUS ADDRESS INCORRECT
4029	017722	023737	003502	003442	30\$:	CMP	E.WC, T.WC	: CHECK WORD COUNT CORRECT
4030	017730	001401				BEQ	31\$: YES, GET TEXT WORD

4087	020144				3\$:		
4088	020144	012762	100000	000000		MOV	#CLR,RKCS1(R2) ;CLEAR RK611
4089	020152	012762	000040	000026		MOV	#DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4090	020160	012762	177777	000002		MOV	#-1,RKWC(R2)
4091	020166	012737	001427	003500		MOV	#BA16:BA17:WRHEAD,E.CS1 ;LOAD COMMAND
4092	020174	012762	001427	000000		MOV	#BA16:BA17:WRHEAD,RKCS1(R2)
4093	020202	012700	000312			MOV	#50,#4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
4094	020206	012762	000440	000026	5\$:	MOV	#DMD:MCLK,RKMR1(R2) ; INDEX PULSE
4095	020214	012762	000040	000026		MOV	#DMD,RKMR1(R2)
4096	020222	005300				DEC	RO
4097	020224	001370				BNE	5\$
4098	020226	012700	000004			MOV	#4,RO ;SIMULATE INDEX PULSE
4099	020232	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)
4100	020240	012762	000640	000026	6\$:	MOV	#DMD:MIND:MCLK,RKMR1(R2)
4101	020246	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)
4102	020254	005300				DEC	RO
4103	020256	001370				BNE	6\$
4104	020260	012762	000040	000026		MOV	#DMD,RKMR1(R2)
4105	020266	012700	000045			MOV	#37,RO ;ISSUE 1 NPR TRANSFER
4106	020272	012762	000440	000026	7\$:	MOV	#DMD:MCLK,RKMR1(R2)
4107	020300	012762	000040	000026		MOV	#DMD,RKMR1(R2)
4108	020306	005300				DEC	RO
4109	020310	001370				BNE	7\$
4110	020312	016237	000000	003440		MOV	RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4111	020320	016237	000004	003444		MOV	RKBA(R2),T.BA ;STORE BUS ADDRESS
4112	020326	016237	000002	003442		MOV	RKWC(R2),T.WC ;STORE WOR
4113	020334	012700	000024			MOV	#20,RO ;WAIT FOR OUTPUT READY
4114	020340	005300			8\$:	DEC	RO
4115	020342	001376				BNE	8\$
4116	020344	016237	000010	003450		MOV	RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4117	020352	012737	000300	003510		MOV	#IR:OR,E.CS2 ;LOAD EXPECTED CS2
4118	020360	012737	000002	003504		MOV	#2,E.BA ;LOAD EXPECTED BUS ADDRESS
4119	020366	005037	003502			CLR	E.WC ;LOAD EXPECTED WORD COUNT
4120	020372	023737	003500	003440		CMP	E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
4121	020400	001401				BEQ	10\$
4122	020402	104101				ERROR	101 ;YES, CHECK CS2
4123	020404	023737	003510	003450	10\$:	CMP	E.CS2,T.CS2 ;CS1 INCORRECT
4124	020412	001401				BEQ	11\$;CHECK IF CS2 CORRECT
4125	020414	104102				ERROR	102 ;YES, CHECK BUS ADDRESS CORRECT
4126	020416	023737	003504	003444	11\$:	CMP	E.BA,T.BA ;CS2 INCORRECT
4127	020424	001401				BEQ	12\$;CHECK IF BUS ADD INCORRECT
4128	020426	104103				ERROR	103 ;YES, CHECK WORD COUNT CORRECT
4129	020430	023737	003502	003442	12\$:	CMP	E.WC,T.WC ;BUS ADDRESS INCORRECT
4130	020436	001401				BEQ	13\$;CHECK IF WORD COUNT CORRECT
4131	020440	104104				ERROR	104 ;YES, CHECK DATA BUFFER
4132	020442	016237	000024	003462	13\$:	MOV	RKDB(R2),T.DB ;WORD COUNT INCORRECT
4133	020450	013737	067032	003522		MOV	NPRBUF,E.DB ;READ DATA BUFFER
4134	020456	023737	003522	003462		CMP	E.DB,T.DB ;LOAD EXPECTED DATA BUFFER
4135	020464	001401				BEQ	15\$;CHECK IF DATA CORRECT
4136	020466	104105				ERROR	105 ;YES, CHECK IF LOOP ON ERROR
4137	020470	104415			15\$:	SCOP1 ;DATA INCORRECT	
4138	020472	012737	005777	172354		MOV	#6000-1,KIPAR6 ;CHECK IF LOOP ON ERROR
4139	020500	005237	177572			INC	SRO ;LOAD PAGE ADDRESS 6 FOR DATA
4140	020504	013737	067032	140076		MOV	NPRBUF,140076 ;TURN ON MEMORY MANAGEMENT
4141	020512	013737	067034	140100		MOV	NPRBUF+2,140100 ;LOAD WORDS IN MEMORY
4142	020520	005037	177572			CLR	SRO ;TURN OFF MEMORY MANAGEMENT

E07

4143	020524	012737	020532	001110		MOV	#20\$, \$LPERR	;LOAD LOOP ON ERROR LOCATION FOR
4144								; SUBTEST LOOP
4145								
4146	020532				20\$:			
4147	020532	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	;CLEAR RK611
4148	020540	012762	000040	000026		MOV	#DMD, RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
4149	020546	012737	177777	003502		MOV	#-1, E.WC	;LOAD WORD COUNT REG.
4150	020554	012762	177776	000002		MOV	#-2, RKWC(R2)	
4151	020562	005037	003504			CLR	E.BA	;LOAD BUS ADDRESS
4152	020566	012762	177776	000004		MOV	#177776, RKBA(R2)	
4153	020574	012737	001427	003500		MOV	#BA16!BA17!WRHEAD, E.CS1	;LOAD COMMAND
4154	020602	012762	001027	000000		MOV	#BA17!WRHEAD, RKCS1(R2)	
4155	020610	012700	000312			MOV	#50, #4+2, R0	;ISSUE ENOUGH CLOCKS UNTIL
4156	020614	012762	000440	000026	21\$:	MOV	#DMD!MCLK, RKMR1(R2)	; INDEX PULSE
4157	020622	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4158	020630	005300				DEC	R0	
4159	020632	001370				BNE	21\$	
4160	020634	012700	000004			MOV	#4, R0	;SIMULATE INDEX PULSE
4161	020640	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
4162	020646	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
4163	020654	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
4164	020662	005300				DEC	R0	
4165	020664	001370				BNE	22\$	
4166	020666	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4167	020674	012701	000002			MOV	#2, R1	;ISSUE 2 NPR TRANSFERS
4168	020700	012700	000045		23\$:	MOV	#37, R0	
4169	020704	012762	000440	000026	24\$:	MOV	#DMD!MCLK, RKMR1(R2)	
4170	020712	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4171	020720	005300				DEC	R0	
4172	020722	001370				BNE	24\$	
4173	020724	016237	000000	003440		MOV	RKCS1(R2), T.CS1	;STORE COMMAND AND STATUS REG. 1
4174	020732	016237	000004	003444		MOV	RKBA(R2), T.BA	;STORE BUS ADDRESS REG.
4175	020740	016237	000002	003442		MOV	RKWC(R2), T.WC	;STORE WORD COUNT
4176	020746	005737	003502			TST	E.WC	;CHECK IF FIRST WORD
4177	020752	001404				BEQ	26\$;NO, GET CS2
4178	020754	012700	000024			MOV	#20., R0	;WAIT FOR OUTPUT READY
4179	020760	005300			25\$:	DEC	R0	
4180	020762	001376				SNE	25\$	
4181	020764	016237	000010	003450	26\$:	MOV	RKCS2(R2), T.CS2	;STORE COMMAND AND STATUS REG 2
4182	020772	012737	000300	003510		MOV	#IR!OR, E.CS2	;LOAD EXPECTED CS2
4183	021000	023737	003500	003440		CMP	E.CS1, T.CS1	;CHECK COMMAND STATUS REG 1 CORRECT
4184	021006	001401				BEQ	28\$;YES, CHECK CS2
4185	021010	104101				ERROR	101	;CS1 INCORRECT
4186	021012	023737	003510	003450	28\$:	CMP	E.CS2, T.CS2	;CHECK COMMAND STATUS REG 2 CORRECT
4187	021020	001401				BEQ	29\$;YES, CHECK BUS ADDRESS
4188	021022	104102				ERROR	102	;CS2 INCORRECT
4189	021024	023737	003504	003444	29\$:	CMP	E.BA, T.BA	;CHECK BUS ADDRESS CORRECT
4190	021032	001401				BEQ	30\$;YES, CHECK WORD COUNT
4191	021034	104103				ERROR	103	;BUS ADDRESS INCORRECT
4192	021036	023737	003502	003442	30\$:	CMP	E.WC, T.WC	;CHECK WORD COUNT CORRECT
4193	021044	001401				BEQ	31\$;YES, GET TEXT WORD
4194	021046	104104				ERROR	104	;WORD COUNT INCORRECT
4195	021050	062737	000002	003504	31\$:	ADD	#2, E.BA	;INCREMENT BUS ADDRESS AND
4196	021056	005237	003502			INC	E.WC	WORD COUNT
4197	021062	005301				DEC	R1	;CHECK IF FINISHED
4198	021064	001305				BNE	23\$;NO, GET SECOND WORD

F07

```

4199 021066 012700 000002      MOV      #2,R0      ;LOAD COUNT AND ADDRESS WORD
4200 021072 012703 067032      MOV      #NPRBUF,R3 ; DATA COMPARE
4201 021076 016237 000024 003462 35$: MOV      RKDB(R2),T.DB ; READ DATA BUFFER
4202 021104 012337 003522      MOV      (R3)+,E.DB ; GET EXPECTED DATA
4203 021110 023737 003522 003462  CMP      E.DB,T.DB ; CHECK IF CORRECT
4204 021116 001401      BEQ      36$      ; YES, CHECK IF FINISHED
4205 021120 104105      ERROR    105     ; DATA BUFFER INCORRECT
4206 021122 005300 36$: DEC      R0      ; CHECK IF FINISHED
4207 021124 001364      BNE      35$     ; NO READ SECOND WORD
4208 021126 104415      SCOPE1 ; CHECK IF LOOP ON ERROR
4209 021130 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ; CLEAR RK611
4210
4211 ;*****
4212 ;TEST 22 UNIBUS PARITY ERROR
4213 ;
4214 ; CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4215 ; IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
4216 ; IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4217 ; SPECIFY A ONE WORD DATA TRANSFER FROM A LOCATION
4218 ; WITH BAD PARITY. MAKE SURE A UNIBUS PARITY ERROR
4219 ; OCCURS.
4220 ;
4221 ; REPEAT FOR A ONE WORD DATA TRANSFER FROM THE
4222 ; LOCATION PRIOR TO THE LOCATION WITH BAD PARITY.
4223 ; MAKE SURE UNIBUS PARITY ERROR DOES NOT OCCUR. REPEAT
4224 ; FOR A ONE WORD DATA TRANSFER FROM THE LOCATION AFTER
4225 ; THE LOCATION WITH BAD PARITY. MAKE SURE UNIBUS PARITY
4226 ; ERROR DOES NOT OCCUR.
4227 ;
4228 ; REPEAT FOR A TWO WORD DATA TRANSFER STARTING WITH
4229 ; THE ADDRESS PRIOR TO THE LOCATION WITH BAD PARITY.
4230 ; MAKE SURE UNIBUS PARITY ERROR DOES OCCUR.
4231 ;
4232 ; NOTE: THIS TEST IS EXECUTED ONLY IF MEMORY PARITY
4233 ; EXISTS FOR SPECIFIED LOCATION AND IS NOT
4234 ; RUN ON AN 11/70.
4235 ;
4236 ;
4237 ;*****
4238 021136 000004 1$T22: SCOPE
4239 021140 012737 000144 001200  MOV      #100,$TIMES ;DO 100. ITERATIONS
4240 021146 005737 003636      TST      PARPR# ;CHECK IF MEMORY PARITY AVAILABLE
4241 021152 001017      BNE      1$      ;YES, DO TEST
4242 021154 012737 000001 001200  MOV      #1,$TIMES ;FORCE INTERATION COUNT TO 1
4243 021162 005227 177777      INC      #-1 ;ONLY DO ONCE
4244 021166 001007      BNE      64$     ;NO GO TO NEXT TEST
4245 021170 104401 055004      TYPE    TSTBY1 ;TYPE TEST N BYPASSED
4246 021174 013746 001220      MOV      $TESTN,-(SP) ;SAVE $TESTN FOR TYPEOUT
4247 021200 104402      TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4248 021202 104401 055014      TYPE    TSTBY2
4249 021206 000137 022374 64$: JMP      1$T23 ;GO TO NEXT TEST
4250
4251 021212 013702 001270 1$: MOV      $BASE,R2 ;LOAD RK611 BASE
4252 021216 004737 044706      JSR     PC,WRTPAR ;GENERATE BAD PARITY
4253 021222 012737 021230 001110  MOV      #2$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
4254 ; SUBTEST LOOP

```

4255									
4256	021230				25:	MOV	#CCLR,RKCS1(R2) ;CLEAR RK611		
4257	021230	012762	100000	000000		MOV	#DMD,RKMR1(R2)		
4258	021236	012762	000040	000026		MOV	#-1,RKWC(R2) ;LOAD WORD COUNT NEG		
4259	021244	012762	177777	000002		CLR	E.WC		
4260	021252	005037	003502			MOV	#BADPAR,RKBA(R2);LOAD BUS ADDRESS		
4261	021256	012762	067270	000004		MOV	#BADPAR+2,E.BA		
4262	021264	012737	067272	003504		MOV	#CERR!RDY!WRHEADS<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1		
4263	021272	012737	100226	003500		MOV	#WRHEAD,RKCS1(R2) ;LOAD COMMAND		
4264	021300	012762	000027	000000		MOV	#50,#4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL		
4265	021306	012700	000312			MOV	#DMD!MCLK,RKMR1(R2) ; INDEX PULSE		
4266	021312	012762	000440	000026	55:	MOV	#DMD,RKMR1(R2)		
4267	021320	012762	000040	000026		DEC	RO		
4268	021326	005300				BNE	55		
4269	021330	001370				MOV	#4,RO ;SIMULATE INDEX PULSE		
4270	021332	012700	000004			MOV	#DMD!MIND,RKMR1(R2)		
4271	021336	012762	000240	000026		MOV	#DMD!MIND!MCLK,RKMR1(R2)		
4272	021344	012762	000640	000026	65:	MOV	#DMD!MIND,RKMR1(R2)		
4273	021352	012762	000240	000025		DEC	RO		
4274	021360	005300				BNE	65		
4275	021362	001370				MOV	#DMD,RKMR1(R2)		
4276	021364	012762	000040	000026		MOV	#37,RO ;ISSUE 1 NPR TRANSFER		
4277	021372	012700	000045			MOV	#DMD!MCLK,RKMR1(R2)		
4278	021376	012762	000440	000026	75:	MOV	#DMD,RKMR1(R2)		
4279	021404	012762	000040	000026		DEC	RO		
4280	021412	005300				BNE	75		
4281	021414	001370				MOV	RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG.1		
4282	021416	016237	000000	003440		MOV	RKBA(R2),↑.BA ;STORE BUS ADDRESS		
4283	021424	016237	000004	003444		MOV	RKWC(R2),↑.WC ;STORE WORD COUNT		
4284	021432	016237	000002	003442		MOV	RKER(R2),↑.ER ;STORE ERROR REG		
4285	021440	016237	000014	003454		MOV	#20.,RO ;WAIT FOR OUTPUT READY		
4286	021446	012700	000024			DEC	RO		
4287	021452	005300			85:	BNE	85		
4288	021454	001376				MOV	RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG.2		
4289	021456	016237	000010	003450		MOV	#IR!OR!UPÉ,E.CS2 ;LOAD EXPECTED CS2		
4290	021464	012737	020300	003510		CLR	E.ER ;LOAD EXPECTED ERROR REG		
4291	021472	005037	003514			CMP	E.CS1,T.CS1 ;CHECK CS1 CORRECT		
4292	021476	023737	003500	003440		BEQ	.05 ;YES CHECK CS2		
4293	021504	001401				ERROR	.06 ;CS1 INCORRECT		
4294	021506	104106				CMP	E.CS2,T.CS2 ;CHECK CS2 CORRECT		
4295	021510	023737	003510	003450	105:	BEQ	.115 ;YES, CHECK BUS ADDRESS		
4296	021516	001401				ERROR	.107 ;CS2 INCORRECT		
4297	021520	104107				CMP	E.BA,T.BA ;CHECK BUS ADDRESS CORRECT		
4298	021522	023737	003504	003444	115:	BEQ	.125 ;YES, CHECK WORD COUNT		
4299	021530	001401				ERROR	.110 ;BUS ADDRESS INCORRECT		
4300	021532	104110				CMP	E.WC,T.WC ;CHECK WORD COUNT CORRECT		
4301	021534	023737	003502	003442	125:	BEQ	.135 ;YES,CHECK ERROR REG.		
4302	021542	001401				ERROR	.111 ;WORD COUNT INCORRECT		
4303	021544	104111				CMP	E.ER,T.ER ;CHECK ERROR REG CORRECT		
4304	021546	023737	003514	003454	135:	BEQ	.145 ;YES,CLEAR ERROR		
4305	021554	001401				ERROR	.116 ;ERROR REG CORRECT		
4306	021556	104116				MOV	#CCLR,RKCS1(R2) ;CLEAR RK611		
4307	021560	012762	100000	000000	145:	MOV	RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1		
4308	021566	016237	000000	003440		MOV	RKCS2(R2),T.CS2 ;STORE COMMAND STATUS REG. 2		
4309	021574	016237	000010	003450		MOV	#RDY,E.CS1 ;LOAD EXPECTED CS1		
4310	021602	012737	000200	003500		MOV			

H07

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 84
 DZR6CB.P11 28-JAN-77 14:04 T22 UNIBUS PARITY ERROR

SEQ 0084

4311	021610	012737	000100	003510		MOV	#IR, E.CS2	: LOAD EXPECTED CS2
4312	021616	023737	003500	003440		CMP	E.CS1, T.CS1	: CHECK CS1 CORRECT
4313	021624	001401				BEQ	15\$: YES, CHECK CS2
4314	021626	104117				ERROR	117	: CS1 INCORRECT
4315	021630	023737	003510	003450	15\$:	CMP	E.CS2, T.CS2	: CHECK IF CS2 CORRECT
4316	021636	001401				BEQ	16\$: YES, CONTINUE
4317	021640	104120				ERROR	120	: CS2 INCORRECT
4318	021642	104415			16\$:	SCOPI		: CHECK IF LOOP ON ERROR
4319	021644	012737	021652	001110		MOV	#20\$, \$LPERR	: LOAD LOOP ON ERROR LOCATION FOR
4320								: SUBTEST LOOP
4321								
4322	021652				20\$:			
4323	021652	012762	100000	000000		MOV	#CLR, RKCS1(R2)	: CLEAR RK611
4324	021660	012762	000040	000026		MOV	#DMD, RKMR1(R2)	: PUT RK611-IN-DIAGNOSTIC MODE
4325	021666	012762	177776	000002		MOV	#-2, RKWC(R2)	: LOAD WORD COUNT REG
4326	021674	012737	177777	003502		MOV	#-1, E.WC	
4327	021702	012762	067266	000004		MOV	#BAOPAR-2, RKBA(R2)	: LOAD BUS ADDRESS REG
4328	021710	012737	067270	003504		MOV	#BAOPAR, E.BA	
4329	021716	012737	000027	003500		MOV	#IRHEAD, E.CS1	: LOAD EXPECTED CS1
4330	021724	012762	000027	000000		MOV	#IRHEAD, RKCS1(R2)	: LOAD COMMAND
4331	021732	012700	000312			MOV	#50, #4+2, RO	: ISSUE ENOUGH CLOCKS UNTIL
4332	021736	012762	000440	000026	21\$:	MOV	#DMD!MCLK, RKMR1(R2)	: INDEX PULSE
4333	021744	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4334	021752	005300				DEC	RO	
4335	021754	001370				BNE	21\$	
4336	021756	012700	000004			MOV	#4, RO	: SIMULATE INDEX PULSE
4337	021762	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
4338	021770	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
4339	021776	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
4340	022004	005300				DEC	RO	
4341	022006	001370				BNE	22\$	
4342	022010	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4343	022016	012700	000045			MOV	#37, RO	: ISSUE 1 NPR TRANSFER
4344	022022	012762	000440	000026	23\$:	MOV	#DMD!MCLK, RKMR1(R2)	
4345	022030	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4346	022036	005300				DEC	RO	
4347	022040	001370				BNE	23\$	
4348	022042	016237	000000	003440		MOV	RKCS1(R2), T.CS1	: STORE COMMAND AND STATUS REG. 1
4349	022050	016237	000004	003444		MOV	RKBA(R2), T.BA	: STORE BUS ADDRESS
4350	022056	016237	000002	003442		MOV	RKWC(R2), T.WC	: STORE WORD COUNT
4351	022064	012700	000024			MOV	#20., RO	: WAIT FOR OUTPUT READY
4352	022070	005300			24\$:	DEC	RO	
4353	022072	001376				BNE	24\$	
4354	022074	016237	000010	003450		MOV	RKCS2(R2), T.CS2	: STORE COMMAND AND STATUS REG. 2
4355	022102	012737	000300	003510		MOV	#IR!OR, E.CS2	: LOAD EXPECTED CS2
4356	022110	023737	003500	003440		CMP	E.CS1, T.CS1	: CHECK CS1 CORRECT
4357	022116	001401				BEQ	25\$: YES, CHECK CS2
4358	022120	104112				ERROR	112	: CS1 INCORRECT
4359	022122	023737	003510	003450	25\$:	CMP	E.CS2, T.CS2	: CHECK CS2 CORRECT
4360	022130	001401				BEQ	26\$: YES, CHECK BUS ADDRESS
4361	022132	104113				ERROR	113	: CS2 INCORRECT
4362	022134	023737	003504	003444	26\$:	CMP	E.BA, T.BA	: CHECK BUS ADDRESS CORRECT
4363	022142	001401				BEQ	27\$: YES, CHECK WORD COUNT
4364	022144	104114				ERROR	114	: BUS ADDRESS INCORRECT
4365	022146	023737	003502	003442	27\$:	CMP	E.WC, T.WC	: CHECK WORD COUNT CORRECT
4366	022154	001401				BEQ	28\$: YES, DO NEXT NPR TRANSFER

4367	022156	104115				ERROR	115		;WORD COUNT INCORRECT
4368	022160	012700	000045		28\$:	MOV	#37, R0		;ISSUE 1 NPR TRANSFER
4369	022164	012762	000440	000026	30\$:	MOV	#DMD, MCLK, RKMR1(R2)		
4370	022172	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
4371	022200	005300				DEC	R0		
4372	022202	001370				BNE	30\$		
4373	022204	016237	000000	003440		MOV	RKCS1(R2), T.CS1		;STORE COMMAND AND STATUS REG 1
4374	022212	016237	000010	003450		MOV	RKCS2(R2), T.CS2		;STORE COMMAND AND STATUS REG 2
4375	022220	016237	000004	003444		MOV	RKBA(R2), T.BA		;STORE BUS ADDRESS
4376	022226	016237	000002	003442		MOV	RKWC(R2), T.WC		;STORE WORD COUNT
4377	022234	016237	000014	003454		MOV	RKER(R2), T.ER		;STORE ERROR REG
4378	022242	012737	100226	003500		MOV	#CERR!RDY!WRHEAD<↑C<GO>>, E.CS1		;LOAD EXPECTED CS1
4379	022250	012737	020300	003510		MOV	#OR!IR!UPE, E.CS2		;LOAD EXPECTED CS2
4380	022256	005237	003502			INC	E.WC		;LOAD EXPECTED WORD COUNT
4381	022262	062737	000002	003504		ADD	#2, E.BA		;LOAD EXPECTED BUS ADDRESS
4382	022270	005037	003514			CLR	E.ER		;LOAD EXPECTED WORD COUNT
4383	022274	023737	003500	003440		CMP	E.CS1, T.CS1		;CHECK CS1 CORRECT
4384	022302	001401				BEQ	31\$;YES, CHECK CS2
4385	022304	104106				ERROR	106		;CS1 INCORRECT
4386	022306	023737	003510	003450	31\$:	CMP	E.CS2, T.CS2		;CHECK CS2 CORRECT
4387	022314	001401				BEQ	32\$;YES, CHECK BUS ADDRESS
4388	022316	104107				ERROR	107		;CS2 INCORRECT
4389	022320	023737	003504	003444	32\$:	CMP	E.BA, T.BA		;CHECK BUS ADDRESS CORRECT
4390	022326	001401				BEQ	33\$;YES, CHECK WORD COUNT
4391	022330	104110				ERROR	110		;BUS ADDRESS INCORRECT
4392	022332	023737	003502	003442	33\$:	CMP	E.WC, T.WC		;CHECK WORD COUNT CORRECT
4393	022340	001401				BEQ	34\$;YES, CHECK ERROR REG
4394	022342	104111				ERROR	111		;WORD COUNT INCORRECT
4395	022344	023737	003514	003454	34\$:	CMP	E.ER, T.ER		;CHECK ERROR REG CORRECT
4396	022352	001401				BEQ	35\$;YES, CONTINUE
4397	022354	104116				ERROR	116		;WORD COUNT INCORRECT
4398	022356	104415			35\$:	SCOPI			;CHECK IF WORD ON ERROR
4399	022360	012762	100000	000000		MOV	#CLR, RKCS1(R2)		;CLEAR RK611
4400	022366	012737	000000	067270		MOV	#0, BAOPAR		;WRITE GOOD PARITY

 *TEST 23 SILO FILL IN 18 BIT MODE

* CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER
 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
 * IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0,
 * SPECIFY 66 WORD DATA TRANSFER. CLOCK
 * ALL 66 WORD INTO THE SILO. CHECK THAT ALL 66
 * WORDS ARE CORRECT (16 LEAST SIGNIFICANT BITS).

4401									
4402									
4403									
4404									
4405									
4406									
4407									
4408									
4409									
4410									
4411									
4412									
4413	022374	000004			†ST23:	SCOPE			
4414	022376	012737	000144	001200		MOV	#100, STIMES		;DO 100. ITERATIONS
4415	022404	013702	001270			MOV	#BASE, R2		;LOAD RK611 BASE
4416	022410	012762	100000	000000		MOV	#CLR, RKCS1(R2)		;CLEAR RK611
4417	022416	012762	000040	000026		MOV	#DMD, RKMR1(R2)		;PUT RK611 IN MAINTENANCE MODE
4418	022424	012762	177676	000002		MOV	#-66, RKWC(R2)		;LOAD WORD COUNT
4419	022432	012737	177677	003502		MOV	#-65, E.WC		
4420	022440	012762	067032	000004		MOV	#NPRBUF, RKBA(R2)		;LOAD BUS ADDRESS
4421	022446	012737	067034	003504		MOV	#NPRBUF+2, E.BA		
4422	022454	012737	010027	003500		MOV	#WRHEAD!CFMT, E.CS1		;LOAD EXPECTED CS1

#123	022462	012762	010027	000000		MOV	#WRHEAD:CFMT,RKCS1(R2)	:ISSUE COMMAND
#124	022470	012700	000312			MOV	#50.*4+2,RO	:ISSUE ENOUGH CLOCKS DATA
#125								:INDEX PULSE
#126	022474	012762	000440	000026	5\$:	MOV	#DMD:MCLK,RKMR1(R2)	
#127	022502	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
#128	022510	005300				DEC	RO	
#129	022512	001370				BNE	5\$	
#130	022514	012700	000004			MOV	#4,RO	:SIMULATE INDEX PULSE
#131	022520	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
#132	022526	012762	000640	000026	6\$:	MOV	#DMD:MIND:MCLK,RKMR1(R2)	
#133	022534	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
#134	022542	005300				DEC	RO	
#135	022544	001370				BNE	6\$	
#136	022546	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
#137	022554	012737	000300	003510		MOV	#IR:OR,E.CS2	:LOAD EXPECTED CS2
#138	022562	012701	000102			MOV	#66.,R1	:ISSUE 66 NPR TRANSFERS
#139	022566	012700	000050		7\$:	MOV	#40.,RO	
#140	022572	012762	000440	000026	8\$:	MOV	#DMD:MCLK,RKMR1(R2)	
#141	022600	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
#142	022606	005300				DEC	RO	
#143	022610	001370				BNE	8\$	
#144	022612	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
#145	022620	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG. 2
#146	022626	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
#147	022634	022737	067034	003504		CMP	#NPRBUF+2,E.BA	:CHECK IF FIRST WORD
#148	022642	001004				BNE	10\$:NO,CONTINUE
#149	022644	012700	000024			MOV	#20.,RO	:WAIT FOR OUTPUT READY
#150	022650	005300			9\$:	DEC	RO	
#151	022652	001376				BNE	9\$	
#152	022654	016237	000010	003450	10\$:	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
#153	022662	005737	003502			TST	E.WC	:CHECK IF LAST WORD
#154	022666	001003				BNE	11\$:NO,CONTINUE
#155	022670	012737	000200	003510		MOV	#OR,E.CS2	:LOAD EXPECTED CS2
#156	022676	023737	003500	003440	11\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
#157	022704	001401				BEQ	12\$:YES,CONTINUE
#158	022706	104121				ERROR	121	:CS1 INCORRECT
#159	022710	023737	003510	003450	12\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
#160	022716	001401				BEQ	13\$:YES,CONTINUE
#161	022720	104122				ERROR	122	:CS2 INCORRECT
#162	022722	023737	003504	003444	13\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
#163	022730	001401				BEQ	14\$:YES,CONTINUE
#164	022732	104123				ERROR	123	:BUS ADDRESS INCORRECT
#165	022734	023737	003502	003442	14\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT REG CORRECT
#166	022742	001401				BEQ	15\$:YES,CONTINUE
#167	022744	104124				ERROR	124	:WORD COUNT INCORRECT
#168	022746	062737	000002	003504	15\$:	ADD	#2,E.BA	:INCREMENT BUS ADDRESS AND
#169	022754	005237	003502			INC	E.WC	:WORD COUNT
#170	022760	005301				DEC	R1	:CHECK IF SILO FULL
#171	022762	001301				BNE	7\$:NO,GET NEXT WORD
#172	022764	012700	000120			MOV	#2*40.,RO	:ISSUE CLOCKS FOR TWO NPR'S
#173	022770	012762	000440	000026	20\$:	MOV	#DMD:MCLK,RKMR1(R2)	
#174	022776	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
#175	023004	005300				DEC	RO	
#176	023006	001370				BNE	20\$	
#177	023010	162737	000002	003504		SUB	#2,E.BA	:ADJUST BUS ADDRESS AND WORD COUNT
#178	023016	005037	003502			CLR	E.WC	

K07

```

4479 023022 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4480 023030 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4481 023036 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
4482 023044 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
4483 023052 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4484 023056 001401 BEQ 21$ ;YES, CONTINUE
4485 023062 104125 ERROR 125 ;CS1 INCORRECT
4486 023064 023737 003510 003450 21$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4487 023072 001401 BEQ 22$ ;YES, CONTINUE
4488 023074 104126 ERROR 126 ;CS2 INCORRECT
4489 023076 023737 003504 003444 22$: CMP E.BA,T.BA ;CHECK BUS ADDRESS INCORRECT
4490 023104 001401 BEQ 23$ ;YES, CONTINUE
4491 023106 104127 ERROR 127 ;BUS ADDRESS INCORRECT
4492 023110 023737 003502 003442 23$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
4493 023116 001401 BEQ 24$ ;YES, CONTINUE
4494 023120 104130 ERROR 130 ;WORD COUNT INCORRECT
4495 023122 012703 067032 24$: MOV #NPRBUF,R3 ;LOAD BUFFER ADDRESS FOR COMPARE
4496 023126 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4497 023134 012701 000102 MOV #66.,R1 ;LOAD COUNT
4498 023140 005037 003624 CLR WRDCNT ;INITIALIZE WORD COUNT
4499 023144 016237 000024 003462 25$: MOV RKDB(R2),T.DB ;GET DATA BUFFER
4500 023152 012337 003522 MOV (R3)+,E.DB ;GET EXPECTED DATA
4501 023156 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4502 023164 001401 BEQ 26$ ;YES, CONTINUE
4503 023166 104131 ERROR 131 ;DATA READ INCORRECT
4504 023170 012700 000050 26$: MOV #40.,R0 ;SET STALL
4505 023174 005300 27$: DEC R0 ;RUN STALL TO ZERO
4506 023176 001376 BNE 27$
4507 023200 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
4508 023206 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
4509 023214 022701 000001 CMP #1,R1 ;CHECK IF LAST WORD
4510 023220 001003 BNE 28$ ;NO, CONTINUE
4511 023222 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4512 023230 023737 003500 003440 28$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4513 023236 001401 BEQ 29$ ;YES, CONTINUE
4514 023240 104132 ERROR 132 ;CS1 INCORRECT
4515 023242 023737 003510 003450 29$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4516 023250 001401 BEQ 30$ ;YES, CONTINUE
4517 023252 104133 ERROR 133 ;CS2 INCORRECT
4518 023254 005237 003624 30$: INC WRDCNT ;INCREMENT WORD COUNT
4519 023260 005301 DEC R1 ;CHECK IF FINISHED
4520 023262 001330 BNE 25$ ;NO, CONTINUE
    
```

```

*****
*TEST 24 BIT 16 AND 17 READING WITH NPR
*
* CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 24 SECTOR FORMAT. CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY ONE WORD DATA TRANSFER FROM A LOCATION
* WITH BAD PARITY. MAKE SURE A UNIBUS PARITY
* DOES NOT OCCUR.
*
* NOTE: THIS TEST IS EXECUTED ONLY IF MEMORY PARITY
* ENABLE EXISTS FOR SPECIFIED LOCATION AND IS
* NOT RUN ON AN 11/70.
    
```

4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534

```

4535
4536
4537
4538 023264 000004
4539 023266 012737 000144 001200
4540 023274 005737 003636
4541 023300 001017
4542 023302 012737 000001 001200
4543 023310 005227 177777
4544 023314 001007
4545 023316 104401 055004
4546 023322 013746 001220
4547 023326 104402
4548 023330 104401 055014
4549 023334 000137 024110
4550
4551 023340 013702 001270
4552 023344 004737 044706
4553 023350 012762 100000 000000
4554 023356 012762 000040 000026
4555 023364 012762 177777 000002
4556 023372 005037 003502
4557 023376 012762 067270 000004
4558 023404 012737 067272 003504
4559 023412 012737 010027 003500
4560 023420 012762 010027 000000
4561 023426 012700 000312
4562
4563 023432 012762 000440 000026
4564 023440 012762 000040 000026
4565 023446 005300
4566 023450 001370
4567 023452 012700 000004
4568 023456 012762 000240 000026
4569 023464 012762 000640 000026
4570 023472 012762 000240 000026
4571 023500 005300
4572 023502 001370
4573 023504 012762 000040 000026
4574 023512 012737 000300 003510
4575 023520 012700 000050
4576 023524 012762 000440 000026
4577 023532 012762 000040 000026
4578 023540 005300
4579 023542 001370
4580 023544 016237 000000 003440
4581 023552 016237 000004 003444
4582 023560 016237 000002 003442
4583 023566 012700 000024
4584 023572 005300
4585 023574 001376
4586 023576 016237 000010 003450
4587 023604 023737 003500 003440
4588 023612 001401
4589 023614 104134
4590 023616 023737 003510 003450

```

;*
 ;*****
 †ST24: SCOPE
 MOV #100, \$TIMES ;DO 100. ITERATIONS
 PARPR# ;CHECK IF MEMORY PARITY AVAILABLE
 BNE IS ;YES, DO TEST
 MOV #1, \$TIMES ;FORCE INTERATION COUNT TO 1
 INC #-1 ;ONLY DO ONCE
 BNE 64\$;NO, GO TO NEXT TEST
 TYPE TSTBY1 ;TYPE TEST N BYPASSED
 MOV \$TESTN, -(SP) ;SAVE \$TESTN FOR TYPEOUT
 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
 TYPE TSTBY2
 64\$: JMP †ST25 ;GO TO NEXT TEST
 IS: MOV \$BASE, R2 ;LOAD RK611 BASE
 JSR PC, WRTPAR ;GENERATE BAD PARITY
 MC / #CLR, RKCS1(R2) ;CLEAR RK611
 MOV #DMD, RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
 MOV #-1, RKWC(R2) ;LOAD WORD COUNT
 CLR E, WC
 MOV #BADPAR, RKBA(R2) ;LOAD BUS ADDRESS
 MOV #BADPAR+2, E, BA
 MOV #IRHEAD:CFMT, E, CS1 ;LOAD EXPECTED CS1
 MOV #IRHEAD:CFMT, RKCS1(R2) ;ISSUE COMMAND
 MOV #50, *4+2, RO ;ISSUE ENOUGH CLOCKS UNTIL
 ; INDEX PULSE
 5\$: MOV #DMD:MCLK, RKMR1(R2)
 MOV #DMD, RKMR1(R2)
 DEC RO
 BNE 5\$
 MOV #4, RO ;SIMULATE INDEX PULSE
 MOV #DMD:MIND, RKMR1(R2)
 6\$: MOV #DMD:MIND:MCLK, RKMR1(R2)
 MOV #DMD:MIND, RKMR1(R2)
 DEC RO
 BNE 6\$
 MOV #DMD, RKMR1(R2)
 MOV #IR:OR, E, CS2 ;LOAD EXPECTED CS2
 MOV #40, RO ;ISSUE NPR TRANSFER
 8\$: MOV #DMD:MCLK, RKMR1(R2)
 MOV #DMD, RKMR1(R2)
 DEC RO
 BNE 8\$
 MOV RKCS1(R2), T, CS1 ;STORE COMMAND AND STATUS REG. 1
 MOV RKBA(R2), †, BA ;STORE BUS ADDRESS
 MOV RKWC(R2), T, WC ;STORE WORD COUNT
 MOV #20, RO ;WAIT FOR OUTPUT READY
 9\$: DEC RO
 BNE 9\$
 MOV RKCS2(R2), T, CS2 ;STORE COMMAND AND STATUS REG. 2
 CMP E, CS1, T, CS1 ;CHECK CS1 CORRECT
 BEQ 12\$;YES, CHECK CS2
 ERROR 134 ;CS1 INCORRECT
 12\$: CMP E, CS2, T, CS2 ;CHECK CS2 CORRECT

MO7

4591	023624	001401				BEQ	13\$: YES, CHECK BUS ADDRESS
4592	023626	104135				ERROR	135		: CS2 INCORRECT
4593	023630	023737	003504	003444	13\$:	CMP	E.BA, T.BA		: CHECK BUS ADDRESS CORRECT
4594	023636	001401				BEQ	14\$: YES, CHECK WORD COUNT
4595	023640	104136				ERROR	136		: BUS ADDRESS INCORRECT
4596	023642	023737	003502	003442	14\$:	CMP	E.WC, T.WC		: CHECK WORD COUNT CORRECT
4597	023650	001401				BEQ	15\$: YES, CONTINUE
4598	023652	104137				ERROR	137		: WORD COUNT INCORRECT
4599	023654	01270C	000120		15\$:	MOV	#2*40, R0		: ISSUE CLOCKS FOR TWO NPR'S
4600	023660	012762	000440	000026	20\$:	MOV	#DMD, MCLK, RKMR1(R2)		
4601	023666	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
4602	023674	005300				DEC	R0		
4603	023676	001370				BNE	20\$		
4604	023700	016237	000000	003440		MOV	RKCS1(R2), T.CS1		: STORE COMMAND STATUS REG. 1
4605	023706	016237	000010	003450		MOV	RKCS2(R2), T.CS2		: STORE COMMAND AND STATUS REG. 2
4606	023714	016237	000004	003444		MOV	RKBA(R2), T.BA		: STORE BUS ADDRESS REG.
4607	023722	016237	000002	003442		MOV	RKWC(R2), T.WC		: STORE WORD COUNT
4608	023730	023737	003500	003440		CMP	E.CS1, T.CS1		: CHECK CS1 CORRECT
4609	023736	001401				BEQ	21\$: YES, CHECK CS2
4610	023740	104140				ERROR	140		: CS1 INCORRECT
4611	023742	023737	003510	003450	21\$:	CMP	E.CS2, T.CS2		: CHECK CS2 CORRECT
4612	023750	001401				BEQ	22\$: YES, CHECK BUS ADDRESS
4613	023752	104141				ERROR	141		: CS2 INCORRECT
4614	023754	023737	003504	003444	22\$:	CMP	E.BA, T.BA		: CHECK BUS ADDRESS CORRECT
4615	023762	001401				BEQ	23\$: YES, CHECK WORD COUNT
4616	023764	104142				ERROR	142		: BUS ADDRESS INCORRECT
4617	023766	023737	003502	003442	23\$:	CMP	E.WC, T.WC		: CHECK WORD COUNT REG. CORRECT
4618	023774	001401				BEQ	24\$: YES, CHECK DATA
4619	023776	104143				ERROR	143		: WORD COUNT INCORRECT
4620	024000	016237	000024	003462	24\$:	MOV	RKDB(R2), T.DB		: READ DATA BUFFER
4621	024006	012737	000157	003522		MOV	#157, E.DB		: LOAD EXPECT DATA
4622	024014	023737	003522	003462		CMP	E.DB, T.DB		: CHECK TO MAKE SURE CORRECT
4623	024022	001401				BEQ	26\$: YES, CONTINUE
4624	024024	104144				ERROR	144		: DATA INCORRECT
4625	024026	016237	000000	003440	26\$:	MOV	RKCS1(R2), T.CS1		: STORE COMMAND AND STATUS REG 1
4626	024034	016237	000010	003450		MOV	RKCS2(R2), T.CS2		: STORE COMMAND AND STATUS REG 2
4627	024042	012737	000100	003510		MOV	#IR, E.CS2		: LOAD EXPECTED CS2
4628	024050	023737	003500	003440		CMP	E.CS1, T.CS1		: CHECK CS1 CORRECT
4629	024056	001401				BEQ	29\$: YES, CHECK SC2
4630	024060	104145				ERROR	145		: CS1 INCORRECT
4631	024062	023737	003510	003450	29\$:	CMP	E.CS2, T.CS2		: CHECK CS1 CORRECT
4632	024070	001401				BEQ	30\$: YES, CONTINUE
4633	024072	104146				ERROR	146		: CS2 INCORRECT
4634	024074	012737	000000	067270	30\$:	MOV	#0, BADPAR		: WRITE GOOD PARITY
4635	024102	012762	100000	000000		MOV	#CCLR, RKCS1(R2)		: CLEAR RK611

.SBTTL **MFM READ LOOPBACK TESTS

```

:*****
:TEST 25      READ LOOPBACK (PART 1)
:
:
:   CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
:   IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN RK06
:   IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
:   CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
:   SIMULATE SECTOR PULSE, 255 ZEROES, A

```

4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646

```

4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658 024110 000004
4659 024112 012737 000144 001200
4660 024120 013702 001270
4661 024124 012762 100000 000000
4662 024132 012762 000040 000026
4663 024140 012762 000025 000000
4664 024146 012700 000312
4665 024152 012762 000440 000026 1$:
4666 024160 012762 000040 000026
4667 024166 005300
4668 024170 001370
4669 024172 012762 000140 000026
4670 024200 012762 000040 000026
4671 024206 005037 003614
4672 024212 005037 003616
4673 024216 012700 000341
4674 024222 004737 046156 2$:
4675 024226 005300
4676 024230 001374
4677 024232 012737 000001 003614
4678 024240 004737 046156
4679 024244 012701 000003
4680 024250 012703 066752
4681 024254 012737 000025 003500
4682 024262 012304 5$:
4683 024264 012700 000020
4684 024270 013737 003614 6$:
4685 024276 006004
4686 024300 103403
4687 024302 005037 003614
4688 024306 000403
4689
4690 024310 012737 000001 003614 7$:
4691 024316 004737 046156 8$:
4692 024322 016237 000000 003440
4693 024330 023737 003500 003440
4694 024336 001417
4695 024340 012737 000003 003624
4696 024346 160137 003624
4697 024352 012737 000020 003622
4698 024360 160037 003622
4699 024364 104150
4700 024366 012762 100000 000000
4701 024374 000522
4702

```

* ONE, AND A HEADER CONSISTING OF THE THREE
* FOLLOWING WORDS:

* 177777
* 000000
* 177777

* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

```

TST25: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD, RKCS1(R2) ; ISSUE READ HEADER
MOV #50, #4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK, RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225, R0
2$: JSR PC, R0BIT ; SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1, PR.BIT
JSR PC, R0BIT
MOV #3, R1 ; LOAD NUMBER OF WORDS
MOV #HEAD1, R3 ; LOAD ADDRESS OF DATA
MOV #RDHEAD, E.CS1 ; LOAD EXPECTED CS1
5$: MOV (R3)+, R4 ; GET DATA
MOV #16, R0 ; LOAD BIT COUNT
6$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 7$ ; CHECK IF 1
CLR PR.BIT ; NO, ZERO
BR 8$ ; SIMULATE READ DATA
7$: MOV #1, PR.BIT ; ONE
8$: JSR PC, R0BIT ; SIMULATE READ DATA
MOV RKCS1(R2), T.CS1 ; READ COMMAND AND STATUS REG. 1
CMP E.CS1, T.CS1 ; CHECK IF CS1 CORRECT
BEQ 9$ ; YES, SIMULATE NEXT BIT
MOV #3, WRDCNT ; LOAD WORD COUNT
SUB R1, WRDCNT
MOV #16, BITCNT ; LOAD BIT COUNT
SUB R0, BITCNT
ERROR 150 ; CS1 INCORRECT DURING HEADER
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
BR TST26 ; GO ON TO NEXT TEST

```

```

4703 024376 005300          9$: DEC      R0          ;CHECK IF READY FOR NEXT WORD
4704 024400 001333          BNE      6$          ;NO, GET NEXT BIT
4705 024402 005301          DEC      R1          ;CHECK IF HEADER FINISHED
4706 024404 001326          BNE      5$          ;NO, GET NEXT WORD
4707 024406 012700 000004      MOV      #4,R0       ;LOAD COUNT FOR POSTAMBLE
4708 024412 013737 003614 003616 15$: MOV      PR.BIT,M1.BIT ;STORE LAST BIT
4709 024420 005037 003614      CLR      PR.BIT     ;LOAD NEXT BIT
4710 024424 004737 046156      JSR      PC,ROBIT   ;SIMULATE 1 BIT READ
4711 024430 005300          DEC      R0          ;CHECK IF TIME FOR READY
4712 024432 001367          BNE      15$        ;NO, CONTINUE WITH POSTAMBLE
4713 024434 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
4714 024442 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
4715 024450 012737 000224 003500      MOV      #RDY!RDHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4716 024456 012737 000300 003510      MOV      #OR!IR,E.CS2  ;LOAD EXPECTED CS2
4717 024464 023737 003500 003440      CMP      E.CS1,↑.CS1  ;CHECK CS1 CORRECT
4718 024472 001401          BEQ      16$        ;YES, CHECK CS2
4719 024474 104151          ERROR   151        ;CS1 INCORRECT
4720 024476 023737 00.~10 003450 16$: CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
4721 024504 001401          BEQ      17$        ;YES, CHECK DATA
4722 024506 104152          ERROR   152        ;CS2 INCORRECT
4723 024510 005037 003624          CLR      WRDCNT     ;INITIALIZE WORD COUNT
4724 024514 012703 066752          MOV      #HEAD1,R3  ;GET ADDRESS OF DATA
4725 024520 012337 003522          MOV      (R3)+,E.DB  ;GET EXPECTED DATA
4726 024524 016237 000024 003462      MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
4727 024532 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4728 024540 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4729 024546 022737 000002 003624      CMP      #2,WRDCNT  ;CHECK IF LAST WORD IN DATA BUFFER
4730 024554 001003          BNE      21$        ;NO, CHECK CS1
4731 024556 012737 000100 003510      MOV      #IR,E.CS2   ;STORE EXPECTED CS2
4732 024564 023737 003500 003440 21$: CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
4733 024572 001402          BEQ      22$        ;YES, CHECK CS2
4734 024574 104153          ERROR   153        ;CS1 INCORRECT
4735 024576 000421          BR      TST26      ;GO ON TO NEXT TEST
4736
4737 024600 023737 003510 003450 22$: CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
4738 024606 001402          BEQ      23$        ;YES, CHECK DATA
4739 024610 104154          ERROR   154        ;CS2 INCORRECT
4740 024612 000413          BR      TST26      ;GO ON TO NEXT TEST
4741
4742 024614 023737 003522 003462 23$: CMP      E.DB,T.DB   ;CHECK IF DATA CORRECT
4743 024622 001401          BEQ      24$        ;YES, GET NEXT HEADER WORD
4744 024624 104155          ERROR   155        ;DATA INCORRECT
4745 024626 005237 003624          INC      WRDCNT     ;INCREMENT WORD COUNT
4746 024632 022737 000003 003624 24$: CMP      #3,WRDCNT  ;CHECK IF ALL THREE WORDS CHECK
4747 024640 001327          BNE      20$        ;NO, GET NEXT WORD

```

```

4748
4749 *****
4750 *TEST 26      READ LOOPBACK (PART 2)
4751 *
4752 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4753 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
4754 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4755 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
4756 * SIMULATE SECTOR PULSE, 255 ZEROES, A ONE,
4757 * AND A HEADER CONSISTING OF THE THREE
4758 * FOLLOWING WORDS:

```

```

4759      *
4760      *          000000
4761      *          177777
4762      *          000000
4763      *
4764      *          MAKE SURE THAT READY COMES UP AFTER THIRD WORD
4765      *          IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
4766      *
4767      *
4768      * *****
4768      * TST26: SCOPE
4769      * MOV      #100.,$TIMES      ;; DO 100. ITERATIONS
4770      * MOV      $BASE,R2        ;; LOAD RK611 BASE
4771      * MOV      #CCLR,RKCS1(R2) ;; CLEAR RK611
4772      * MOV      #DMD,RKMR1(R2)  ;; PUT RK611 IN DIAGNOSTIC MODE
4773      * MOV      #RDHEAD,RKCS1(R2) ;; ISSUE READ HEADER
4774      * MOV      #50.*4+2,R0     ;; ISSUE ENOUGH CLOCKS UNTIL READY
4775      * 1$: MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4776      * MOV      #DMD,RKMR1(R2)
4777      * DEC      R0
4778      * BNE     1$
4779      * MOV      #DMD!MSP,RKMR1(R2) ; SIMULATE SECTOR PULSE
4780      * MOV      #DMD,RKMR1(R2)
4781      * CLR     PR.BIT           ; INITIALIZE PRESENT BIT AND
4782      * CLR     M1.BIT          ; PREVIOUS BIT
4783      * MOV      #25.,R0
4784      * 2$: JSR     PC,RDBIT     ; SIMULATE SYNCH
4785      * DEC     R0
4786      * BNE     2$
4787      * MOV     #1,PR.BIT
4788      * JSR     PC,RDBIT
4789      * MOV     #3,R1           ; LOAD NUMBER OF WORDS
4790      * MOV     #HEAD2,R3       ; LOAD ADDRESS OF DATA
4791      * MOV     #RDHEAD,E.CS1   ; LOAD EXPECTED CS1
4792      * 5$: MOV     (R3)+,R4     ; GET DATA
4793      * MOV     #16.,R0        ; LOAD BIT COUNT
4794      * 6$: MOV     PR.BIT,M1.BIT ; STORE PREVIOUS BIT
4795      * ROR     R4             ; GET NEXT BIT
4796      * BCS     7$            ; CHECK IF 1
4797      * CLR     PR.BIT        ; NO, ZERO
4798      * BR     8$            ; SIMULATE READ DATA
4799
4800      * 7$: MOV     #1,PR.BIT   ; ONE
4801      * 8$: JSR     PC,RDBIT   ; SIMULATE READ DATA
4802      * MOV     RKCS1(R2),T.CS1 ; READ COMMAND AND STATUS REG. 1
4803      * CMP     E.CS1,T.CS1   ; CHECK IF CS1 CORRECT
4804      * BEQ     9$           ; YES, SIMULATE NEXT BIT
4805      * MOV     #3,WRDCNT     ; LOAD WORD COUNT
4806      * SUB     R1,WRDCNT
4807      * MOV     #16.,BITCNT   ; LOAD BIT COUNT
4808      * SUB     R0,BITCNT
4809      * ERROR  150          ; CS1 INCORRECT DURING HEADER
4810      * MOV     #CCLR,RKCS1(R2) ; CLEAR RK611
4811      * BR     TST27        ; GO ON TO NEXT TEST
4812
4813      * 9$: DEC     R0         ; CHECK IF READY FOR NEXT WORD
4814      * BNE     6$         ; NO, GET NEXT BIT

```

4815	025134	005301				DEC	R1	:CHECK IF HEADER FINISHED
4816	025136	001326				BNE	5\$:NO, GET NEXT WORD
4817	025140	012700	000004			MOV	#4,R0	:LOAD COUNT FOR POSTAMBLE
4318	025144	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
4819	025152	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
4820	025156	004737	046156			JSR	PC,ROBIT	:SIMULATE 1 BIT READ
4821	025162	005300				DEC	R0	:CHECK IF TIME FOR READY
4822	025164	001367				BNE	15\$:NO, CONTINUE WITH POSTAMBLE
4823	025166	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
4824	025174	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
4825	025202	012737	000224	003500		MOV	#RDY!RDHEAD<↑C<GO>>	:E.CS1:LOAD EXPECTED CS1
4826	025210	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
4827	025216	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4828	025224	001401				BEQ	16\$:YES, CHECK CS2
4829	025226	104151				ERROR	151	:CS1 INCORRECT
4830	025230	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4831	025236	001401				BEQ	17\$:YES, CHECK DATA
4832	025240	104152				ERROR	152	:CS2 INCORRECT
4833	025242	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
4834	025246	012703	066760			MOV	#HEAD2,R3	:GET ADDRESS OF DATA
4835	025252	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
4836	025256	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
4837	025264	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
4838	025272	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
4839	025300	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
4840	025306	001003				BNE	21\$:NO, CHECK CS1
4841	025310	012737	000100	003510		MOV	#IR,E.CS2	:STORE EXPECTED CS2
4842	025316	023737	003500	003440	21\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4843	025324	001402				BEQ	22\$:YES, CHECK CS2
4844	025326	104153				ERROR	153	:CS1 INCORRECT
4845	025330	000421				BR	TST27	:GO ON TO NEXT TEST
4846								
4847	025332	023737	003510	003450	22\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4848	025340	001402				BEQ	23\$:YES, CHECK DATA
4849	025342	104154				ERROR	154	:CS2 INCORRECT
4850	025344	000413				BR	TST27	:GO ON TO NEXT TEST
4851								
4852	025346	023737	003522	003462	23\$:	CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
4853	025354	001401				BEQ	24\$:YES, GET NEXT HEADER WORD
4854	025356	104155				ERROR	155	:DATA INCORRECT
4855	025360	005237	003624		24\$:	INC	WRDCNT	:INCREMENT WORD COUNT
4856	025364	022737	000003	003624		CMP	#3,WRDCNT	:CHECK IF ALL THREE WORDS CHECK
4857	025372	001327				BNE	20\$:NO, GET NEXT WORD

*TEST 27 READ LOOPBACK (PART 3)

*
 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES
 * SIMULATE SECTOR PULSE, 255 ZEROES, A
 * ONE, AND A HEADER CONSISTING OF THE THREE
 * FOLLOWING WORDS:

125252

4858
 4859
 4860
 4861
 4862
 4863
 4864
 4865
 4866
 4867
 4868
 4869
 4870

E08

```

4871          : *          052525
4872          : *          125252
4873          : *
4874          : *          MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
4875          : *          IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
4876          : *
4877          : * *****
4878 025374 000004          *ST27: SCOPE
4879 025376 012737 000144 001200      MOV      #100.,$TIMES      ;;DO 100. ITERATIONS
4880 025404 013702 001270          MOV      $BASE,R2        ;;LOAD RK611 BASE
4881 025410 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
4882 025416 012762 000040 000026      MOV      #0MD,RKMR1(R2) ;;PUT RK611 IN DIAGNOSTIC MODE
4883 025424 012762 000025 000000      MOV      #RDHEAD,RKCS1(R2) ;;ISSUE READ HEADER
4884 025432 012700 000312          MOV      #50.*4+2,R0     ;;ISSUE ENOUGH CLOCKS UNTIL READY
4885 025436 012762 000440 000026 1$:   MOV      #0MD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4886 025444 012762 000040 000026      MOV      #0MD,RKMR1(R2)
4887 025452 005300          DEC      R0
4888 025454 001370          BNE     1$
4889 025456 012762 000140 000026      MOV      #0MD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4890 025464 012762 000040 000026      MOV      #0MD,RKMR1(R2)
4891 025472 005037 003614          CLR     PR.BIT          ;;INITIALIZE PRESENT BIT AND
4892 025476 005037 003616          CLR     M1.BIT          ;; PREVIOUS BIT
4893 025502 012700 000341          MOV      #225.,R0
4894 025506 004737 046156 2$:   JSR     PC,ROBIT        ;;SIMULATE SYNCH
4895 025512 005300          DEC      R0
4896 025514 001374          BNE     2$
4897 025516 012737 000001 003614      MOV      #1,PR.BIT
4898 025524 004737 046156          JSR     PC,ROBIT
4899 025530 012701 000003          MOV      #3,R1          ;;LOAD NUMBER OF WORDS
4900 025534 012703 066766          MOV      #HEAD03,R3     ;;LOAD ADDRESS OF DATA
4901 025540 012737 000025 003500      MOV      #RDHEAD,E.CS1  ;;LOAD EXPECTED CS1
4902 025546 012304          MOV      (R3)+,R4       ;;GET DATA
4903 025550 012700 000020          MOV      #16.,R0       ;;LOAD BIT COUNT
4904 025554 013737 003614 003616 6$:   MOV      PR.BIT,M1.BIT  ;;STORE PREVIOUS BIT
4905 025562 006004          ROR     R4              ;;GET NEXT BIT
4906 025564 103403          BCS     7$              ;;CHECK IF 1
4907 025566 005037 003614          CLR     PR.BIT         ;;NO, ZERO
4908 025572 000403          BR     8$              ;;SIMULATE READ DATA
4909
4910 025574 012737 000001 003614 7$:   MOV      #1,PR.BIT      ;;ONE
4911 025602 004737 046156 8$:   JSR     PC,ROBIT        ;;SIMULATE READ DATA
4912 025606 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;;READ COMMAND AND STATUS REG. 1
4913 025614 023737 003500 003440      CMP     E.CS1,T.CS1    ;;CHECK IF CS1 CORRECT
4914 025622 001417          BEQ     9$              ;;YES, SIMULATE NEXT BIT
4915 025624 012737 000003 003624      MOV      #3,WDCNT      ;;LOAD WORD COUNT
4916 025632 160137 003624          SUB     R1,WDCNT
4917 025636 012737 000020 003622      MOV      #16.,BITCNT   ;;LOAD BIT COUNT
4918 025644 160037 003622          SUB     R0,BITCNT
4919 025650 104150          ERROR  150             ;;CS1 INCORRECT DURING HEADER
4920 025652 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
4921 025660 000522          BR     TST30           ;;GO ON TO NEXT TEST
4922
4923 025662 005300 9$:   DEC      R0              ;;CHECK IF READY FOR NEXT WORD
4924 025664 001333          BNE     6$              ;;NO, GET NEXT BIT
4925 025666 005301          DEC     R1              ;;CHECK IF HEADER FINISHED
4926 025670 001326          BNE     5$              ;;NO, GET NEXT WORD
    
```

```

4927 025672 012700 000004      MOV      #4,R0      ;LOAD COUNT FOR POSTAMBLE
4928 025676 013737 003614 003616 15$:  MOV      PR.BIT,M1.BIT ;STORE LAST BIT
4929 025704 005037 003614      CLR      PR.BIT      ;LOAD NEXT BIT
4930 025710 004737 046156      JSR      PC,ROBIT     ;SIMULATE 1 BIT READ
4931 025714 005300      DEC      R0          ;CHECK IF TIME FOR READY
4932 025716 001367      BNE      15$         ;NO, CONTINUE WITH POSTAMBLE
4933 025720 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
4934 025726 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
4935 025734 012737 000224 003500      MOV      #RDY!R0HEAD<+C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4936 025742 012737 000300 003510      MOV      #OR!IR,E.CS2   ;LOAD EXPECTED CS2
4937 025750 023737 003500 003440      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4938 025756 001401      BEQ      16$         ;YES, CHECK CS2
4939 025760 104151      ERROR   151         ;CS1 INCORRECT
4940 025762 023737 003510 003450 16$:  CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4941 025770 001401      BEQ      17$         ;YES, CHECK DATA
4942 025772 104152      ERROR   152         ;CS2 INCORRECT
4943 025774 005037 003624      CLR      WRDCNT      ;INITIALIZE WORD COUNT
4944 026000 012703 066766      MOV      #HEAD3,R3    ;GET ADDRESS OF DATA
4945 026004 012337 003522      MOV      (R3)+,E.DB   ;GET EXPECTED DATA
4946 026010 016237 000024 003462      MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
4947 026016 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4948 026024 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4949 026032 022737 000002 003624      CMP      #2,WRDCNT    ;CHECK IF LAST WORD IN DATA BUFFER
4950 026040 001003      BNE      21$         ;NO, CHECK CS1
4951 026042 012737 000100 003510      MOV      #IR,E.CS2    ;STORE EXPECTED CS2
4952 026050 023737 003500 003440 21$:  CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4953 026056 001402      BEQ      22$         ;YES, CHECK CS2
4954 026060 104153      ERROR   153         ;CS1 INCORRECT
4955 026062 000421      BR       TST30       ;GO ON TO NEXT TEST
4956
4957 026064 023737 003510 003450 22$:  CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4958 026072 001402      BEQ      23$         ;YES, CHECK DATA
4959 026074 104154      ERROR   154         ;CS2 INCORRECT
4960 026076 000413      BR       TST30       ;GO ON TO NEXT TEST
4961
4962 026100 023737 003522 003462 23$:  CMP      E.DB,T.DB    ;CHECK IF DATA CORRECT
4963 026106 001401      BEQ      24$         ;YES, GET NEXT HEADER WORD
4964 026110 104155      ERROR   155         ;DATA INCORRECT
4965 026112 005237 003624 24$:  INC      WRDCNT      ;INCREMENT WORD COUNT
4966 026116 022737 000003 003624      CMP      #3,WRDCNT    ;CHECK IF ALL THREE WORDS CHECK
4967 026124 001327      BNE      20$         ;NO, GET NEXT WORD
4968

```

```

*****
*TEST 30      READ LOOPBACK (PART 4)

```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE, 225 ZEROES, A
* ONE, AND A HEADER CONSISTING OF THE THREE
* FOLLOWING WORDS:

```

```

*
*      044444
*      022222
*      111111

```

```

4980
4981
4982

```

```

4983
4984
4985
4986
4987
4988 026126 000004
4989 026130 012737 000144 001200
4990 026136 013702 001270
4991 026142 012762 100000 000000
4992 026150 012762 000040 000026
4993 026156 012762 000025 000000
4994 026164 012700 000312
4995 026170 012762 000440 000026 15:
4996 026176 012762 000040 000026
4997 026204 005300
4998 026206 001370
4999 026210 012762 000140 000026
5000 026216 012762 000040 000026
5001 026224 005037 003614
5002 026230 005037 003616
5003 026234 012700 000341
5004 026240 004737 046156 25:
5005 026244 005300
5006 026246 001374
5007 026250 012737 000001 003614
5008 026256 004737 046156
5009 026262 012701 000003
5010 026266 012703 067002
5011 026272 012737 000025 003500
5012 026300 012304 55:
5013 026302 012700 000020
5014 026306 013737 003614 003616 65:
5015 026314 006004
5016 026316 103403
5017 026320 005037 003614
5018 026324 000403
5019
5020 026326 012737 000001 003614 75:
5021 026334 004737 046156 85:
5022 026340 016237 000000 003440
5023 026346 023737 003500 003440
5024 026354 001417
5025 026356 012737 000003 003624
5026 026364 160137 003624
5027 026370 012737 000020 003622
5028 026376 160037 003622
5029 026402 104150
5030 026404 012762 100000 000000
5031 026412 000522
5032
5033 026414 005300 95:
5034 026416 001333
5035 026420 005301
5036 026422 001326
5037 026424 012700 000004
5038 026430 013737 003614 003616 155:

```

```

;*
;* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
;* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
;*
*****
TST30: SCOPE
MOV #100,STIMES ;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
MOV #50,#4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
15: MOV #DMD:MCLK,RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD,RKMR1(R2)
DEC R0
BNE 15
MOV #DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
CLR PR.BIT ;INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225,R0
25: JSR PC,ROBIT ;SIMULATE SYNCH
DEC R0
BNE 25
MOV #1,PR.BIT
JSR PC,ROBIT
MOV #3,R1 ;LOAD NUMBER OF WORDS
MOV #HEAD4,R3 ;LOAD ADDRESS OF DATA
MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
55: MOV (R3)+,R4 ;GET DATA
MOV #16,R0 ;LOAD BIT COUNT
65: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT
BCS 75 ;CHECK IF 1
CLR PR.BIT ;NO, ZERO
BR 85 ;SIMULATE READ DATA
75: MOV #1,PR.BIT ;ONE
85: JSR PC,ROBIT ;SIMULATE READ DATA
MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
BEQ 95 ;YES, SIMULATE NEXT BIT
MOV #3,WRCNT ;LOAD WORD COUNT
SUB R1,WRCNT
MOV #16,BITCNT ;LOAD BIT COUNT
SUB R0,BITCNT
ERROR 150 ;CS1 INCORRECT DURING HEADER
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
BR TST31 ;GO ON TO NEXT TEST
95: DEC R0 ;CHECK IF READY FOR NEXT WORD
BNE 65 ;NO, GET NEXT BIT
DEC R1 ;CHECK IF HEADER FINISHED
BNE 55 ;NO, GET NEXT WORD
MOV #4,R0 ;LOAD COUNT FOR POSTAMBLE
155: MOV PR.BIT,M1.BIT ;STORE LAST BIT

```

H08

5039	026436	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
5040	026442	004737	046156			JSR	PC,R0BIT	:SIMULATE 1 BIT READ
5041	026446	005300				DEC	R0	:CHECK IF TIME FOR READY
5042	026450	001367				BNE	15\$:NO, CONTINUE WITH POSTAMBLE
5043	026452	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
5044	026460	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
5045	026466	012737	000224	003500		MOV	#RDY!R0HEAD8<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
5046	026474	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
5047	026502	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
5048	026510	001401				BEQ	16\$:YES, CHECK CS2
5049	026512	104151				ERROR	151	:CS1 INCORRECT
5050	026514	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5051	026522	001401				BEQ	17\$:YES, CHECK DATA
5052	026524	104152				ERROR	152	:CS2 INCORRECT
5053	026526	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
5054	026532	012703	067002			MOV	#HEAD4,R3	:GET ADDRESS OF DATA
5055	026536	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
5056	026542	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
5057	026550	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
5058	026556	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
5059	026564	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
5060	026572	001003				BNE	21\$:NO, CHECK CS1
5061	026574	012737	000100	003510		MOV	#IR,E.CS2	:STORE EXPECTED CS2
5062	026602	023737	003500	003440	21\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
5063	026610	001402				BEQ	22\$:YES, CHECK CS2
5064	026612	104153				ERROR	153	:CS1 INCORRECT
5065	026614	000421				BR	TST31	:GO ON TO NEXT TEST
5066								
5067	026616	023737	003510	003450	22\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5068	026624	001402				BEQ	23\$:YES, CHECK DATA
5069	026626	104154				ERROR	154	:CS2 INCORRECT
5070	026630	000413				BR	TST31	:GO ON TO NEXT TEST
5071								
5072	026632	023737	003522	003462	23\$:	CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
5073	026640	001401				BEQ	24\$:YES, GET NEXT HEADER WORD
5074	026642	104155				ERROR	155	:DATA INCORRECT
5075	026644	005237	003624		24\$:	INC	WRDCNT	:INCREMENT WORD COUNT
5076	026650	022737	000003	003624		CMP	#3,WRDCNT	:CHECK IF ALL THREE WORDS CHECK
5077	026656	001327				BNE	20\$:NO, GET NEXT WORD

```

*****
*TEST 31      READ LOOPBACK (PART 5)
*
*      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
*      IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN RK06
*      26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
*      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
*      SIMULATE SECTOR PULSE, 255 ZEROES, A
*      ONE, AND A HEADER CONSISTING OF THE THREE
*      FOLLOWING WORDS.
*
*              052012
*              100520
*              052012
*
*      MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD

```

5078
5079
5080
5081
5082
5083
5084
5085
5086
5087
5088
5089
5090
5091
5092
5093
5094

```

5095      ;*      IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
5096      ;*
5097      ;*
5098      ;*****
5098 026660 000004          TST31: SCOPE
5099 026662 012737 000144 001200      MOV      #100.,$TIMES      ;;DO 100. ITERATIONS
5100 026670 013702 001270          MOV      $BASE,R2        ;;LOAD RK611 BASE
5101 026674 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
5102 026702 012762 000040 000026      MOV      #DMD,RKMR1(R2)  ;;PUT RK611 IN DIAGNOSTIC MODE
5103 026710 012762 000025 000000      MOV      #RDHEAD,RKCS1(R2) ;;ISSUE READ HEADER
5104 026716 012700 000312          MOV      #50.*4+2,R0     ;;ISSUE ENOUGH CLOCKS UNTIL READY
5105 026722 012762 000440 000026 15:      MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5106 026730 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5107 026736 005300          DEC      R0
5108 026740 001370          BNE     15
5109 026742 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5110 026750 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5111 026756 005037 003614          CLR     PR.BIT          ;;INITIALIZE PRESENT BIT AND
5112 026762 005037 003616          CLR     M1.BIT          ;; PREVIOUS BIT
5113 026766 012700 000341          MOV      #225.,R0
5114 026772 004737 046156          JSR     PC,R0BIT        ;SIMULATE SYNCH
5115 026776 005300          DEC     R0
5116 027000 001374          BNE     25
5117 027002 012737 000001 003614      MOV      #1,PR.BIT
5118 027010 004737 046156          JSR     PC,R0BIT
5119 027014 012701 000003          MOV      #3,R1          ;;LOAD NUMBER OF WORDS
5120 027020 012703 067010          MOV      #HEADS,R3      ;;LOAD ADDRESS OF DATA
5121 027024 012737 000025 003500      MOV      #RDHEAD,E.CS1  ;;LOAD EXPECTED CS1
5122 027032 012304          MOV      (R3)+,R4       ;;GET DATA
5123 027034 012700 000020          MOV      #16.,R0       ;;LOAD BIT COUNT
5124 027040 013737 003614 003616 65:      MOV      PR.BIT,M1.BIT  ;;STORE PREVIOUS BIT
5125 027046 006004          ROR     R4              ;;GET NEXT BIT
5126 027050 103403          BCS     75              ;;CHECK IF 1
5127 027052 005037 003614          CLR     PR.BIT         ;;NO, ZERO
5128 027056 000403          BR     B5              ;;SIMULATE READ DATA
5129
5130 027060 012737 000001 003614 75:      MOV      #1,PR.BIT     ;;ONE
5131 027066 004737 046156          JSR     PC,R0BIT       ;;SIMULATE READ DATA
5132 027072 016237 000000 003440 85:      MOV      RKCS1(R2),T.CS1 ;;READ COMMAND AND STATUS REG. 1
5133 027100 023737 003500 003440      CMP     E.CS1,T.CS1    ;;CHECK IF CS1 CORRECT
5134 027106 001417          BEQ     95              ;;YES, SIMULATE NEXT BIT
5135 027110 012737 000003 003624      MOV      #3,WRDCNT     ;;LOAD WORD COUNT
5136 027116 160137 003624          SUB     R1,WRDCNT
5137 027122 012737 000020 003622      MOV      #16.,BITCNT   ;;LOAD BIT COUNT
5138 027130 160037 003622          SUB     R0,BITCNT
5139 027134 104150          ERROR  150             ;;CS1 INCORRECT DURING HEADER
5140 027136 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
5141 027144 000522          BR     TST32          ;;GO ON TO NEXT TEST
5142
5143 027146 005300          95:      DEC     R0              ;;CHECK IF READY FOR NEXT WORD
5144 027150 001333          BNE     65              ;;NO, GET NEXT BIT
5145 027152 005301          DEC     R1              ;;CHECK IF HEADER FINISHED
5146 027154 001326          BNE     55              ;;NO, GET NEXT WORD
5147 027156 012700 000004          MOV      #4,R0         ;;LOAD COUNT FOR POSTAMBLE
5148 027162 013737 003614 003616 155:      MOV      PR.BIT,M1.BIT  ;;STORE LAST BIT
5149 027170 005037 003614          CLR     PR.BIT         ;;LOAD NEXT BIT
5150 027174 004737 046156          JSR     PC,R0BIT       ;;SIMULATE 1 BIT READ

```

JOB

5151	027200	005300				DEC	R0	:CHECK IF TIME FOR READY
5152	027202	001367				BNE	15\$:NO, CONTINUE WITH POSTAMBLE
5153	027204	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
5154	027212	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
5155	027220	012737	000224	003500		MOV	#RDY!RDHEAD<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
5156	027226	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
5157	027234	023737	003500	003440		CMP	E.CS1,↑.CS1	:CHECK CS1 CORRECT
5158	027242	001401				BEQ	16\$:YES, CHECK CS2
5159	027244	104151				ERROR	151	:CS1 INCORRECT
5160	027246	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5161	027254	001401				BEQ	17\$:YES, CHECK DATA
5162	027256	104152				ERROR	152	:CS2 INCORRECT
5163	027260	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
5164	027264	012703	067010			MOV	#HEAD5,R3	:GET ADDRESS OF DATA
5165	027270	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
5166	027274	016237	000024	003462		MOV	RKDB(R2),↑.DB	:GET ACTUAL DATA
5167	027302	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
5168	027310	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
5169	027316	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
5170	027324	001003				BNE	21\$:NO, CHECK CS1
5171	027326	012737	000100	003510		MOV	#IR,E.CS2	:STORE EXPECTED CS2
5172	027334	023737	003500	003440	21\$:	CMP	E.CS1,↑.CS1	:CHECK CS1 CORRECT
5173	027342	001402				BEQ	22\$:YES, CHECK CS2
5174	027344	104153				ERROR	153	:CS1 INCORRECT
5175	027346	000421				BR	TST32	:GO ON TO NEXT TEST
5176								
5177	027350	023737	003510	003450	22\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5178	027356	001402				BEQ	23\$:YES, CHECK DATA
5179	027360	104154				ERROR	154	:CS2 INCORRECT
5180	027362	000413				BR	TST32	:GO ON TO NEXT TEST
5181								
5182	027364	023737	003522	003462	23\$:	CMP	E.DB,↑.DB	:CHECK IF DATA CORRECT
5183	027372	001401				BEQ	24\$:YES, GET NEXT HEADER WORD
5184	027374	104155				ERROR	155	:DATA INCORRECT
5185	027376	005237	003624		24\$:	INC	WRDCNT	:INCREMENT WORD COUNT
5186	027402	022737	000003	003624		CMP	#3,WRDCNT	:CHECK IF ALL THREE WORDS CHECK
5187	027410	001327				BNE	20\$:NO, GET NEXT WORD
5188								

```

*****
*TEST 32      READ HEADER IN 18 BIT MODE
*
* CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE, 255 ZEROES, A
* ONE, AND A HEADER CONSISTING OF THE THREE
* FOLLOWING WORDS:
*
*           177777
*           000000
*           177777
*
* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
* IS TRANSFERRED.  CHECK THE SILO FOR CORRECT CONTENTS.
*****

```

```

*****

```

K08

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 100
 DZR6CB.P11 28-JAN-77 14:04 T32 READ HEADER IN 18 BIT MODE

SEQ 0100

5207	027412	000004			TST32:	SCOPE		
5208	027414	012737	000144	001200		MOV	#100, \$TIMES	:: DO 100. ITERATIONS
5209	027422	013702	001270			MOV	\$BASE, R2	:: LOAD RK611
5210	027426	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:: CLEAR RK611
5211	027434	012762	000040	000026		MOV	#DMD, RKMR1(R2)	:: PUT RK611 IN DIAGNOSTIC MODE
5212	027442	012762	010025	000000		MOV	#CFMT!RDHEAD, RKCS1(R2)	:: ISSUE READ HEADER (24 SECTOR FORMAT)
5213	027450	012700	000312			MOV	#SO, #4+2, R0	:: ISSUE ENOUGH CLOCKS UNTIL READY
5214	027454	012762	000440	000026	15:	MOV	#DMD!MCLK, RKMR1(R2)	:: FOR SECTOR PULSE
5215	027462	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5216	027470	005300				DEC	R0	
5217	027472	001370				BNE	15	
5218	027474	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	:: SIMULATE SECTOR PULSE
5219	027502	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5220	027510	005037	003614			CLR	PR.BIT	:: INITIALIZE PRESENT BIT AND
5221	027514	005037	003616			CLR	M1.BIT	:: PREVIOUS BIT
5222	027520	012700	000377			MOV	#255, R0	
5223	027524	004737	046156		25:	JSR	PC, R0BIT	:: SIMULATE SYNCH
5224	027530	005300				DEC	R0	
5225	027532	001374				BNE	25	
5226	027534	012737	000001	003614		MOV	#1, PR.BIT	
5227	027542	004737	046156			JSR	PC, R0BIT	
5228	027546	012701	000003			MOV	#3, R1	:: LOAD NUMBER OF WORDS
5229	027552	012703	066752			MOV	#HEAD, R3	:: LOAD ADDRESS OF DATA
5230	027556	012737	010025	003500		MOV	#CFMT!RDHEAD, E.CS1	:: LOAD EXPECTED CS1
5231	027564	012304			55:	MOV	(R3)+, R4	:: GET DATA
5232	027566	012700	000020			MOV	#16, R0	:: LOAD BIT COUNT
5233	027572	013737	003614	003616	65:	MOV	PR.BIT, M1.BIT	:: STORE PRESENT BIT
5234	027600	006004				ROR	R4	:: GET NEXT BIT
5235	027602	103403				BCS	75	:: CHECK IF 1
5236	027604	005037	003614			CLR	PP.BIT	:: NO, ZERO
5237	027610	000403				BR	85	:: SIMULATE READ DATA
5238								
5239	027612	012737	000001	003614	75:	MOV	#1, PR.BIT	:: ONE
5240	027620	004737	046156		85:	JSR	PC, R0BIT	:: SIMULATE READ DATA
5241	027624	016237	000000	003440		MOV	RKCS1(R2), T.CS1	:: READ COMMAND AND STATUS REG. 1
5242	027632	023737	003500	003440		CMP	E.CS1, T.CS1	:: CHECK IF CS1 CORRECT
5243	027640	001417				BEQ	95	:: YES, SIMULATE NEXT BIT
5244	027642	012737	000003	003624		MOV	#3, WRDCNT	:: LOAD WORD COUNT
5245	027650	160137	003624			SUB	R1, WRDCNT	
5246	027654	012737	000020	003622		MOV	#16, BITCNT	:: LOAD BIT COUNT
5247	027662	160037	003622			SUB	R0, BITCNT	
5248	027666	104156				ERROR	156	:: CS1 INCORRECT DURING HEADER
5249	027670	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:: CLEAR RK611
5250	027676	000522				BR	TST33	:: GO ON TO NEXT TEST
5251								
5252	027700	005300			95:	DEC	R0	:: CHECK IF READY FOR NEXT WORD
5253	027702	001333				BNE	65	:: NO, GET NEXT BIT
5254	027704	005301				DEC	R1	:: CHECK IF HEADER FINISHED
5255	027706	001326				BNE	55	:: NO, GET NEXT WORD
5256	027710	012700	000004			MOV	#4, R0	:: LOAD COUNT FOR POSTAMBLE
5257	027714	013737	003614	003616	155:	MOV	PR.BIT, M1.BIT	:: STORE LAST BIT
5258	027722	005037	003614			CLR	PR.BIT	:: LOAD NEXT BIT
5259	027726	004737	046156			JSR	PC, R0BIT	:: READ BIT
5260	027732	005300				DEC	R0	:: CHECK IF TIME FOR READY
5261	027734	001367				BNE	155	:: NO, CONTINUE WITH POSTAMBLE
5262	027736	016237	000000	003440		MOV	RKCS1(R2), T.CS1	:: GET CURRENT CS1

L08

```

5263 027744 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
5264 027752 012737 010224 003500 MOV #CFMT!RDY!RDHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5265 027760 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
5266 027766 023737 003500 003440 CMP E.CS1,↑.CS1 ;CHECK CS1 CORRECT
5267 027774 001401 BEQ 16$ ;YES, CHECK CS2
5268 027776 104157 ERROR 157 ;CS1 INCORRECT
5269 030000 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5270 0 1006 001401 BEQ 17$ ;YES, CHECK DATA
5271 0 1010 104160 ERROR 160 ;CS2 INCORRECT
5272 0 1012 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
5273 0 1016 012703 066752 MOV #HEAD1,R3 ;GET ADDRESS OF DATA
5274 0 1022 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
5275 0 1026 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
5276 0 1034 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
5277 0 1042 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
5278 0 1050 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
5279 0 1056 001003 BNE 21$ ;NO, CHECK CS1
5280 030060 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
5281 030066 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5282 030074 001402 BEQ 22$ ;YES, CHECK CS2
5283 030076 104161 ERROR 161 ;CS1 INCORRECT
5284 030100 000421 BR ↑T33 ;GO ON TO NEXT TEST
5285
5286 030102 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5287 030110 001402 BEQ 23$ ;YES, CHECK DATA BUFFER
5288 030112 104162 ERROR 162 ;CS2 INCORRECT
5289 030114 000413 BR ↑T33 ;GO ON TO NEXT TEST
5290
5291 030116 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK DATA BUFFER CORRECT
5292 030124 001401 BEQ 24$ ;YES, GET NEXT WORD
5293 030126 104163 ERROR 163 ;DATA BUFFER INCORRECT
5294 030130 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
5295 030134 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF FINISHED
5296 030142 001327 BNE 20$ ;NO READ NEXT WORD
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310 030144 000004 ↑T33: SCOPE
5311 030146 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS
5312 030154 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5313 030160 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5314 030166 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
5315 030174 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEAD
5316 030202 012700 000312 MOV #50,*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
5317 030206 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5318 030214 012762 000040 000026 MOV #DMD,RKMR1(R2)
    
```

```

*****
*TEST 33 SYNCH DETECT IN READ HEADER
*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE AND 350 ZEROES. MAKE
* SURE READY REMAINS RESET AND THE SILO REMAINS
* EMPTY.
*****
    
```

MO8

```

5319 030222 005300 DEC R0
5320 030224 001370 BNE 1$
5321 030226 012762 000140 000026 MOV #0MD!MSP,RKMR1(R2);SIMULATE TO SECTOR PULSE
5322 030234 012762 000040 000026 MOV #0MD,RKMR1(R2)
5323 030242 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND
5324 030246 005037 003616 CLR M1.BIT ;PREVIOUS BIT
5325 030252 012737 000025 003500 MOV #ROHEAD,E.CS1 ;LOAD EXPECTED CS1
5326 030260 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
5327 030266 005037 003622 CLR BITCNT ;SIMULATE 350 ZEROES
5328 030272 004737 046156 JSR PC,ROBIT 2$:
5329 030276 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE CS1
5330 030304 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE CS2
5331 030312 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5332 030320 001402 BEQ 3$ ;YES, CHECK CS2
5333 030322 104164 ERROR 164 ;CS1 INCORRECT
5334 030324 000447 BR TST34 ;GO ON TO NEXT TEST
5335
5336 030326 023737 003510 003450 3$: CMP E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
5337 030334 001402 BEQ 4$ ;YES, CHECK IF SILO EMPTY
5338 030336 104165 ERROR 165 ;CS2 INCORRECT
5339 030340 000441 BR TST34 ;GO ON TO NEXT TEST
5340 030342 005237 003622 4$: INC BITCNT ;INCREMENT BIT COUNT
5341 030346 022737 000536 003622 CMP #350,BITCNT ;CHECK IF FINISHED
5342 030354 001346 BNE 2$ ;NO, SIMULATE NEXT ZERO
5343 030356 005762 000024 TST RKDB(R2) ;READ DATA BUFFER
5344 030362 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE CS1 AND CS2
5345 030370 016237 000010 003450 MOV RKCS2(R2),T.CS2
5346 030376 016237 100224 003500 MOV #CERR!RDY!ROHEAD<1C<GO>>,E.CS1 ;LOAD EXPECT CS1
5347 030404 016237 100100 003510 MOV #OCK!IR,E.CS2 ;LOAD EXPECTED CS2
5348 030412 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK FOR CONTROLLER ERROR
5349 030420 001401 BEQ 5$ ;YES, CHECK FOR DATA LATE
5350 030422 104166 ERROR 166 ;CS1 INCORRECT
5351 030424 023737 003510 003450 5$: CMP E.CS2,T.CS2 ;CHECK FOR DATA LATE
5352 030432 001401 BEQ 6$ ;YES, CHECK RK611
5353 030434 104167 ERROR 167 ;CS2 INCORRECT
5354 030436 012762 100000 000000 6$: MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5355
5356 *****
5357 *TEST 34 ZERO SYNCH ON READ
5358 *
5359 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5360 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5361 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
5362 * SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF
5363 * BIT TIME, A ONE, AND A HEADER CONSISTING OF THE
5364 * THREE FOLLOWING WORDS:
5365 *
5366 * 177777
5367 * 000000
5368 * 177777
5369 *
5370 * MAKE SURE THAT READY COMES AFTER THE THIRD WORD
5371 * IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
5372 *
5373 *****
5374 030444 000004 TST34: SCOPE
    
```

N08

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MO-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 103
 DZR6CB.P11 28-JAN-77 14:04 T34 ZERO SYNCH ON READ

SEQ 0103

5375	030446	012737	000144	001200		MOV	#100,STIMES	::DO 100, ITERATIONS
5376	030454	013702	001270			MOV	\$BASE,R2	::LOAD RK611 BASE
5377	030460	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	::CLEAR RK611
5378	030466	012762	000040	000026		MOV	#DMD,RKMR1(R2)	::PUT RK611 IN DIAGNOSTIC MODE
5379	030474	012762	000025	000000		MOV	#RDHEAD,RKCS1(R2)	::ISSUE READ HEADER
5380	030502	012700	000312			MOV	#50,#4+2,R0	::ISSUE ENOUGH CLOCKS UNTIL READY
5381	030506	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	::FOR SECTOR PULSE
5382	030514	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5383	030522	005300				DEC	R0	
5384	030524	001370				BNE	1\$	
5385	030526	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	::SIMULATE SECTOR PULSE
5386	030534	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5387	030542	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	::SHIFT DATA ONE HALF BIT TIME
5388	030550	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5389	030556	005037	003614			CLR	PR.BIT	::INITIALIZE PRESENT BIT AND
5390	030562	005037	003616			CLR	M1.BIT	::PREVIOUS BIT
5391	030566	012700	000341			MOV	#225,R0	
5392	030572	004737	046156		2\$:	JSR	PC,R0BIT	::SIMULATE SYNCH
5393	030576	005300				DEC	R0	
5394	030600	001374				BNE	2\$	
5395	030602	012737	000001	003614		MOV	#1,PR.BIT	
5396	030610	004737	046156			JSR	PC,R0BIT	
5397	030614	012701	000003			MOV	#3,R1	::LOAD NUMBER OF WORDS
5398	030620	012703	066752			MOV	#HEAD1,R3	::LOAD ADDRESS OF DATA
5399	030624	012737	000025	003500		MOV	#RDHEAD,E.CS1	::LOAD EXPECTED CS1
5400	030632	012304			5\$:	MOV	(R3)+R4	::GET DATA
5401	030634	012700	000020			MOV	#16,R0	::LOAD BIT COUNT
5402	030640	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT	::STORE PREVIOUS BIT
5403	030646	006004				ROR	R4	::GET NEXT BIT
5404	030650	103403				BCS	7\$::CHECK IF 1
5405	030652	005037	003614			CLR	PR.BIT	::NO, ZERO
5406	030656	000403				BR	8\$::SIMULATE READ DATA
5407								
5408	030660	012737	000001	003614	7\$:	MOV	#1,PR.BIT	::ONE
5409	030666	004737	046156		8\$:	JSR	PC,R0BIT	::SIMULATE READ DATA
5410	030672	016237	000000	003440		MOV	RKCS1(R2),T.CS1	::READ COMMAND AND STATUS REG. 1
5411	030700	023737	003500	003440		CMP	E.CS1,T.CS1	::CHECK IF CS1 CORRECT
5412	030706	001417				BEQ	9\$::YES, SIMULATE NEXT BIT
5413	030710	012737	000003	003624		MOV	#3,WRDCNT	::LOAD WORD COUNT
5414	030716	160137	003624			SUB	R1,WRDCNT	
5415	030722	012737	000020	003622		MOV	#16,BITCNT	::LOAD BIT COUNT
5416	030730	160037	003622			SUB	R0,BITCNT	
5417	030734	104150				ERROR	150	::CS1 INCORRECT DURING HEADER
5418	030736	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	::CLEAR RK611
5419	030744	000522				BR	TST35	::GO ON TO NEXT TEST
5420								
5421	030746	005300			9\$:	DEC	R0	::CHECK IF READY FOR NEXT WORD
5422	030750	001333				BNE	6\$::NO, GET NEXT BIT
5423	030752	005301				DEC	R1	::CHECK IF HEADER FINISHED
5424	030754	001326				BNE	5\$::NO, GET NEXT WORD
5425	030756	012700	000004			MOV	#4,R0	::LOAD COUNT FOR POSTAMBLE
5426	030762	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	::STORE LAST BIT
5427	030770	005037	003614			CLR	PR.BIT	::LOAD NEXT BIT
5428	030774	004737	046156			JSR	PC,R0BIT	::SIMULATE 1 BIT READ
5429	031000	005300				DEC	R0	::CHECK IF TIME FOR READY
5430	031002	001367				BNE	15\$::NO, CONTINUE WITH POSTAMBLE

```

5431 031004 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
5432 031012 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
5433 031020 012737 000224 003500 MOV #RDY!RDHEAD8<+C<GO>> E.CS1 ;LOAD EXPECTED CS1
5434 031026 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
5435 031034 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5436 031042 001401 BEQ 16$ ;YES, CHECK CS2
5437 031044 104151 ERROR 151 ;CS1 INCORRECT
5438 031044 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5439 031044 001401 BEQ 17$ ;YES, CHECK DATA
5440 031046 104152 ERROR 152 ;CS2 INCORRECT
5441 031050 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
5442 031064 012703 066752 MOV #HEAD1,R3 ;GET ADDRESS OF DATA
5443 031070 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
5444 031074 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
5445 031102 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
5446 031110 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
5447 031116 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
5448 031124 001003 BNE 21$ ;NO, CHECK CS1
5449 031126 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
5450 031134 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5451 031142 001402 BEQ 22$ ;YES, CHECK CS2
5452 031144 104153 ERROR 153 ;CS1 INCORRECT
5453 031146 000421 BR TST35 ;GO ON TO NEXT TEST
5454
5455 031150 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5456 031156 001402 BEQ 23$ ;YES, CHECK DATA
5457 031160 104154 ERROR 154 ;CS2 INCORRECT
5458 031162 000413 BR TST35 ;GO ON TO NEXT TEST
5459
5460 031164 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
5461 031172 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
5462 031174 104155 ERROR 155 ;DATA INCORRECT
5463 031176 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
5464 031202 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
5465 031210 001327 BNE 20$ ;NO, GET NEXT WORD

```

.SBTTL **MFM WRITE LOOPBACK TESTS

```

*****
*TEST 35 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
* INDEX PULSE AND 500 DATA BITS. MAKE SURE THAT
* ZEROS ARE WRITTEN. SIMULATE SECTOR PULSE AND MAKE SURE
* WRITE GATE RESETS.
*****

```

```

5461 031212 000004 †T35: SCOPE
5462 031214 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS
5463 031222 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5464 031226 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
5465 031234 012762 000040 000026 MOV #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5466 031242 012762 000140 000026 MOV #DMD!MSP,RKMRI(R2) ;INITIALIZE ROM

```

```

5487 031250 012762 000040 000026 MOV #DMD,RKMR1(R2)
5488 031256 012762 067254 000004 MOV #WRBUFF,RKBA(R2);ISSUE WRITE HEADER
5489 031264 012762 177777 000002 MOV #-1,RKWC(R2)
5490 031272 012762 000027 000000 MOV #WHEAD,RKCS1(R2)
5491 031300 012700 002364 MOV #<256.+48.+64.+256.+10.>*2,RO;ISSUE ENOUGH CLOCKS
5492 UNTIL READY FOR INDEX PULSE
5493 031304 012762 000440 000026 15: MOV #DMD!MCLK,RKMR1(R2)
5494 031312 012762 000040 000026 MOV #DMD,RKMR1(R2)
5495 031320 005300 DEC RO
5496 031322 001370 BNE 15
5497 031324 012700 000004 MOV #4,RO;SIMULATE INDEX PULSE
5498 031330 012762 000240 000026 MOV #MIND!DMD,RKMR1(R2)
5499 031336 012762 000640 000026 25: MOV #DMD!MIND!MCLK,RKMR1(R2)
5500 031344 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5501 031352 005300 DEC RO
5502 031354 001370 BNE 25
5503 031356 012762 000040 000026 MOV #DMD,RKMR1(R2)
5504 031364 012737 062040 003524 MOV #DMD!MEND!ECCW!WRTGAT,E.MR1;INITIALIZE EXPECTED
5505 ; MAINT. REG
5506 031372 012700 000002 MOV #2,RO;WAIT FOR WRITE GATE
5507 031376 012762 000440 000026 35: MOV #DMD!MCLK,RKMR1(R2)
5508 031404 012762 000040 000026 MOV #DMD,RKMR1(R2)
5509 031412 005300 DEC RO
5510 031414 001370 BNE 35
5511 031416 005037 003626 CLR SECCNT;CLEAR SECTOR COUNT
5512 031422 005037 003612 CLR P1.BIT;INITIALIZE BIT GENERATION
5513 031426 005037 003614 CLR PR.BIT
5514 031432 005037 003616 CLR M1.BIT
5515 031436 005037 003620 CLR M2.BIT
5516 031442 012700 000764 MOV #500,RO;LOAD COUNT FOR 500 BITS
5517 031446 012737 062464 003170 MOV #EM230,EMW;LOAD ERROR MESSAGE
5518 031454 005037 003622 CLR BITCNT;CLEAR BIT COUNT
5519 031460 004737 045206 55: JSR PC,WRTBIT;WRITE ONE BIT
5520 031464 104170 ERROR 170;ERROR IN WRITE
5521 031466 005237 003622 INC BITCNT;INCREMENT NUMBER OF BITS WRITTEN
5522 031472 005300 DEC RO;CHECK IF FINISHED
5523 031474 001371 BNE 55;NO,CONTINUE
5524 031476 042737 040000 003524 BIC #WRTGAT,E.MR1;GENERATE EXPECTED MR1
5525 031504 052737 000100 003524 BIS #MSP,E.MR1
5526 031512 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2);RAISE SECTOR PULSE
5527 031520 016237 000026 003464 MOV RKMR1(R2),T.MR1;STORE MAINT. REG. 1
5528 031526 023737 003524 003464 CMP E.MR1,T.MR1;STORE MAINT REG 1
5529 031534 001401 BEQ 105;YES,LOWER SECTOR PULSE
5530 031536 104171 ERROR 171;WRITE GATE DID NOT RESET
5531 031540 042737 000100 003524 105: BIC #MSP,E.MR1;GENERATE EXPECTED MR1
5532 031546 052737 040000 003524 BIS #WRTGAT,E.MR1
5533 031554 012762 000040 000026 MOV #DMD,RKMR1(R2);RESET SECTOR PULSE
5534 031562 016237 000026 003464 MOV RKMR1(R2),T.MR1;STORE MAINT REG 1
5535 031570 023737 003524 003464 CMP E.MR1,T.MR1;CHECK MR1 CORRECT
5536 031576 001401 BEQ TST36;YES,GO ON TO NEXT TEST
5537 031600 104172 ERROR 172;WRITE GATE DID NOT SET

```

```

5538
5539 *****
5540 *TEST 36 WRITE LOOPBACK (PART 1)
5541 *
5542 * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

```

```

5543
5544
5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557 031602 000004
5558 031604 012737 000144 001200
5559 031612 013702 001270
5560 031616 012762 100000 000000
5561 031624 012762 000040 000026
5562 031632 012762 066752 000004
5563 031640 012762 177775 000002
5564 031646 012762 000027 000000
5565 031654 012700 000312
5566
5567 031660 012762 000440 000026 15:
5568 031666 012762 000040 000026
5569 031674 005300
5570 031676 001370
5571 031700 012700 000004
5572 031704 012762 000240 000026
5573 031712 012762 000640 000026 25:
5574 031720 012762 000240 000026
5575 031726 005300
5576 031730 001370
5577 031732 012762 000040 000026
5578 031740 012700 000010
5579 031744 012762 000440 000026 35:
5580 031752 012762 000040 000026
5581 031760 005300
5582 031762 001370
5583 031764 012762 000140 000026
5584 031772 012762 000040 000026
5585 032000 005037 003626
5586 032004 012737 063036 003170
5587 032012 012737 062040 003524
5588
5589 032020 005037 003612
5590 032024 005037 003614
5591 032030 005037 003616
5592 032034 005037 003620
5593 032040 012700 000400
5594 032044 005037 003622
5595 032050 004737 045206 55:
5596 032054 104170
5597 032056 005237 003622
5598 032062 005300

```

```

* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
* CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:
*
* 177777
* 000000
* 177777
*
* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
*
*****
†ST36: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD1, RKBA(R2) ; ISSUE WRITE HEADER
MOV #-3, RKWC(R2)
MOV #WRHEAD, RKCS1(R2)
MOV #50, #4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
15: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 15
MOV #4, R0 ; ISSUE INDEX PULSE
MOV #DMD!MIND, RKMR1(R2)
25: MOV #DMD!MIND!MCLK, RKMR1(R2)
MOV #DMD!MIND, RKMR1(R2)
DEC R0
BNE 25
MOV #DMD, RKMR1(R2)
MOV #8, R0 ; WAIT FOR WRITE GATE
35: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 35
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR SECCNT ; INITIALIZE SECTOR COUNT
MOV #EM233, EMW ; LOAD ERROR MESSAGE
MOV #DMD!MEWD!ECCW!WRTGAT, E.MR1 ; INITIALIZE EXPECTED
; MAINT REG 1
; INITIALIZE BIT GENERATION
CLR P1.BIT
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256, R0 ; SIMULATE SYNCH
CLR BITCNT ; INITIALIZE BIT COUNT
55: JSR PC, WRTBIT ; WRITE ONE BIT
ERROR 170 ; DATA INCORRECT
INC BITCNT
DEC R0 ; CHECK IF READY FOR DATA

```

5599	032064	001371				BNE	55		;NO, GENERATE NEXT BIT
5600	032066	012737	000001	003612		MOV	#1, P1.BIT		;PUT IN SYNCH BIT
5601	032074	004737	045206			JSR	PC, WRTBIT		
5602	032100	104170				ERROR	170		;DATA INCORRECT
5603	032102	005037	003622			CLR	BITCNT		;INITIALIZE BIT COUNT
5604	032106	012737	063102	003170		MOV	#EM234, EMW		;LOAD ERROR MESSAGE
5605	032114	012703	066752			MOV	#HEAD1, R3		;LOAD ADDRESS OF DATA
5606	032120	012700	000003			MOV	#3, R0		;LOAD NUMBER WORDS IN HEADER
5607	032124	012304			10\$:	MOV	(R3)+, R4		;GET NEXT WORD
5608	032126	012701	000020			MOV	#16, R1		;LOAD BIT COUNT
5609	032132	013737	003616	003620	12\$:	MOV	M1.BIT, M2.BIT		;SHIFT BITS
5610	032140	013737	003614	003616		MOV	PR.BIT, M1.BIT		
5611	032146	013737	003612	003614		MOV	P1.BIT, PR.BIT		
5612	032154	006004				ROR	R4		;SHIFT IN NEXT BIT
5613	032156	103403				BCS	14\$;CHECK IF ONE
5614	032160	005037	003612			CLR	P1.BIT		;ZERO
5615	032164	000403				BR	15\$;CLOCK IN BIT
5616									
5617	032166	012737	000001	003612	14\$:	MOV	#1, P1.BIT		;ONE
5618	032174	004737	045206		15\$:	JSR	PC, WRTBIT		;WRITE BIT
5619	032200	104170				ERROR	170		;BIT INCORRECT
5620	032202	005237	003622			INC	BITCNT		;INCREMENT BIT COUNT
5621	032206	005301				DEC	R1		;CHECK IF WORD FINISHED
5622	032210	001350				BNE	12\$;NO, CONTINUE WITH NEXT BIT
5623	032212	005300				DEC	R0		;CHECK IF HEADER COMPLETE
5624	032214	001343				BNE	10\$;NO, GET NEXT WORD
5625	032216	012701	000020			MOV	#16, R1		;LOAD BIT COUNT FOR NEXT WORD
5626	032222	013737	003616	003620	18\$:	MOV	M1.BIT, M2.BIT		;SHIFT BIT
5627	032230	013737	003614	003616		MOV	PR.BIT, M1.BIT		
5628	032236	013737	003612	003614		MOV	P1.BIT, PR.BIT		
5629	032244	005037	003612			CLR	P1.BIT		
5630	032250	004737	045206			JSR	PC, WRTBIT		;WRITE ZERO
5631	032254	104170				ERROR	170		;BIT INCORRECT
5632	032256	005237	003622			INC	BITCNT		;INCREMENT BIT COUNT
5633	032262	005301				DEC	R1		;CHECK IF FINISHED
5634	032264	001356				BNE	18\$;NO, CLOCK NEXT BIT
5635	032266	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		;SIMULATE INDEX
5636	032274	012700	000004			MOV	#4, R0		
5637	032300	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)		
5638	032306	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		
5639	032314	005300				DEC	R0		
5640	032316	001370				BNE	20\$		
5641	032320	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
5642	032326	016237	000026	003464		MOV	RKMR1(R2), T.MR1		;GET MAINT REG 1
5643	032334	012737	022040	003524		MOV	#MEWD!ECCW!DMD, E.MR1		;LOAD EXPECTED MR1
5644	032342	023737	003524	003464		CMP	E.MR1, T.MR1		;CHECK MR1 CORRECT (WRITE GATE RESET)
5645	032350	001401				BEQ	25\$;YES, CHECK IF READY SET
5646	032352	104173				ERROR	173		;MAINT REG 1 INCORRECT
5647	032354	012700	000010		25\$:	MOV	#8, R0		;FINISH COMMAND
5648	032360	012762	000440	000026	26\$:	MOV	#DMD!MCLK, RKMR1(R2)		
5649	032366	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
5650	032374	005300				DEC	R0		
5651	032376	001370				BNE	26\$		
5652	032400	016237	000000	003440		MOV	RKCS1(R2), T.CS1		;GET COMMAND AND STATUS REG 1
5653	032406	012737	000226	003500		MOV	#RDY!WHEAD&<↑<GO>>, E.CS1		;LOAD EXPECTED CS1
5654	032414	023737	003500	003440		CMP	E.CS1, T.CS1		;CHECK IF CS1 CORRECT

F09

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 108
 DZR6CB.P11 28-JAN-77 14:04 T36 WRITE LOOPBACK (PART 1)

SEQ 0108

5655 032422 001401
 5656 032424 104174

BEG TST37 ;: YES, GO ON TO NEXT TEST
 ERROR 174 ;: CSI INCORRECT

 *TEST 37 WRITE LOOPBACK (PART 2)

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

000000
 177777
 000000

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
 IS TRANSFERRED. CHECK FOR CORRECT WRITE ENCODED DATA
 AND PRECOMPENSATION.

5677 032426 000004
 5678 032430 012737 000144 001200
 5679 032436 013702 001270
 5680 032442 012762 100000 000000
 5681 032450 012762 000040 000026
 5682 032456 012762 066760 000004
 5683 032464 012762 177775 000002
 5684 032472 012762 000027 000000
 5685 032500 012700 000312
 5686
 5687 032504 012762 000440 000026 15:
 5688 032512 012762 000040 000026
 5689 032520 005300
 5690 032522 001370
 5691 032524 012700 000004
 5692 032530 012762 000240 000026
 5693 032536 012762 000640 000026 25:
 5694 032544 012762 000240 000026
 5695 032552 005300
 5696 032554 001370
 5697 032556 012762 000040 000026
 5698 032564 012700 000010
 5699 032570 012762 000440 000026 35:
 5700 032576 012762 000040 000026
 5701 032604 005300
 5702 032606 001370
 5703 032610 012762 000140 000026
 5704 032616 012762 000040 000026
 5705 032624 005037 003626
 5706 032630 012737 063036 003170
 5707 032636 012737 062040 003524
 5708
 5709 032644 005037 003612
 5710 032650 005037 003614

TST37: SCOPE
 MOV #100, \$TIMES ;: DO 100. ITERATIONS
 MOV \$BASE, R2 ;: LOAD RK611 BASE
 #CCLR, RKCS1(R2) ;: CLEAR RK611
 #DMD, RKMR1(R2) ;: PUT RK611 IN DIAGNOSTIC MODE
 #HEAD2, RKBA(R2) ;: ISSUE WRITE HEADER
 #-3, RKWC(R2)
 #WRHEAD, RKCS1(R2)
 #50. #4+2, R0 ;: ISSUE ENOUGH CLOCKS UNTIL
 ;: READY FOR INDEX PULSE
 15: MOV #DMD!MCLK, RKMR1(R2)
 MOV #DMD, RKMR1(R2)
 DEC R0
 BNE 15
 MOV #4, R0 ;: ISSUE INDEX PULSE
 25: MOV #DMD!MIND, RKMR1(R2)
 MOV #DMD!MIND!MCLK, RKMR1(R2)
 MOV #DMD!MIND, RKMR1(R2)
 DEC R0
 BNE 25
 MOV #DMD, RKMR1(R2)
 #8, R0 ;: WAIT FOR WRITE GATE
 35: MOV #DMD!MCLK, RKMR1(R2)
 MOV #DMD, RKMR1(R2)
 DEC R0
 BNE 35
 MOV #DMD!MSP, RKMR1(R2) ;: SIMULATE SECTOR PULSE
 MOV #DMD, RKMR1(R2)
 CLR SECCNT ;: INITIALIZE SECTOR COUNT
 MOV #EM233, EMW ;: LOAD ERROR MESSAGE
 MOV #DMD!MEWD!ECCW!WRTGAT, E.MR1 ;: INITIALIZE EXPECTED
 ;: MAINT REG 1
 CLR P1.BIT ;: INITIALIZE BIT GENERATION
 CLR PR.BIT

5711	032654	005037	003616		CLR	M1.BIT		
5712	032660	005037	003620		CLR	M2.BIT		
5713	032664	012700	000400		MOV	#256, R0	; SIMULATE SYNCH	
5714	032670	005037	003622		CLR	BITCNT	; INITIALIZE BIT COUNT	
5715	032674	004737	045206	55:	JSR	PC, WRTBIT	; WRITE ONE BIT	
5716	032700	104170			ERROR	170	; DATA INCORRECT	
5717	032702	005237	003622		INC	BITCNT		
5718	032706	005300			DEC	R0	; CHECK IF READY FOR DATA	
5719	032710	001371			BNE	55	; NO, GENERATE NEXT BIT	
5720	032712	012737	000001	003612	MOV	#1, P1.BIT	; PUT IN SYNCH BIT	
5721	032720	004737	045206		JSR	PC, WRTBIT		
5722	032724	104170			ERROR	170	; DATA INCORRECT	
5723	032726	005037	003622		CLR	BITCNT	; INITIALIZE BIT COUNT	
5724	032732	012737	063102	003170	MOV	#EM234, EMW	; LOAD ERROR MESSAGE	
5725	032740	012703	066760		MOV	#HEAD2, R3	; LOAD ADDRESS OF DATA	
5726	032744	012700	000003		MOV	#3, R0	; LOAD NUMBER WORDS IN HEADER	
5727	032750	012304		105:	MOV	(R3)+, R4	; GET NEXT WORD	
5728	032752	012701	000020		MOV	#16, R1	; LOAD BIT COUNT	
5729	032756	013737	003616	003620	125:	MOV	M1.BIT, M2.BIT	; SHIFT BITS
5730	032764	013737	003614	003616		MOV	PR.BIT, M1.BIT	
5731	032772	013737	003612	003614		MOV	P1.BIT, PR.BIT	
5732	033000	006004			ROR	R4	; SHIFT IN NEXT BIT	
5733	033002	103403			BCS	145	; CHECK IF ONE	
5734	033004	005037	003612		CLR	P1.BIT	; ZERO	
5735	033010	000403			BR	155	; CLOCK IN BIT	
5736								
5737	033012	012737	000001	003612	145:	MOV	#1, P1.BIT	; ONE
5738	033020	004737	045206		155:	JSR	PC, WRTBIT	; WRITE BIT
5739	033024	104170			ERROR	170	; BIT INCORRECT	
5740	033026	005237	003622		INC	BITCNT	; INCREMENT BIT COUNT	
5741	033032	005301			DEC	R1	; CHECK IF WORD FINISHED	
5742	033034	001350			BNE	125	; NO, CONTINUE WITH NEXT BIT	
5743	033036	005300			DEC	R0	; CHECK IF HEADER COMPLETE	
5744	033040	001343			BNE	105	; NO, GET NEXT WORD	
5745	033042	012701	000020		MOV	#16, R1	; LOAD BIT COUNT FOR NEXT WORD	
5746	033046	013737	003616	003620	185:	MOV	M1.BIT, M2.BIT	; SHIFT BITS
5747	033054	013737	003614	003616		MOV	PR.BIT, M1.BIT	
5748	033062	013737	003612	003614		MOV	P1.BIT, PR.BIT	
5749	033070	005037	003612		CLR	P1.BIT		
5750	033074	004737	045206		JSR	PC, WRTBIT	; WRITE ZERO	
5751	033100	104170			ERROR	170	; BIT INCORRECT	
5752	033102	005237	003622		INC	BITCNT	; INCREMENT BIT COUNT	
5753	033106	005301			DEC	R1	; CHECK IF FINISHED	
5754	033110	001356			BNE	185	; NO, CLOCK NEXT BIT	
5755	033112	012762	000240	000026	MOV	#DMD!MIND, RKMRI(R2)	; SIMULATE INDEX	
5756	033120	012700	000004		MOV	#4, R0		
5757	033124	012762	000640	000026	205:	MOV	#DMD!MIND!MCLK, RKMRI(R2)	
5758	033132	012762	000240	000026		MOV	#DMD!MIND, RKMRI(R2)	
5759	033140	005300			DEC	R0		
5760	033142	001370			BNE	205		
5761	033144	012762	000040	000026	MOV	#DMD, RKMRI(R2)		
5762	033152	016237	000026	003464	MOV	RKMRI(R2), T.MR1	; GET MAINT REG 1	
5763	033160	012737	022040	003524	MOV	#MEWD!ECCW!DMD, E.MR1	; LOAD EXPECTED MR1	
5764	033166	023737	003524	003464	CMP	E.MR1, T.MR1	; CHECK MR1 CORRECT (WRITE GATE RESET)	
5765	033174	001401			BEQ	255	; YES, CHECK IF READY SET	
5766	033176	104173			ERROR	173	; MAINT REG 1 INCORRECT	

5767	033200	012700	000010		25\$:	MOV	#8, R0	: FINISH COMMAND
5768	033204	012762	000440	000026	26\$:	MOV	#DMD!MCLK, RKMRI(R2)	
5769	033212	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
5770	033220	005300				DEC	R0	
5771	033222	001370				BNE	26\$	
5772	033224	016237	000000	003440		MOV	RKCSI(R2), T.CSI	: GET COMMAND AND STATUS REG 1
5773	033232	012737	000226	003500		MOV	#RDY!WRHEAD8<↑C<GO>>, E.CSI	: LOAD EXPECTED CSI
5774	033240	023737	003500	003440		CMP	E.CSI, T.CSI	: CHECK IF CSI CORRECT
5775	033246	001401				BEQ	TST40	: YES, GO ON TO NEXT TEST
5776	033250	104174				ERROR	174	: CSI INCORRECT

↑TEST 40 WRITE LOOPBACK (PART 3)

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

125252
052525
125252

* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

5796	033252	000004			↑TST40:	SCOPE		
5797	033254	012737	000144	001200		MOV	#100, \$TIMES	: DO 100. ITERATIONS
5798	033262	013702	001270			MOV	\$BASE, R2	: LOAD RK611 BASE
5799	033266	012762	100000	000000		MOV	#CLR, RKCSI(R2)	: CLEAR RK611
5800	033274	012762	000040	000026		MOV	#DMD, RKMRI(R2)	: PUT RK611 IN DIAGNOSTIC MODE
5801	033302	012762	066766	000004		MOV	#HEAD3, RKBA(R2)	: ISSUE WRITE HEADER
5802	033310	012762	177775	000002		MOV	#-3, RKAC(R2)	
5803	033316	012762	000027	000000		MOV	#WRHEAD, RKCSI(R2)	
5804	033324	012700	000312			MOV	#50, #4+2, R0	: ISSUE ENOUGH CLOCKS UNTIL : READY FOR INDEX PULSE
5806	033330	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMRI(R2)	
5807	033336	012762	000440	000026		MOV	#DMD, RKMRI(R2)	
5808	033344	005300				DEC	R0	
5809	033346	001370				BNE	1\$	
5810	033350	012700	000004			MOV	#4, R0	: ISSUE INDEX PULSE
5811	033354	012762	000240	000026		MOV	#DMD!MIND, RKMRI(R2)	
5812	033362	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK, RKMRI(R2)	
5813	033370	012762	000240	000026		MOV	#DMD!MIND, RKMRI(R2)	
5814	033376	005300				DEC	R0	
5815	033400	001370				BNE	2\$	
5816	033402	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
5817	033410	012700	000010			MOV	#8, R0	: WAIT FOR WRITE GATE
5818	033414	012762	000440	000026	3\$:	MOV	#DMD!MCLK, RKMRI(R2)	
5819	033422	012762	000040	000026		MOV	#DMD, RKMRI(R2)	
5820	033430	005300				DEC	R0	
5821	033432	001370				BNE	3\$	
5822	033434	012762	000140	000026		MOV	#DMD!MSP, RKMRI(R2)	: SIMULATE SECTOR PULSE

5823	033442	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5824	033450	005037	003626			CLR	SECCNT	; INITIALIZE SECTOR COUNT
5825	033454	012737	063036	003170		MOV	#EM233,EMW	; LOAD ERROR MESSAGE
5826	033462	012737	062040	003524		MOV	#DMD!MEND!ECCW!WRTGAT E.MR1	; INITIALIZE EXPECTED
5827							MAINT REG	
5828	033470	005037	003612			CLR	P1.BIT	; INITIALIZE BIT GENERATION
5829	033474	005037	003614			CLR	PR.BIT	
5830	033500	005037	003616			CLR	M1.BIT	
5831	033504	005037	003620			CLR	M2.BIT	
5832	033510	012700	000400			MOV	#256,R0	; SIMULATE SYNCH
5833	033514	005037	003622			CLR	BITCNT	; INITIALIZE BIT COUNT
5834	033520	004737	045206		5\$:	JSR	PC,WRTBIT	; WRITE ONE BIT
5835	033524	104170				ERROR	170	; DATA INCORRECT
5836	033526	005237	003622			INC	BITCNT	
5837	033532	005300				DEC	R0	; CHECK IF READY FOR DATA
5838	033534	001371				BNE	5\$; NO, GENERATE NEXT BIT
5839	033536	012737	000001	003612		MOV	#1,P1.BIT	; PUT IN SYNCH BIT
5840	033544	004737	045206			JSR	PC,WRTBIT	
5841	033550	104170				ERROR	170	; DATA INCORRECT
5842	033552	005037	003622			CLR	BITCNT	; INITIALIZE BIT COUNT
5843	033556	012737	063102	003170		MOV	#EM234,EMW	; LOAD ERROR MESSAGE
5844	033564	012703	066766			MOV	#HEAD3,R3	; LOAD ADDRESS OF DATA
5845	033570	012700	000003			MOV	#3,R0	; LOAD NUMBER WORDS IN HEADER
5846	033574	012304			10\$:	MOV	(R3)+,R4	; GET NEXT WORD
5847	033576	012701	000020			MOV	#16,R1	; LOAD BIT COUNT
5848	033602	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	; SHIFT BITS
5849	033610	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5850	033616	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5851	033624	006004				ROR	R4	; SHIFT IN NEXT BIT
5852	033626	103403				BCS	14\$; CHECK IF ONE
5853	033630	005037	003612			CLR	P1.BIT	; ZERO
5854	033634	000403				BR	15\$; CLOCK IN BIT
5855								
5856	033636	012737	000001	003612	14\$:	MOV	#1,P1.BIT	; ONE
5857	033644	004737	045206		15\$:	JSR	PC,WRTBIT	; WRITE BIT
5858	033650	104170				ERROR	170	; BIT INCORRECT
5859	033652	005237	003622			INC	BITCNT	; INCREMENT BIT COUNT
5860	033656	005301				DEC	R1	; CHECK IF WORD FINISHED
5861	033660	001350				BNE	12\$; NO, CONTINUE WITH NEXT BIT
5862	033662	005300				DEC	R0	; CHECK IF HEADER COMPLETE
5863	033664	001343				BNE	10\$; NO, GET NEXT WORD
5864	033666	012701	000020			MOV	#16,R1	; LOAD BIT COUNT FOR NEXT WORD
5865	033672	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	; SHIFT BITS
5866	033700	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5867	033706	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5868	033714	005037	003612			CLR	P1.BIT	
5869	033720	004737	045206			JSR	PC,WRTBIT	; WRITE ZERO
5870	033724	104170				ERROR	170	; BIT INCORRECT
5871	033726	005237	003622			INC	BITCNT	; INCREMENT BIT COUNT
5872	033732	005301				DEC	R1	; CHECK IF FINISHED
5873	033734	001356				BNE	18\$; NO, CLOCK NEXT BIT
5874	033736	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	; SIMULATE INDEX
5875	033744	012700	000004			MOV	#4,R0	
5876	033750	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
5877	033756	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5878	033764	005300				DEC	R0	

5879	033766	001370			BNE	20\$	
5880	033770	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
5881	033776	016237	000026	003464	MOV	RKMR1(R2),T.MR1	;GET MAINT REG 1
5882	034004	012737	022040	003524	MOV	#MEMO!ECCW!DMD,E.MR1	;LOAD EXPECTED MR1
5883	034012	023737	003524	003464	CMP	E.MR1,T.MR1	;CHECK MR1 CORRECT (WRITE GATE RESET)
5884	034020	001401			BEQ	25\$;YES, CHECK IF READY SET
5885	034022	104173			ERROR	173	;MAINT REG 1 INCORRECT
5886	034024	012700	000010		MOV	#8,RO	;FINISH COMMAND
5887	034030	012762	000440	000026	MOV	#DMD!MCLK,RKMR1(R2)	
5888	034036	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
5889	034044	005300			DEC	RO	
5890	034046	001370			BNE	26\$	
5891	034050	016237	000000	003440	MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
5892	034056	012737	000226	003500	MOV	#RDY!WRHEAD<↑C<GO>>,E.CS1	;LOAD EXPECTED CS1
5893	034064	023737	003500	003440	CMP	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
5894	034072	001401			BEQ	TST41	;YES, GO ON TO NEXT TEST
5895	034074	104174			ERROR	174	;CS1 INCORRECT

*TEST 41 WRITE LOOPBACK (PART 4)

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

* 044444
* 022222
* 111111

* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MODE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

†TST41: SCPE

5915	034076	000004			MOV	#100,\$TIMES	;DO 100. ITERATIONS
5916	034100	012737	000144	001200	MOV	\$BASE,R2	;LOAD RK611 BASE
5917	034106	013702	001270		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
5918	034112	012762	100000	000000	MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
5919	034120	012762	000040	000026	MOV	#HEAD4,RKBA(R2)	;ISSUE WRITE HEADER
5920	034126	012762	067002	000004	MOV	#-3,RKWC(R2)	
5921	034134	012762	177775	000002	MOV	#WRHEAD,RKCS1(R2)	
5922	034142	012762	000027	000000	MOV	#50,#4+2,RO	;ISSUE ENOUGH CLOCKS UNTIL ;READY FOR INDEX PULSE
5923	034150	012700	000312				
5924							
5925	034154	012762	000440	000026	MOV	#DMD!MCLK,RKMR1(R2)	
5926	034162	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
5927	034170	005300			DEC	RO	
5928	034172	001370			BNE	1\$	
5929	034174	012700	000004		MOV	#4,RO	;ISSUE INDEX PULSE
5930	034200	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
5931	034206	012762	000640	000026	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
5932	034214	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
5933	034222	005300			DEC	RO	
5934	034224	001370			BNE	2\$	

5935	034226	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5936	034234	012700	000010			MOV	#8,R0	;WAIT FOR WRITE GATE
5937	034240	012762	000440	000026	35:	MOV	#DMD:MCLK,RKMR1(R2)	
5938	034246	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5939	034254	005300				DEC	R0	
5940	034256	001370				BNE	35	
5941	034260	012762	000140	000026		MOV	#DMD:MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
5942	034266	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5943	034274	005037	003626			CLR	SECCNT	;INITIALIZE SECTOR COUNT
5944	034300	012737	063036	003170		MOV	#EM233,EMW	;LOAD ERROR MESSAGE
5945	034306	012737	062040	003524		MOV	#DMD:MEWD:ECCW:WRTGAT,E,MR1	;INITIALIZE EXPECTED
5946								;MAINT REG 1
5947	034314	005037	003612			CLR	P1.BIT	;INITIALIZE BIT GENERATION
5948	034320	005037	003614			CLR	PR.BIT	
5949	034324	005037	003616			CLR	M1.BIT	
5950	034330	005037	003620			CLR	M2.BIT	
5951	034334	012700	000400			MOV	#256,R0	;SIMULATE SYNCH
5952	034340	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5953	034344	004737	045206		55:	JSR	PC,WRTBIT	;WRITE ONE BIT
5954	034350	104170				ERROR	170	;DATA INCORRECT
5955	034352	005237	003622			INC	BITCNT	
5956	034356	005300				DEC	R0	;CHECK IF READY FOR DATA
5957	034360	001371				BNE	55	;NO, GENERATE NEXT BIT
5958	034362	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
5959	034370	004737	045206			JSR	PC,WRTBIT	
5960	034374	104170				ERROR	170	;DATA INCORRECT
5961	034376	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5962	034402	012737	063102	003170		MOV	#EM234,EMW	;LOAD ERROR MESSAGE
5963	034410	012703	067002			MOV	#HEAD4,R3	;LOAD ADDRESS OF DATA
5964	034414	012700	000003			MOV	#3,R0	;LOAD NUMBER WORDS IN HEADER
5965	034420	012304			105:	MOV	(R3)+,R4	;GET NEXT WORD
5966	034422	012701	000020			MOV	#16,R1	;LOAD BIT COUNT
5967	034426	013737	003616	003620	125:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5968	034434	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5969	034442	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5970	034450	006004				ROR	R4	;SHIFT IN NEXT BIT
5971	034452	103403				BCS	145	;CHECK IF ONE
5972	034454	005037	003612			CLR	P1.BIT	;ZERO
5973	034460	000403				BR	155	;CLOCK IN BIT
5974								
5975	034462	012737	000001	003612	145:	MOV	#1,P1.BIT	;ONE
5976	034470	004737	045206		155:	JSR	PC,WRTBIT	;WRITE BIT
5977	034474	104170				ERROR	170	;BIT INCORRECT
5978	034476	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
5979	034502	005301				DEC	R1	;CHECK IF WORD FINISHED
5980	034504	001350				BNE	125	;NO, CONTINUE WITH NEXT BIT
5981	034506	005300				DEC	R0	;CHECK IF HEADER COMPLETE
5982	034510	001343				BNE	105	;NO, GET NEXT WORD
5983	034512	012701	000020			MOV	#16,R1	;LOAD BIT COUNT FOR NEXT WORD
5984	034516	013737	003616	003620	185:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5985	034524	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5986	034532	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5987	034540	005037	003612			CLR	P1.BIT	
5988	034544	004737	045206			JSR	PC,WRTBIT	;WRITE ZERO
5989	034550	104170				ERROR	170	;BIT INCORRECT
5990	034552	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT

5991	034556	005301			DEC	R1	;CHECK IF FINISHED
5992	034560	001356			BNE	18\$;NO, CLOCK NEXT BIT
5993	034562	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	;SIMULATE INDEX
5994	034570	012700	000004		MOV	#4,RO	
5995	034574	012762	000640	000026	20\$: MOV	#DMD:MIND:MCLK,RKMR1(R2)	
5996	034602	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	
5997	034610	005300			DEC	RO	
5998	034612	001370			BNE	20\$	
5999	034614	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
6000	034622	016237	000026	003464	MOV	RKMR1(R2),T.MR1	;GET MAINT REG 1
6001	034630	012737	022040	003524	MOV	#MEMD:ECCW:DMD,E.MR1	;LOAD EXPECTED MR1
6002	034636	023737	003524	003464	CMP	E.MR1,T.MR1	;CHECK MR1 CORRECT (WRITE GATE RESET)
6003	034644	001401			BEQ	25\$;YES, CHECK IF READY SET
6004	034646	104173			ERROR	173	;MAINT REG 1 INCORRECT
6005	034650	012700	000010		25\$: MOV	#8,RO	;FINISH COMMAND
6006	034654	012762	000440	000026	26\$: MOV	#DMD:MCLK,RKMR1(R2)	
6007	034662	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
6008	034670	005300			DEC	RO	
6009	034672	001370			BNE	26\$	
6010	034674	016237	000000	003440	MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
6011	034702	012737	000226	003500	MOV	#RDY:WRHEAD8<↑C<GO>>,E.CS1	;LOAD EXPECTED CS1
6012	034710	023737	003500	003440	CMP	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
6013	034716	001401			BEQ	TST42	;YES, GO ON TO NEXT TEST
6014	034720	104174			ERROR	174	;CS1 INCORRECT

*TEST 42 WRITE LOOPBACK (PART 5)

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

* 052012
* 100520
* 052012

* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

*TST42: SCOPE

6034	034722	000004			MOV	#100,\$TIMES	;DO 100. ITERATIONS
6035	034724	012737	000144	001200	MOV	\$BASE,R2	;LOAD RK611 BASE
6036	034732	013702	001270		MOV	#CLR,RKCS1(R2)	;CLEAR RK611
6037	034736	012762	100000	000000	MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
6038	034744	012762	000040	000026	MOV	#HEAD5,RKBA(R2)	;ISSUE WRITE HEADER
6039	034752	012762	067010	000004	MOV	#-3,RKWC(R2)	
6040	034760	012762	177775	000002	MOV	#WRHEAD,RKCS1(R2)	
6041	034766	012762	000027	000000	MOV	#50.*4+2,RO	;ISSUE ENOUGH CLOCKS UNTIL ;READY FOR INDEX PULSE
6042	034774	012700	000312		MOV		
6043							
6044	035000	012762	000440	000026	1\$: MOV	#DMD:MCLK,RKMR1(R2)	
6045	035006	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
6046	035014	005300			DEC	RO	

6047	035016	001370			BNE	15	
6048	035020	012700	000004		MOV	#4, R0	:ISSUE INDEX PULSE
6049	035024	012762	000240	000026	MOV	#DMO!MIND, RKMR1(R2)	
6050	035032	012762	000640	000026	25: MOV	#DMO!MIND!MCLK, RKMR1(R2)	
6051	035040	012762	000240	000026	MOV	#DMO!MIND, RKMR1(R2)	
6052	035046	005300			DEC	R0	
6053	035050	001370			BNE	25	
6054	035052	012762	000040	000026	MOV	#DMO, RKMR1(R2)	
6055	035060	012700	000010		MOV	#8, R0	:WAIT FOR WRITE GATE
6056	035064	012762	000440	000026	35: MOV	#DMO!MCLK, RKMR1(R2)	
6057	035072	012762	000040	000026	MOV	#DMO, RKMR1(R2)	
6058	035100	005300			DEC	R0	
6059	035102	001370			BNE	35	
6060	035104	012762	000140	000026	MOV	#DMO!MSP, RKMR1(R2)	:SIMULATE SECTOR PULSE
6061	035112	012762	000040	000026	MOV	#DMO, RKMR1(R2)	
6062	035120	005037	003626		CLR	SECCNT	:INITIALIZE SECTOR COUNT
6063	035124	012737	063036	003170	MOV	#EM233, EMW	:LOAD ERROR MESSAGE
6064	035132	012737	062040	003524	MOV	#DMO!MEND!ECCW!WRTGAT, E.MR1	:INITIALIZE EXPECTED
6065							:MAINT REG 1
6066	035140	005037	003612		CLR	P1.BIT	:INITIALIZE BIT GENERATION
6067	035144	005037	003614		CLR	PR.BIT	
6068	035150	005037	003616		CLR	M1.BIT	
6069	035154	005037	003620		CLR	M2.BIT	
6070	035160	012700	000400		MOV	#256, R0	:SIMULATE SYNCH
6071	035164	005037	003622		CLR	BITCNT	:INITIALIZE BIT COUNT
6072	035170	004737	045206		55: JSR	PC, WRTBIT	:WRITE ONE BIT
6073	035174	104170			ERROR	170	:DATA INCORRECT
6074	035176	005237	003622		INC	BITCNT	
6075	035202	005300			DEC	R0	:CHECK IF READY FOR DATA
6076	035204	001371			BNE	55	:NO, GENERATE NEXT BIT
6077	035206	012737	000001	003612	MOV	#1, P1.BIT	:PUT IN SYNCH BIT
6078	035214	004737	045206		JSR	PC, WRTBIT	
6079	035220	104170			ERROR	170	:DATA INCORRECT
6080	035222	005037	003622		CLR	BITCNT	:INITIALIZE BIT COUNT
6081	035226	012737	063102	003170	MOV	#EM234, EMW	:LOAD ERROR MESSAGE
6082	035234	012703	067010		MOV	#HEAD5, R3	:LOAD ADDRESS OF DATA
6083	035240	012700	000003		MOV	#3, R0	:LOAD NUMBER WORDS IN HEADER
6084	035244	012304			105: MOV	(R3)+, R4	:GET NEXT WORD
6085	035246	012701	000020		MOV	#16, R1	:LOAD BIT COUNT
6086	035252	013737	003616	003620	125: MOV	M1.BIT, M2.BIT	:SHIFT BITS
6087	035260	013737	003614	003616	MOV	PR.BIT, M1.BIT	
6088	035266	013737	003612	003614	MOV	P1.BIT, PR.BIT	
6089	035274	006004			ROR	R4	:SHIFT IN NEXT BIT
6090	035276	103403			BCS	145	:CHECK IF ONE
6091	035300	005037	003612		CLR	P1.BIT	:ZERO
6092	035304	000403			BR	155	:CLOCK IN BIT
6093							
6094	035306	012737	000001	003612	145: MOV	#1, P1.BIT	:ONE
6095	035314	004737	045206		155: JSR	PC, WRTBIT	:WRITE BIT
6096	035320	104170			ERROR	170	:BIT INCORRECT
6097	035322	005237	003622		INC	BITCNT	:INCREMENT BIT COUNT
6098	035326	005301			DEC	R1	:CHECK IF WORD FINISHED
6099	035330	001350			BNE	125	:NO, CONTINUE WITH NEXT BIT
6100	035332	005300			DEC	R0	:CHECK IF HEADER COMPLETE
6101	035334	001343			BNE	105	:NO, GET NEXT WORD
6102	035336	012701	000020		MOV	#16, R1	:LOAD BIT COUNT FOR NEXT WORD

6103	035342	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6104	035350	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6105	035356	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6106	035364	005037	003612			CLR	P1.BIT	
6107	035370	004737	045206			JSR	PC.WRTBIT	;WRITE ZERO
6108	035374	104170				ERROR	170	;BIT INCORRECT
6109	035376	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
6110	035402	005301				DEC	R1	;CHECK IF FINISHED
6111	035404	001356				BNE	18\$;NO, CLOCK NEXT BIT
6112	035406	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	;SIMULATE INDEX
6113	035414	012700	000004			MOV	#4,R0	
6114	035420	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6115	035426	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6116	035434	005300				DEC	R0	
6117	035436	001370				BNE	20\$	
6118	035440	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6119	035446	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;GET MAINT REG 1
6120	035454	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	;LOAD EXPECTED MR1
6121	035462	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK MR1 CORRECT (WRITE GATE RESET)
6122	035470	001401				BEQ	25\$;YES, CHECK IF READY SET
6123	035472	104173				ERROR	173	;MAINT REG 1 INCORRECT
6124	035474	012700	000010		25\$:	MOV	#8,R0	;FINISH COMMAND
6125	035500	012762	000440	000026	26\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6126	035506	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6127	035514	005300				DEC	R0	
6128	035516	001370				BNE	26\$	
6129	035520	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
6130	035526	012737	000226	003500		MOV	#RDY!WRHEAD&<↑C<GO>>,E.CS1	;LOAD EXPECTED CS1
6131	035534	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
6132	035542	001401				BEQ	TST43	;YES, GO ON TO NEXT TEST
6133	035544	104174				ERROR	174	;CS1 INCORRECT

 *TEST 43 WRITE LOOPBACK (PART 6)

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

155555
 066666
 155555

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
 CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

 *TST43: SCOPE

6153	035546	000004			TST43:	SCOPE		
6154	035550	012737	000144	001200		MOV	#100,\$TIMES	;DO 100. ITERATIONS
6155	035556	013702	001270			MOV	\$BASE,R2	;LOAD RK611 BASE
6156	035562	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
6157	035570	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
6158	035576	012762	067016	000004		MOV	#HEAD6,RKBA(R2)	;ISSUE WRITE HEADER

B10

6159	035604	012762	177775	000002		MOV	#-3, RKWC(R2)	
6160	035612	012762	000027	000000		MOV	#WHEAD, RKCS1(R2)	
6161	035620	012700	000312			MOV	#50, #4+2, R0	;ISSUE ENOUGH CLOCKS UNTIL ;READY FOR INDEX PULSE
6162								
6163	035624	012762	000440	000026	15:	MOV	#DMD!MCLK, RKMR1(R2)	
6164	035632	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6165	035640	005300				DEC	R0	
6166	035642	001370				BNE	15	
6167	035644	012700	000004			MOV	#4, R0	;ISSUE INDEX PULSE
6168	035650	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6169	035656	012762	000640	000026	25:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
6170	035664	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6171	035672	005300				DEC	R0	
6172	035674	001370				BNE	25	
6173	035676	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6174	035704	012700	000010			MOV	#8, R0	;WAIT FOR WRITE GATE
6175	035710	012762	000440	000026	35:	MOV	#DMD!MCLK, RKMR1(R2)	
6176	035716	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6177	035724	005300				DEC	R0	
6178	035726	001370				BNE	35	
6179	035730	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	;SIMULATE SECTOR PULSE
6180	035736	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6181	035744	005037	003626			CLR	SECCNT	;INITIALIZE SECTOR COUNT
6182	035750	012737	063036	003170		MOV	#EM233, EMW	;LOAD ERROR MESSAGE
6183	035756	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT, E.MR1	;INITIALIZE EXPECTED ;MAINT REG 1
6184								
6185	035764	005037	003612			CLR	P1.BIT	;INITIALIZE BIT GENERATION
6186	035770	005037	003614			CLR	PR.BIT	
6187	035774	005037	003616			CLR	M1.BIT	
6188	036000	005037	003620			CLR	M2.BIT	
6189	036004	012700	000400			MOV	#256, R0	;SIMULATE SYNCH
6190	036010	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6191	036014	004737	045206		55:	JSR	PC, WRTBIT	;WRITE ONE BIT
6192	036020	104170				ERROR	170	;DATA INCORRECT
6193	036022	005237	003622			INC	BITCNT	
6194	036026	005300				DEC	R0	;CHECK IF READY FOR DATA
6195	036030	001371				BNE	55	;NO, GENERATE NEXT BIT
6196	036032	012737	000001	003612		MOV	#1, P1.BIT	;PUT IN SYNCH BIT
6197	036040	004737	045206			JSR	PC, WRTBIT	
6198	036044	104170				ERROR	170	;DATA INCORRECT
6199	036046	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6200	036052	012737	063102	003170		MOV	#EM234, EMW	;LOAD ERROR MESSAGE
6201	036060	012703	067016			MOV	#HEAD6, R3	;LOAD ADDRESS OF DATA
6202	036064	012700	000003			MOV	#3, R0	;LOAD NUMBER WORDS IN HEADER
6203	036070	012304			105:	MOV	(R3)+, R4	;GET NEXT WORD
6204	036072	012701	000020			MOV	#16, R1	;LOAD BIT COUNT
6205	036076	013737	003616	003620	125:	MOV	M1.BIT, M2.BIT	;SHIFT BITS
6206	036104	013737	003614	003616		MOV	PR.BIT, M1.BIT	
6207	036112	013737	003612	003614		MOV	P1.BIT, PR.BIT	
6208	036120	006004				ROR	R4	;SHIFT IN NEXT BIT
6209	036122	103403				BCS	145	;CHECK IF ONE
6210	036124	005037	003612			CLR	P1.BIT	;ZERO
6211	036130	000403				BR	155	;CLOCK IN BIT
6212								
6213	036132	012737	000001	003612	145:	MOV	#1, P1.BIT	;ONE
6214	036140	004737	045206		155:	JSR	PC, WRTBIT	;WRITE BIT

6215	036144	104170				ERROR	170		;BIT INCORRECT
6216	036146	005237	003622			INC	BITCNT		;INCREMENT BIT COUNT
6217	036152	005301				DEC	R1		;CHECK IF WORD FINISHED
6218	036154	001350				BNE	12\$;NO, CONTINUE WITH NEXT BIT
6219	036156	005300				DEC	RO		;CHECK IF HEADER COMPLETE
6220	036160	001343				BNE	10\$;NO, GET NEXT WORD
6221	036162	012701	000020			MOV	#16, R1		;LOAD BIT COUNT FOR NEXT WORD
6222	036166	013737	003616	003620	18\$:	MOV	M1.BIT, M2.BIT		;SHIFT BITS
6223	036174	013737	003614	003616		MOV	PR.BIT, M1.BIT		
6224	036202	013737	003612	003614		MOV	P1.BIT, PR.BIT		
6225	036210	005037	003612			CLR	P1.BIT		
6226	036214	004737	045206			JSR	PC WRTBIT		;WRITE ZERO
6227	036220	104170				ERROR	170		;BIT INCORRECT
6228	036222	005237	003622			INC	BITCNT		;INCREMENT BIT COUNT
6229	036226	005301				DEC	R1		;CHECK IF FINISHED
6230	036230	001356				BNE	18\$;NO, CLOCK NEXT BIT
6231	036232	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		;SIMULATE INDEX
6232	036240	012700	000004			MOV	#4, RO		
6233	036244	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)		
6234	036252	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		
6235	036260	005300				DEC	RO		
6236	036262	001370				BNE	20\$		
6237	036264	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
6238	036272	016237	000026	003464		MOV	RKMR1(R2), T.MR1		;GET MAINT REG 1
6239	036300	012737	022040	003524		MOV	#MEMD!ECCW!DMD, E.MR1		;LOAD EXPECTED MR1
6240	036306	023737	003524	003464		CMP	E.MR1, T.MR1		;CHECK MR1 CORRECT (WRITE GATE RESET)
6241	036314	001401				BEQ	25\$;YES, CHECK IF READY SET
6242	036316	104173				ERROR	173		;MAINT REG 1 INCORRECT
6243	036320	012700	000010		25\$:	MOV	#8, RO		;FINISH COMMAND
6244	036324	012762	000440	000026	26\$:	MOV	#DMD!MCLK, RKMR1(R2)		
6245	036332	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
6246	036340	005300				DEC	RO		
6247	036342	001370				BNE	26\$		
6248	036344	016237	000000	003440		MOV	RKCS1(R2), T.CS1		;GET COMMAND AND STATUS REG 1
6249	036352	012737	000226	003500		MOV	#RDY!WRHEAD&<IC<GO>>, E.CS1		;LOAD EXPECTED CS1
6250	036360	023737	003500	003440		CMP	E.CS1, T.CS1		;CHECK IF CS1 CORRECT
6251	036366	001401				BEQ	TST44		;YES, GO ON TO NEXT TEST
6252	036370	104174				ERROR	174		;CS1 INCORRECT

*TEST 44 WRITE LOOPBACK (PART 7)
*

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

* 104210
* 104210
* 104210
*

* MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
*

6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270

```

6271
6272 036372 000004
6273 036374 012737 000144 001200
6274 036402 013702 001270
6275 036406 012762 100000 000000
6276 036414 012762 000040 000026
6277 036422 012762 067024 000004
6278 036430 012762 177775 000002
6279 036436 012762 000027 000000
6280 036444 012700 000312
6281
6282 036450 012762 000440 000026 15:
6283 036456 012762 000040 000026
6284 036464 005300
6285 036466 001370
6286 036470 012700 000004
6287 036474 012762 000240 000026
6288 036502 012762 000640 000026 25:
6289 036510 012762 000240 000026
6290 036516 005300
6291 036520 001370
6292 036522 012762 000040 000026
6293 036530 012700 000010
6294 036534 012762 000440 000026 35:
6295 036542 012762 000040 000026
6296 036550 005300
6297 036552 001370
6298 036554 012762 000140 000026
6299 036562 012762 000040 000026
6300 036570 005037 003626
6301 036574 012737 063036 003170
6302 036602 012737 062040 003524
6303
6304 036610 005037 003612
6305 036614 005037 003614
6306 036620 005037 003616
6307 036624 005037 003620
6308 036630 012700 000400
6309 036634 005037 003622
6310 036640 004737 045206 55:
6311 036644 104170
6312 036646 005237 003622
6313 036652 005300
6314 036654 001371
6315 036656 012737 000001 003612
6316 036664 004737 045206
6317 036670 104170
6318 036672 005037 003622
6319 036676 012737 063102 003170
6320 036704 012703 067024
6321 036710 012700 000003
6322 036714 012304 105:
6323 036716 012701 000020
6324 036722 013737 003616 003620 125:
6325 036730 013737 003614 003616
6326 036736 013737 003612 003614

```

```

*****
↑ST44: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD7, RKBA(R2) ; ISSUE WRITE HEADER
MOV #-3, RKWC(R2)
MOV #WRHEAD, RKCS1(R2)
MOV #50.*4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
15: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 15
MOV #4, R0 ; ISSUE INDEX PULSE
MOV #DMD!MIND, RKMR1(R2)
25: MOV #DMD!MIND!MCLK, RKMR1(R2)
MOV #DMD!MIND, RKMR1(R2)
DEC R0
BNE 25
MOV #DMD, RKMR1(R2)
MOV #8, R0 ; WAIT FOR WRITE GATE
35: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 35
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR SECCNT ; INITIALIZE SECTOR COUNT
MOV #EM233, EMW ; LOAD ERROR MESSAGE
MOV #DMD!MEND!ECCW!WRTGAT, E.MR1 ; INITIALIZE EXPECTED
; MAINT REG 1
; INITIALIZE BIT GENERATION
CLR P1.BIT
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256, R0 ; SIMULATE SYNCH
CLR BITCNT ; INITIALIZE BIT COUNT
55: JSR PC, WRTBIT ; WRITE ONE BIT
ERROR 170 ; DATA INCORRECT
INC BITCNT
DEC R0 ; CHECK IF READY FOR DATA
BNE 55 ; NO, GENERATE NEXT BIT
MOV #1, P1.BIT ; PUT IN SYNCH BIT
JSR PC, WRTBIT
ERROR 170 ; DATA INCORRECT
CLR BITCNT ; INITIALIZE BIT COUNT
MOV #EM234, EMW ; LOAD ERROR MESSAGE
MOV #HEAD7, R3 ; LOAD ADDRESS OF DATA
MOV #3, R0 ; LOAD NUMBER WORDS IN HEADER
105: MOV (R3)+, R4 ; GET NEXT WORD
MOV #16, R1 ; LOAD BIT COUNT
125: MOV M1.BIT, M2.BIT ; SHIFT BITS
MOV PR.BIT, M1.BIT
MOV P1.BIT, PR.BIT

```

E10

```

6327 036744 006004 ROR R4 ;SHIFT IN NEXT BIT
6328 036746 103403 BCS 14$ ;CHECK IF ONE
6329 036750 005037 003612 CLR P1.BIT ;ZERO
6330 036754 000403 BR 15$ ;CLOCK IN BIT
6331
6332 036756 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
6333 036764 004737 045206 15$: JSR PC,WRTBIT ;WRITE BIT
6334 036770 104170 ERROR 170 ;BIT INCORRECT
6335 036772 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6336 036776 005301 DEC R1 ;CHECK IF WORD FINISHED
6337 037000 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
6338 037002 005300 DEC R0 ;CHECK IF HEADER COMPLETE
6339 037004 001343 BNE 10$ ;NO, GET NEXT WORD
6340 037006 012701 000020 MOV #16,R1 ;LOAD BIT COUNT FOR NEXT WORD
6341 037012 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6342 037020 013737 003614 00361E MOV PR.BIT,M1.BIT
6343 037026 013737 003612 003614 MOV P1.BIT,PR.BIT
6344 037034 005037 003612 CLR P1.BIT
6345 037040 004737 045206 JSR PC,WRTBIT ;WRITE ZERO
6346 037044 104170 ERROR 170 ;BIT INCORRECT
6347 037046 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6348 037052 005301 DEC R1 ;CHECK IF FINISHED
6349 037054 001356 BNE 18$ ;NO, CLOCK NEXT BIT
6350 037056 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
6351 037064 012700 000004 MOV #4,R0
6352 037070 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6353 037076 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6354 037104 005300 DEC R0
6355 037106 001370 BNE 20$
6356 037110 012762 000040 000026 MOV #DMD,RKMR1(R2)
6357 037116 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6358 037124 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6359 037132 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
6360 037140 001401 BEQ 25$ ;YES, CHECK IF READY SET
6361 037142 104173 ERROR 173 ;MAINT REG 1 INCORRECT
6362 037144 012700 000010 25$: MOV #8,R0 ;FINISH COMMAND
6363 037150 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6364 037156 012762 000040 000026 MOV #DMD,RKMR1(R2)
6365 037164 005300 DEC R0
6366 037166 001370 BNE 26$
6367 037170 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6368 037176 012737 000226 003500 MOV #RDY!WRHEAD&<FC<GO>>,E.CS1 ;LOAD EXPECTED CS1
6369 037204 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6370 037212 001401 BEQ TST45 ;YES, GO ON TO NEXT TEST
6371 037214 104174 ERROR 174 ;CS1 INCORRECT

```

```

6372
6373 *****
6374 *TEST 45 WRITE TWO HEADERS
6375 *
6376 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
6377 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
6378 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
6379 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
6380 * INDEX PULSE, SECTOR PULSE, THREE WORD HEADER
6381 * CONSISTING OF THE FOLLOWING DATA:
6382 *

```

F10

```

6383          : *          177777
6384          : *          000000
6385          : *          177777
6386          : *
6387          : *          FOLLOW THAT BY A SECTOR PULSE AND ONE THREE WORD
6388          : *          HEADER CONSISTING OF THE FOLLOWING DATA:
6389          : *
6390          : *          000000
6391          : *          177777
6392          : *          000000
6393          : *
6394          : *          SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP.
6395          : *          CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMENSATION.
6396          : *
6397          : * *****
6398          : * T45: SCOPE
6399          : * MOV      #100, $TIMES      ;; DO 100. ITERATIONS
6400          : * MOV      $BASE, R2      ;; LOAD RK611 BASE
6401          : * MOV      #CCLR, RKCS1(R2) ;; CLEAR RK611
6402          : * MOV      #DMD, RKMR1(R2) ;; PUT RK611 TO DIAGNOSTIC MODE
6403          : * MOV      #HEAD1, RKBA(R2) ;; ISSUE WRITE HEADER
6404          : * MOV      #HEAD1, R3
6405          : * MOV      #-6, RKWC(R2)
6406          : * MOV      #WRHEAD, RKCS1(R2)
6407          : * MOV      #50, *4+2, R0      ;; ISSUE ENOUGH CLOCKS UNTIL
6408          : *          ;; READY FOR INDEX PULSE
6409          : * MOV      #DMD!MCLK, RKMR1(R2)
6410          : * MOV      #DMD, RKMR1(R2)
6411          : * DEC      R0
6412          : * BNE     1$
6413          : * MOV      #4, R0      ;; ISSUE INDEX PULSE
6414          : * MOV      #DMD!MIND, RKMR1(R2)
6415          : * MOV      #DMD!MIND!MCLK, RKMR1(R2)
6416          : * MOV      #DMD!MIND, RKMR1(R2)
6417          : * DEC      R0
6418          : * BNE     2$
6419          : * CLR      SECCNT      ;; CLEAR SECTOR COUNT
6420          : * MOV      #DMD, RKMR1(R2)
6421          : * MOV      #2, R5      ;; LOAD NUMBER OF HEADERS
6422          : * MOV      #8, R0      ;; WAIT FOR WRITE GATE
6423          : * MOV      #DMD!MCLK, RKMR1(R2)
6424          : * MOV      #DMD, RKMR1(R2)
6425          : * DEC      R0
6426          : * BNE     3$
6427          : * MOV      #DMD!MSP, RKMR1(R2) ;; SIMULATE SECTOR PULSE
6428          : * MOV      #DMD, RKMR1(R2)
6429          : * MOV      #EM233, EMW      ;; LOAD ERROR MESSAGE
6430          : * MOV      #DMD!MEWD!ECCW!WRTGAT, E.MR1 ;; INITIALIZE EXPECTED
6431          : *          ;; MAINT REG 1
6432          : * CLR      P1.BIT
6433          : * CLR      PR.BIT
6434          : * CLR      M1.BIT
6435          : * CLR      M2.BIT
6436          : * MOV      #256, R0      ;; SIMULATE SYNCH
6437          : * CLR      BITCNT      ;; INITIALIZE BIT COUNT
6438          : * JSR     PC, WRTBIT    ;; WRITE ONE BIT
    
```

G10

6439	037500	104170				ERROR	170		; DATA INCORRECT
6440	037502	005237	003622			INC	BITCNT		
6441	037506	005300				DEC	R0		; CHECK IF READY FOR DATA
6442	037510	001371				BNE	5\$; NO GENERATE NEXT BIT
6443	037512	012737	000001	003612		MOV	#1, P1.BIT		; PUT IN SYNCH BIT
6444	037520	004737	045206			JSR	PC, WRTBIT		
6445	037524	104170				ERROR	170		; DATA INCORRECT
6446	037526	005037	003622			CLR	BITCNT		; INITIALIZE BIT COUNT
6447	037532	012737	063102	003170		MOV	#EM234, EMW		; LOAD ERROR MESSAGE
6448	037540	012700	000003			MOV	#3, R0		; LOAD NUMBER OF WORDS IN HEADER
6449	037544	012304			10\$:	MOV	(R3)+, R4		; GET NEXT WORD
6450	037546	012701	000020			MOV	#16, R1		; LOAD BIT COUNT
6451	037552	013737	003616	003620	12\$:	MOV	M1.BIT, M2.BIT		; SHIFT BITS
6452	037560	013737	003614	003616		MOV	PR.BIT, M1.BIT		
6453	037566	013737	003612	003614		MOV	P1.BIT, PR.BIT		
6454	037574	006004				ROR	R4		; SHIFT IN NEXT BIT
6455	037576	103403				BCS	14\$; CHECK IF ONE
6456	037600	005037	003612			CLR	P1.BIT		; ZERO
6457	037604	000403				BR	15\$; CLOCK IN BIT
6458									
6459	037606	012737	000001	003612	14\$:	MOV	#1, P1.BIT		; ONE
6460	037614	004737	045206		15\$:	JSR	PC, WRTBIT		; WRITE BIT
6461	037620	104170				ERROR	170		; BIT INCORRECT
6462	037622	005237	003622			INC	BITCNT		; INCREMENT BIT COUNT
6463	037626	005301				DEC	R1		; CHECK IF WORD FINISHED
6464	037630	001350				BNE	12\$; NO, CONTINUE
6465	037632	005300				DEC	R0		; CHECK IF HEADER COMPLETE
6466	037634	001343				BNE	10\$; NO, GET NEXT WORD
6467	037636	012701	000020			MOV	#16, R1		; LOAD BIT COUNT FOR NEXT WORD
6468	037642	013737	003616	003620	18\$:	MOV	M1.BIT, M2.BIT		; SHIFT BITS
6469	037650	013737	003614	003616		MOV	PR.BIT, M1.BIT		
6470	037656	013737	003612	003614		MOV	P1.BIT, PR.BIT		
6471	037664	005037	003612			CLR	P1.BIT		
6472	037670	004737	045206			JSR	PC, WRTBIT		; WRITE ZERO
6473	037674	104170				ERROR	170		; BIT INCORRECT
6474	037676	005237	003622			INC	BITCNT		; INCREMENT
6475	037702	005301				DEC	R1		; CHECK IF READY FOR NEXT HEADER
6476	037704	001356				BNE	18\$; NO, CONTINUE
6477	037706	005237	003626			INC	SECCNT		; INCREMENT
6478	037712	005305				DEC	R5		; CHECK IF SECOND HEADER WRITTEN
6479	037714	001237				BNE	4\$; NO, DO SECOND HEADER
6480	037716	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		; SIMULATE INDEX PULSE
6481	037724	012700	000004			MOV	#4, R0		
6482	037730	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)		
6483	037736	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)		
6484	037744	005300				DEC	R0		
6485	037746	001370				BNE	20\$		
6486	037750	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
6487	037756	016237	000026	003464		MOV	RKMR1(R2), T.MR1		; GET MAINT REG 1
6488	037764	012737	022040	003524		MOV	#MEWD!ECCW!DMD, E.MR1		; LOAD EXPECTED MR1
6489	037772	023737	003524	003464		CMP	E.MR1, T.MR1		; CHECK MR1 CORRECT (WRITE GATE RESET
6490	040000	001401				BEQ	25\$; YES, CHECK IF READY SET
6491	040002	104173				ERROR	173		; MAINT REG 1 INCORRECT
6492	040004	012700	000010		25\$:	MOV	#8, R0		; FINISH COMMAND
6493	040010	012762	000440	000026	26\$:	MOV	#DMD!MCLK, RKMR1(R2)		
6494	040016	012762	000040	000026		MOV	#DMD, RKMR1(R2)		

H10

6495	040024	005300		
6496	040026	001370		
6497	040030	016237	000000	003440
6498	040036	012737	000226	003500
6499	040044	023737	003500	003440
6500	040052	001401		
6501	040054	104174		
6502				
6503				
6504				
6505				
6506				
6507				
6508				
6509				
6510				
6511				
6512				
6513				
6514				
6515				
6516				
6517				
6518				
6519				
6520				
6521				
6522				
6523	040056	000004		
6524	040060	012737	000144	001200
6525	040066	013702	001270	
6526	040072	012762	100000	000000
6527	040100	012762	000040	000026
6528	040106	012762	066766	000004
6529	040114	012703	066766	
6530	040120	012762	177772	000002
6531	040126	012762	000027	000000
6532	040134	012700	000312	
6533				
6534	040140	012762	000440	000026 1\$:
6535	040146	012762	000040	000026
6536	040154	005300		
6537	040156	001370		
6538	040160	012700	000004	
6539	040164	012762	000240	000026
6540	040172	012762	000640	000026 2\$:
6541	040200	012762	000240	000026
6542	040206	005300		
6543	040210	001370		
6544	040212	012762	000040	000026
6545	040220	012700	000010	
6546	040224	012762	000440	000026 3\$:
6547	040232	012762	000040	000026
6548	040240	005300		
6549	040242	001370		
6550	040244	005037	003626	

```

DEC      R0
BNE      26$
MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
MOV      #RDY!WRHEAD8<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
CMP      E.CS1,T.CS1    ;CHECK IF CS1 CORRECT
BEQ      TST46          ;;YES, GO ON TO NEXT TEST
ERROR    174

```

```

*****
*TEST 46      DATA FIELD FILLING ON WRITE HEADER
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND
* SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE
* FOLLOWING DATA:
*
*          125252
*          052525
*          125252
*          052525
*          125252
*          052525
*
* MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
* ECC FIELD ARE WRITTEN CORRECTLY.
*
*****

```

```

TST46:  SCOPE
MOV      #100,$TIMES      ;;DO 100. ITERATIONS
MOV      $BASE,R2        ;;LOAD RK611 BASE
MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
MOV      #DMD,RKMR1(R2)  ;PUT RK611 IN DIAGNOSTIC MODE
MOV      #HEAD3,RKBA(R2) ;ISSUE READ HEADER
MOV      #HEAD3,R3
MOV      #-2*3,RKWC(R2)
MOV      #WRHEAD,RKCS1(R2)
MOV      #50.*4+2,R0     ;ISSUE CLOCKS UNTIL READY
                          ; FOR INDEX PULSE
1$:      MOV      #DMD!MCLK,RKMR1(R2)
MOV      #DMD,RKMR1(R2)
DEC      R0
BNE      1$
MOV      #4,R0          ;ISSUE INDEX PULSE
MOV      #DMD!MIND,RKMR1(R2)
2$:      MOV      #DMD!MIND!MCLK,RKMR1(R2)
MOV      #DMD!MIND,RKMR1(R2)
DEC      R0
BNE      2$
MOV      #DMD,RKMR1(R2)
MOV      #8,R0         ;WAIT FOR WRITE GATE
3$:      MOV      #DMD!MCLK,RKMR1(R2)
MOV      #DMD,RKMR1(R2)
DEC      R0
BNE      3$
CLR      SECCNT        ;CLEAR SECTOR COUNT

```

6551	040250	012705	000002			MOV	#2,R5	:LOAD NUMBER OF HEADERS
6552	040254	012762	000140	000026	4\$:	MOV	#0MD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
6553	040262	012762	000040	000026		MOV	#0MD,RKMR1(R2)	
6554	040270	012737	063036	003170		MOV	#EM233,EMW	:LOAD ERROR MESSAGE
6555	040276	012737	062040	003524		MOV	#0MD!MEWD!ECCW!WRTGAT.E.MR1	:INITIALIZE EXPECTED
6556								: MAINT REG 1
6557	040304	005037	003612			CLR	P1.BIT	
6558	040310	005037	003614			CLR	PR.BIT	
6559	040314	005037	003616			CLR	M1.BIT	
6560	040320	005037	003620			CLR	M2.BIT	
6561	040324	012700	000400			MOV	#255,R0	:SIMULATE SYNCH
6562	040330	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
6563	040334	004737	045206		5\$:	JSR	PC,WRTBIT	:WRITE ONE BIT
6564	040340	104170				ERROR	170	:DATA INCORRECT
6565	040342	005237	003622			INC	BITCNT	
6566	040346	005300				DEC	R0	:CHECK IF READY FOR DATA
6567	040350	001371				BNE	5\$:NO, GENERATE NEXT BIT
6568	040352	012737	000001	003612		MOV	#1,P1.BIT	:PUT IN SYNCH BIT
6569	040360	004737	045206			JSR	PC,WRTBIT	
6570	040364	104170				ERROR	170	:DATA INCORRECT
6571	040366	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
6572	040372	012737	063102	003170		MOV	#EM234,EMW	:LOAD ERROR MESSAGE
6573	040400	012700	000003			MOV	#3,R0	:LOAD NUMBER OF WORDS IN HEADER
6574	040404	012304			10\$:	MOV	(R3)+,R4	:GET NEXT WORD
6575	040406	012701	000020			MOV	#16,R1	:LOAD BIT COUNT
6576	040412	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6577	040420	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6578	040426	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6579	040434	006004				ROR	R4	:SHIFT IN NEXT BIT
6580	040436	103403				BCS	14\$:CHECK IF ONE
6581	040440	005037	003612			CLR	P1.BIT	:ZERO
6582	040444	000403				BR	15\$:CLOCK IN BIT
6583								
6584	040446	012737	000001	003612	14\$:	MOV	#1,P1.BIT	:ONE
6585	040454	004737	045206		15\$:	JSR	PC,WRTBIT	:WRITE BIT
6586	040460	104170				ERROR	170	:BIT INCORRECT
6587	040462	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6588	040466	005301				DEC	R1	:CHECK IF WORD FINISHED
6589	040470	001350				BNE	12\$:NO, CONTINUE
6590	040472	005300				DEC	R0	:CHECK IF HEADER COMPLETE
6591	040474	001343				BNE	10\$:NO, GET NEXT WORD
6592	040476	012737	063334	003170		MOV	#EM237,EMW	:LOAD ERROR MESSAGE
6593	040504	012701	000477			MOV	#64.+255.,R1	:LOAD COUNT
6594	040510	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6595	040516	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6596	040524	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6597	040532	005037	003612			CLR	P1.BIT	
6598	040536	004737	045206			JSR	PC,WRTBIT	:WRITE ZERO
6599	040542	104170				ERROR	170	:BIT INCORRECT
6600	040544	005301				DEC	R1	:CHECK IF READY FOR DATA
6601	040546	001360				BNE	18\$:NO, CONTINUE
6602	040550	012737	000001	003612		MOV	#1,P1.BIT	:SET SYNCH BIT
6603	040556	004737	045206			JSR	PC,WRTBIT	:WRITE SYNCH BIT
6604	040562	104170				ERROR	170	:SYNCH BIT INCORRECT
6605	040564	012737	063402	003170		MOV	#EM238,EMW	:LOAD ERROR MESSAGE
6606	040572	005037	003622			CLR	BITCNT	:CLEAR BIT COUNT

J10

```

6607 040576 012701 007777      MOV      #256, #16, -1, R1 ;LOAD COUNT
6608 040602 013737 003616 003620 20$:  MOV      M1.BIT, M2.BIT ;SHIFT BITS
6609 040610 013737 003614 003616      MOV      PR.BIT, M1.BIT
6610 040616 013737 003612 003614      MOV      P1.BIT, PR.BIT
6611 040624 005037 003612      CLR      P1.BIT
6612 040630 004737 045206      JSR      PC, WRTBIT ;WRITE ZEROS
6613 040634 104170      ERROR   170 ;BIT INCORRECT
6614 040636 005237 003622      INC      BITCNT ;INCREMENT BIT COUNT
6615 040642 005301      DEC      R1 ;CHECK IF READY FOR ECC
6616 040644 001356      BNE     20$ ;NO, CONTINUE
6617 040646 012701 000040      MOV      #32, R1 ;LOAD COUNT
6618 040652 042737 020000 003524      BIC     #ECCW, E.MR1 ;RESET ECCW BIT
6619 040660 004737 045206 21$:  JSR      PC, WRTBIT ;WRITE ECC FIELD
6620 040664 104170      ERROR   170 ;BIT INCORRECT
6621 040666 005237 003622      INC      BITCNT ;INCREMENT COUNT
6622 040672 005301      DEC      R1 ;CHECK IF READY FOR POST AMBLE
6623 040674 001371      BNE     21$ ;NO CONTINUE
6624 040676 052737 020000 003524      BIS     #ECCW, E.MR1 ;SET ECCW
6625 040704 012701 000012      MOV      #10, R1 ;CHECK IF READY FOR NEXT HEADER
6626 040710 004737 045206 22$:  JSR      PC, WRTBIT ;WRITE POSTAMBLE
6627 040714 104170      ERROR   170 ;BIT INCORRECT
6628 040716 005237 003622      INC      BITCNT ;INCREMENT COUNT
6629 040722 005301      DEC      R1 ;CHECK IF READY FOR NEXT HEADER
6630 040724 001371      BNE     22$ ;NO, COMPLETE POSTAMBLE
6631 040726 005237 003626      INC      SECCNT ;INCREMENT COUNT
6632 040732 005305      DEC      R5 ;CHECK IF ALL HEADERS RECEIVED
6633 040734 001402      BEQ     23$ ;YES, SIMULATE INDEX
6634 040736 000137 040254      JMP     4$ ;NO GET NEXT HEADER
6635
6636 040742 012762 000240 000026 23$:  MOV      #DMD!MIND, RKMRI(R2) ;SIMULATE INDEX PULSE
6637 040750 012700 000004      MOV      #4, R0
6638 040754 012762 000640 000026 25$:  MOV      #DMD!MIND!MCLK, RKMRI(R2)
6639 040762 012762 000240 000026      MOV      #DMD!MIND, RKMRI(R2)
6640 040770 005300      DEC      R0
6641 040772 001370      BNE     25$
6642 040774 012762 000040 000026      MOV      #DMD, RKMRI(R2)
6643 041002 016237 000026 003464      MOV      RKMRI(R2), T.MR1 ;GET MAINT REG 1
6644 041010 012737 022040 003524      MOV      #MEWD!ECCW!DMD, E.MR1 ;LOAD EXPECTED MR1
6645 041016 023737 003524 003464      CMP     E.MR1, T.MR1 ;CHECK IF MR1 CORRECT
6646                                     ; (WRITE GATE RESET)
6647                                     ; YES, CHECK IF READY SET
6647 041024 001401      BEQ     23$ ;YES, CHECK IF READY SET
6648 041026 104173      ERROR   173 ;MAINTENANCE REG 1 INCORRECT
6649 041030 012700 000010 28$:  MOV      #8, R0 ;FINISH COMMAND
6650 041034 012762 000440 000026 29$:  MOV      #DMD!MCLK, RKMRI(R2)
6651 041042 012762 000040 000026      MOV      #DMD, RKMRI(R2)
6652 041050 005300      DEC      R0
6653 041052 001370      BNE     29$
6654 041054 016237 000000 003440      MOV      RKCSI(R2), T.CS1 ;GET COMMAND AND STATUS REG 1
6655 041062 012737 000226 003500      MOV      #RDY!WRHEAD&(C<GO>>, E.CS1 ;LOAD EXPECTED CSI
6656 041070 023737 003500 003440      CMP     E.CS1, T.CS1 ;CHECK IF CSI CORRECT
6657 041076 001401      BEQ     15147 ;YES, GO TO NEXT TEST
6658 041100 104174      ERROR   174 ;CSI INCORRECT
6659
6660 ;*****
6661 ;*TEST 47 WRITE HEADER FOR 26 SECTORS
6662 ;*

```

K10

```

6663      * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
6664      * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
6665      * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIFYING
6666      * 66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTLY.
6667      *
6668      * *****
6669      * ST47: SCOPE
6670      * MOV      #10,STIMES      ;;DO 10 ITERATIONS
6671      * MOV      $BASE,R2      ;;LOAD RK611 BASE
6672      * MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
6673      * MOV      #DMD,RKMR1(R2) ;;PUT RK611 IN DIAGNOSTIC MODE
6674      * MOV      #NPRBUF,RKBA(R2) ;;ISSUE READ HEADER
6675      * MOV      #NPRBUF,R3
6676      * MOV      #-22,#3,RKWC(R2)
6677      * MOV      #WHEAD,RKCS1(R2)
6678      * MOV      #50,#4+2,R0      ;;ISSUE CLOCKS UNTIL READY
6679      *                               ;;FOR INDEX PULSE
6680      * MOV      #DMD!MCLK,RKMR1(R2) 15:
6681      * MOV      #DMD,RKMR1(R2)
6682      * DEC      R0
6683      * BNE     1$
6684      * MOV      #4,R0      ;;ISSUE INDEX PULSE
6685      * MOV      #DMD!MIND,RKMR1(R2)
6686      * MOV      #DMD!MIND!MCLK,RKMR1(R2) 25:
6687      * MOV      #DMD!MIND,RKMR1(R2)
6688      * DEC      R0
6689      * BNE     2$
6690      * MOV      #DMD,RKMR1(R2)
6691      * MOV      #8,R0      ;;WAIT FOR WRITE GATE
6692      * MOV      #DMD!MCLK,RKMR1(R2) 35:
6693      * MOV      #DMD,RKMR1(R2)
6694      * DEC      R0
6695      * BNE     3$
6696      * CLR      SECCNT      ;;CLEAR SECTOR COUNT
6697      * MOV      #22,R5      ;;LOAD NUMBER OF HEADERS
6698      * MOV      #DMD!MSP,RKMR1(R2) ;;SIMULATE SECTOR PULSE
6699      * MOV      #DMD,RKMR1(R2)
6700      * MOV      #EM233,EMW      ;;LOAD ERROR MESSAGE
6701      * MOV      #DMD!MEND!ECCW!WRTGAT,E.MR1 ;;INITIALIZE EXPECTED
6702      *                               ;;MAINT REG 1
6703      * CLR      P1.BIT
6704      * CLR      PR.BIT
6705      * CLR      M1.BIT
6706      * CLR      M2.BIT
6707      * MOV      #256,R0      ;;SIMULATE SYNCH
6708      * CLR      BITCNT      ;;INITIALIZE BIT COUNT
6709      * JSR     PC,WRTBIT      ;;WRITE ONE BIT
6710      * ERROR   170      ;;DATA INCORRECT
6711      * INC      BITCNT
6712      * DEC      R0      ;;CHECK IF READY FOR DATA
6713      * BNE     5$      ;;NO, GENERATE NEXT BIT
6714      * MOV      #1,P1.BIT      ;;PUT IN SYNCH BIT
6715      * JSR     PC,WRTBIT
6716      * ERROR   170      ;;DATA INCORRECT
6717      * CLR      BITCNT      ;;INITIALIZE BIT COUNT
6718      * MOV      #EM234,EMW      ;;LOAD ERROR MESSAGE
    
```

L10

6719	041424	012700	000003			MOV	#3,R0	:LOAD NUMBER OF WORDS IN HEADER
6720	041430	012304		10\$:		MOV	(R3)+,R4	:GET NEXT WORD
6721	041432	012701	000020			MOV	#16,R1	:LOAD BIT COUNT
6722	041436	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6723	041444	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6724	041452	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6725	041460	006004				ROR	R4	:SHIFT IN NEXT BIT
6726	041462	103403				BCS	14\$:CHECK IF ONE
6727	041464	005037	003612			CLR	P1.BIT	:ZERO
6728	041470	000403				BR	15\$:CLOCK IN BIT
6729								
6730	041472	012737	000001	003612	14\$:	MOV	#1,P1.BIT	:ONE
6731	041500	004737	045206		15\$:	JSR	PC,WRTBIT	:WRITE BIT
6732	041504	104170				ERROR	170	:BIT INCORRECT
6733	041506	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6734	041512	005301				DEC	R1	:CHECK IF WORD FINISHED
6735	041514	001350				BNE	12\$:NO, CONTINUE
6736	041516	005300				DEC	R0	:CHECK IF HEADER COMPLETE
6737	041520	001343				BNE	10\$:NO, GET NEXT WORD
6738	041522	012737	063334	003170		MOV	#EM237,EMW	:LOAD ERROR MESSAGE
6739	041530	012701	000477			MOV	#64,+255,R1	:LOAD COUNT
6740	041534	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6741	041542	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6742	041550	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6743	041556	005037	003612			CLR	P1.BIT	
6744	041562	004737	045206			JSR	PC,WRTBIT	:WRITE ZERO
6745	041566	104170				ERROR	170	:BIT INCORRECT
6746	041570	005301				DEC	R1	:CHECK IF READY FOR DATA
6747	041572	001360				BNE	18\$:NO, CONTINUE
6748	041574	012737	000001	003612		MOV	#1,P1.BIT	:SET SYNCH BIT
6749	041602	004737	045206			JSR	PC,WRTBIT	:WRITE SYNCH BIT
6750	041606	104170				ERROR	170	:SYNCH BIT INCORRECT
6751	041610	012737	063402	003170		MOV	#EM238,EMW	:LOAD ERROR MESSAGE
6752	041616	005037	003622			CLR	BITCNT	:CLEAR BIT COUNT
6753	041622	012701	007777			MOV	#256,#16,-1,R1	:LOAD COUNT
6754	041626	013737	003616	003620	20\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6755	041634	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6756	041642	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6757	041650	005037	003612			CLR	P1.BIT	
6758	041654	004737	045206			JSR	PC,WRTBIT	:WRITE ZEROS
6759	041660	104170				ERROR	170	:BIT INCORRECT
6760	041662	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6761	041666	005301				DEC	R1	:CHECK IF READY FOR ECC
6762	041670	001356				BNE	20\$:NO, CONTINUE
6763	041672	012701	000040			MOV	#32,R1	:LOAD COUNT
6764	041676	042737	020000	003524		BIC	#ECCW,E.MR1	:RESET ECCW BIT
6765	041704	004737	045206		21\$:	JSR	PC,WRTBIT	:WRITE ECC FIELD
6766	041710	104170				ERROR	170	:BIT INCORRECT
6767	041712	005237	003622			INC	BITCNT	:INCREMENT COUNT
6768	041716	005301				DEC	R1	:CHECK IF READY FOR POST AMBLE
6769	041720	001371				BNE	21\$:NO CONTINUE
6770	041722	052737	020000	003524		BIS	#ECCW,E.MR1	:SET ECCW
6771	041730	012701	000012			MOV	#10,R1	:CHECK IF READY FOR NEXT HEADER
6772	041734	004737	045206		22\$:	JSR	PC,WRTBIT	:WRITE POSTAMBLE
6773	041740	104170				ERROR	170	:BIT INCORRECT
6774	041742	005237	003622			INC	BITCNT	:INCREMENT COUNT

M10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 128
 DZR6CB.P11 28-JAN-77 14:04 T47 WRITE HEADER FOR 26 SECTORS

SEQ 0129

6775	041746	005301				DEC	R1	:CHECK IF READY FOR NEXT HEADER
6776	041750	001371				BNE	22\$:NO COMPLETE POSTAMBLE
6777	041752	005237	003626			INC	SECCNT	:INCREMENT COUNT
6778	041756	005305				DEC	R5	:CHECK IF ALL HEADERS RECEIVED
6779	041760	001402				BEQ	23\$:YES, SIMULATE INDEX
6780	041762	000137	041300			JMP	4\$:NO GET NEXT HEADER
6781								
6782	041766	012762	000240	000026	23\$:	MOV	#DMD!MIND,RKMR1(R2)	:SIMULATE INDEX PULSE
6783	041774	012700	000004			MOV	#4,R0	
6784	042000	012762	000640	000026	25\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6785	042006	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6786	042014	005300				DEC	R0	
6787	042016	001370				BNE	25\$	
6788	042020	012762	000040	000026		MOV	#C'D,RKMR1(R2)	
6789	042026	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:GET MAINT REG 1
6790	042034	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	:LOAD EXPECTED MR1
6791	043042	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MR1 CORRECT
6792								: (WRITE GATE RESET)
6793	042050	001401				BEQ	28\$:YES, CHECK IF READY SET
6794	042052	104173				ERROR	173	:MAINTENANCE REG 1 INCORRECT
6795	042054	012700	000010		28\$:	MOV	#8,R0	:FINISH COMMAND
6796	042060	012762	000440	000026	29\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6797	042066	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6798	042074	005300				DEC	R0	
6799	042076	001370				BNE	29\$	
6800	042100	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET COMMAND AND STATUS REG 1
6801	042106	012737	000226	003500		MOV	#RDY!WRHEADS<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
6802	042114	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK IF CS1 CORRECT
6803	042122	001401				BEQ	↑ST50	:YES, GO TO NEXT TEST
6804	042124	104174				ERROR	174	:CS1 INCORRECT

 *TEST 50 WRITE HEADER IN 24 SECTOR FORMAT

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER OF SIX WORDS IN 24 SECTOR FORMAT TO AN RK06, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK THROUGH THE SEEK AND DRIVE CLEAR MESSAGES, SIMULATE INDEX PULSE, SECTOR PULSE, 3 HEADER WORDS, SYNCH AND DATA, ANOTHER SECTOR PULSE, 3 HEADER WORDS, AND AN INDEX PULSE. CHECK DATA WRITTEN TO MAKE SURE ONLY LOW 16 BITS OF SILO ARE USED.

6819	042126	000004				↑ST50: SCOPE		
6820	042130	012737	000144	001200		MOV	#100.,STIMES	:DO 100. ITERATIONS
6821	042136	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
6822	042142	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
6823	042150	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 TO DIAGNOSTIC MODE
6824	042156	012762	066752	000004		MOV	#HEAD1,RKBA(R2)	:ISSUE WRITE HEADER
6825	042164	012703	066752			MOV	#HEAD1,R3	
6826	042170	012762	177772	000002		MOV	#-6,RKWC(R2)	
6827	042176	012762	010027	000000		MOV	#CFMT!WRHEAD,RKCS1(R2)	
6828	042204	012700	000312			MOV	#50.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL
6829								:READY FOR INDEX PULSE
6830	042210	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	

N10

6831	042216	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6832	042224	005300				DEC	R0	
6833	042226	001370				BNE	1\$	
6834	042230	012700	000004			MOV	#4, R0	:ISSUE INDEX PULSE
6835	042234	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6836	042242	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
6837	042250	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6838	042256	005300				DEC	R0	
6839	042260	001370				BNE	2\$	
6840	042262	005037	003626			CLR	SECCNT	:CLEAR SECTOR COUNT
6841	042266	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6842	042274	012705	000002			MOV	#2, R5	:LOAD NUMBER OF HEADERS
6843	042300	012700	000010			MOV	#8, R0	:WAIT FOR WRITE GATE
6844	042304	012762	000440	000026	3\$:	MOV	#DMD!MCLK, RKMR1(R2)	
6845	042312	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6846	042320	005300				DEC	R0	
6847	042322	001370				BNE	3\$	
6848	042324	012762	000140	000026	4\$:	MOV	#DMD!MSP, RKMR1(R2)	:SIMULATE SECTOR PULSE
6849	042332	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6850	042340	012737	063441	003170		MOV	#EM239, EMW	:LOAD ERROR MESSAGE
6851	042346	012737	062040	003524		MOV	#DMD!MEND!ECCW!WRTGAT.E.MR1	:INITIALIZE EXPECTED : MAINT REC 1
6852								
6853	042354	005037	003612			CLR	P1.BIT	
6854	042360	005037	003614			CLR	PR.BIT	
6855	042364	005037	003616			CLR	M1.BIT	
6856	042370	005037	003620			CLR	M2.BIT	
6857	042374	012700	000400			MOV	#256, R0	:SIMULATE SYNCH
6858	042400	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
6859	042404	004737	045206		5\$:	JSR	PC, WRTBIT	:WRITE ONE BIT
6860	042410	104170				ERROR	170	:DATA INCORRECT
6861	042412	005237	003622			INC	BITCNT	
6862	042416	005300				DEC	R0	:CHECK IF READY FOR DATA
6863	042420	001371				BNE	5\$:NO GENERATE NEXT BIT
6864	042422	012737	000001	003612		MOV	#1, P1.BIT	:PUT IN SYNCH BIT
6865	042430	004737	045206			JSR	PC, WRTBIT	
6866	042434	104170				ERROR	170	:DATA INCORRECT
6867	042436	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
6868	042442	012737	063533	003170		MOV	#EM240, EMW	:LOAD ERROR MESSAGE
6869	042450	012700	000003			MOV	#3, R0	:LOAD NUMBER OF WORDS IN HEADER
6870	042454	012304			10\$:	MOV	(R3)+, R4	:GET NEXT WORD
6871	042456	012701	000020			MOV	#16, R1	:LOAD BIT COUNT
6872	042462	013737	003616	003620	12\$:	MOV	M1.BIT, M2.BIT	:SHIFT BITS
6873	042470	013737	003614	003616		MOV	PR.BIT, M1.BIT	
6874	042476	013737	003612	003614		MOV	P1.BIT, PR.BIT	
6875	042504	006004				ROR	R4	:SHIFT IN NEXT BIT
6876	042506	103403				BCS	14\$:CHECK IF ONE
6877	042510	005037	003612			CLR	P1.BIT	:ZERO
6878	042514	000403				BR	15\$:CLOCK IN BIT
6879								
6880	042516	012737	000001	003612	14\$:	MOV	#1, P1.BIT	:ONE
6881	042524	004737	045206		15\$:	JSR	PC, WRTBIT	:WRITE BIT
6882	042530	104170				ERROR	170	:BIT INCORRECT
6883	042532	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6884	042536	005301				DEC	R1	:CHECK IF WORD FINISHED
6885	042540	001350				BNE	12\$:NO CONTINUE
6886	042542	005300				DEC	R0	:CHECK IF HEADER COMPLETE

B11

```

6887 042544 001343          BNE      10$          ;NO, GET NEXT WORD
6888 042546 012701 000020    MOV      #16, R1     ;LOAD BIT COUNT FOR NEXT WORD
6889 042552 013737 003616 003620 18$:    MOV      M1.BIT, M2.BIT ;SHIFT BITS
6890 042560 013737 003614 003616    MOV      PR.BIT, M1.BIT
6891 042566 013737 003612 003614    MOV      P1.BIT, PR.BIT
6892 042574 005037 003612          CLR      P1.BIT
6893 042600 004737 045206    JSR      PC, WRTBIT  ;WRITE ZERO
6894 042604 104170          ERROR    170         ;BIT INCORRECT
6895 042606 005237 003622    INC      BITCNT     ;INCREMENT
6896 042612 005301          DEC      R1         ;CHECK IF READY FOR NEXT HEADER
6897 042614 001356          BNE      18$        ;NO, CONTINUE
6898 042616 005237 003626    INC      SECCNT    ;INCREMENT
6899 042622 005305          DEC      R5         ;CHECK IF SECOND HEADER WRITTEN
6900 042624 001237          BNE      4$         ;NO, DO SECOND HEADER
6901 042626 012762 000240 000026    MOV      #DMD!MIND, RKMR1(R2) ;SIMULATE INDEX PULSE
6902 042634 012700 000004          MOV      #4, R0
6903 042640 012762 000640 000026 20$:    MOV      #DMD!MIND!MCLK, RKMR1(R2)
6904 042646 012762 000240 000026    MOV      #DMD!MIND, RKMR1(R2)
6905 042654 005300          DEC      R0
6906 042656 001370          BNE      20$
6907 042660 012762 000040 000026    MOV      #DMD, RKMR1(R2)
6908 042666 016237 000026 003464    MOV      RKMR1(R2), T.MR1 ;GET MAINT REG 1
6909 042674 012737 022040 003524    MOV      #MEWD!ECCW!DMD, E.MR1 ;LOAD EXPECTED MR1
6910 042702 023737 003524 003464    CMP      E.MR1, T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
6911 042710 001401          BEQ      25$        ;YES, CHECK IF READY SET
6912 042712 104173          ERROR    173         ;MAINT REG 1 INCORRECT
6913 042714 012700 000010 25$:    MOV      #8, R0     ;FINISH COMMAND
6914 042720 012762 000440 000026 26$:    MOV      #DMD!MCLK, RKMR1(R2)
6915 042726 012762 000040 000026    MOV      #DMD, RKMR1(R2)
6916 042734 005300          DEC      R0
6917 042736 001370          BNE      26$
6918 042740 016237 000000 003440    MOV      RKCS1(R2), T.CS1 ;GET COMMAND AND STATUS REG 1
6919 042746 012737 010226 003500    MOV      #RDY!CFMT!WRHEAD<↑C<GO>>, E.CS1 ;LOAD EXPECTED CS1
6920 042754 023737 003500 003440    CMP      E.CS1, T.CS1 ;CHECK IF CS1 CORRECT
6921 042762 001401          BEQ      TST51     ;;YES, GO ON TO NEXT TEST
6922 042764 104174          ERROR    174

```

.SBTTL **TYPE B INSTRUCTION ERRORS

;TEST 51 FORMAT ERROR (PART 1)

*
* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN
* RK06, IN 26 SECTOR FORMAT, CYLINDER 43, HEAD 0, DRIVE 0.
* CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF
* MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
* DRIVE AVAILABLE AND CONTROLLER ERROR SET.

```

6937 042766 000004          †TST51: SCOPE
6938 042770 012737 000144 001200    MOV      #100, $TIMES ;DO 100. ITERATIONS
6939 042776 013702 001270          MOV      $BASE, R2   ;LOAD RK611 BASE
6940 043002 012762 000040 000010    MOV      #SCLR, RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
6941 043010 012762 000040 000026    MOV      #DMD, RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
6942 043016 012762 000043 000020    MOV      #43, RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG

```

6943	043024	012762	000025	000000		MOV	#RDHEAD,RKCS1(R2)	:ISSUE READ HEADER
6944	043032	012700	000132			MOV	#22,#4+2,RO	:ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6945	043036	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6946	043044	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6947	043052	005300				DEC	RO	
6948	043054	001370				BNE	1\$	
6949	043056	005062	000026			CLR	RKMR1(R2)	:FINISH COMMAND IN NORMAL MODE
6950	043062	013700	003632			MOV	WAITIM,RO	:WAIT FOR READY
6951	043066	105762	000000		2\$:	TSTB	RKCS1(R2)	
6952	043072	100402				BMI	3\$	
6953	043074	005300				DEC	RO	
6954	043076	001373				BNE	2\$	
6955	043100	016237	000000	003440	3\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
6956	043106	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG 2
6957	043114	016237	000012	003452		MOV	RKDS(R2),T.DS	:STORE DRIVE STATUS REG
6958	043122	016237	000014	003454		MOV	RKER(R2),T.ER	:STORE ERROR REG
6959	043130	012737	100224	003500		MOV	#CERR!RDY!RDHEAD<↑C<GO>>,E.CS1	:LOAD EXPECTED CS1
6960	043136	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
6961	043144	012737	100001	003512		MOV	#SVAL!DRA,E.DS	:LOAD EXPECTED DRIVE STATUS REG
6962	043152	012737	000020	003514		MOV	#FMTE,E.ER	:LOAD EXPECTED ERROR REG
6963	043160	023737	003500	003440		MOV	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
6964	043166	001401				BEQ	4\$:YES, CONTINUE
6965	043170	104175				ERROR	175	
6966	043172	023737	003510	003450	4\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND AND STATUS REG 2 CORRECT
6967	043200	001401				BEQ	5\$:YES, CONTINUE
6968	043202	104176				ERROR	176	
6969	043204	023737	003512	003452	5\$:	CMP	E.DS,T.DS	:CHECK IF DRIVE STATUS REG CORRECT
6970	043212	001401				BEQ	6\$:YES, CONTINUE
6971	043214	104177				ERROR	177	
6972	043216	023737	003514	003454	6\$:	CMP	E.ER,T.ER	:CHECK IF ERR REG CORRECT
6973	043224	001401				BEQ	7\$:YES, CONTINUE
6974	043226	104200				ERROR	200	
6975	043230	013737	003440	003540	7\$:	MOV	T.CS1,P.CS1	:STORE PREVIOUS CONTENTS OF
6976	043236	013737	003450	003550		MOV	T.CS2,P.CS2	:COMMAND AND STATUS REG 1
6977	043244	013737	003452	003552		MOV	T.DS,P.DS	:COMMAND AND STATUS REG 2
6978	043252	013737	003454	003554		MOV	T.ER,P.ER	:DRIVE STATUS REG
6979								:AND ERROR REG
6980	043260	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
6981	043266	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
6982	043274	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG 2
6983	043302	016237	000012	003452		MOV	RKDS(R2),T.DS	:STORE DRIVE STATUS REG
6984	043310	016237	000014	003454		MOV	RKER(R2),T.ER	:STORE ERROR REG
6985	043316	012737	000200	003500		MOV	#RDY,E.CS1	:LOAD EXPECTED CS1
6986	043324	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
6987	043332	005037	003512			CLR	E.DS	:LOAD EXPECTED DRIVE STATUS REG
6988	043336	005037	003514			CLR	E.ER	:LOAD EXPECTED ERROR REG
6989	043342	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
6990	043350	001401				BEQ	11\$:YES, CONTINUE
6991	043352	104211				ERROR	211	:CS1 INCORRECT
6992	043354	023737	003510	003450	11\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
6993	043362	001401				BEQ	12\$:YES, CONTINUE
6994	043364	104212				ERROR	212	:CS2 INCORRECT
6995	043366	023737	003512	003452	12\$:	CMP	E.DS,T.DS	:CHECK DRIVE STATUS CORRECT
6996	043374	001401				BEQ	13\$:YES, CONTINUE
6997	043376	104213				ERROR	213	:DRIVE STATUS REG INCORRECT
6998	043400	023737	003514	003454	13\$:	CMP	E.ER,T.ER	:CHECK IF ERROR REG CORRECT

E11

```

7055 043676 013737 003454 003554      MOV      T.ER,P.ER      ; DRIVE STATUS REG
7056                                     ; AND ERROR REG
7057 043704 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ; CLEAR RK611
7058 043712 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ; STORE COMMAND AND STATUS REG 1
7059 043720 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ; STORE COMMAND AND STATUS REG 2
7060 043726 016237 000012 003452      MOV      RKDS(R2),T.DS   ; STORE DRIVE STATUS REG
7061 043734 016237 000014 003454      MOV      RKER(R2),T.ER   ; STORE ERROR REG
7062 043742 012737 000200 003500      MOV      #ROY,E.CS1     ; LOAD EXPECTED CS1
7063 043750 012737 000100 003510      MOV      #IR,E.CS2      ; LOAD EXPECTED CS2
7064 043756 005037 003512              CLR      E.DS           ; LOAD EXPECTED DRIVE STATUS REG
7065 043762 005037 003514              CLR      E.ER           ; LOAD EXPECTED ERROR REG
7066 043766 023737 003500 003440      CMP      E.CS1,T.CS1    ; CHECK CS1 CORRECT
7067 043774 001401 11$              BEQ      11$            ; YES, CONTINUE
7068 043776 104211 21$              ERROR   21$            ; CS1 INCORRECT
7069 044000 023737 003510 003450 11$:    CMP      E.CS2,T.CS2    ; CHECK CS2 CORRECT
7070 044006 001401 12$              BEQ      12$            ; YES, CONTINUE
7071 044010 104212 22$              ERROR   22$            ; CS2 INCORRECT
7072 044012 023737 003512 003452 12$:    CMP      E.DS,T.DS      ; CHECK DRIVE STATUS CORRECT
7073 044020 001401 13$              BEQ      13$            ; YES, CONTINUE
7074 044022 104213 23$              ERROR   23$            ; DRIVE STATUS REG INCORRECT
7075 044024 023737 003514 003454 13$:    CMP      E.ER,T.ER      ; CHECK IF ERROR REG CORRECT
7076 044032 001401 14$              BEQ      14$            ; YES, GO ON TO NEXT TEST
7077 044034 104214 24$              ERROR   24$            ; ERROR REG INCORRECT
7078                                     ;
7079                                     ;
7080                                     ;
7081                                     ;
7082                                     ;
7083                                     ;
7084                                     ;
7085                                     ;
7086                                     ;
7087                                     ;
7088                                     ;
7089                                     ;
7090                                     ;

```

```

*****
*TEST 53      FAULT SETTING CONTROLLER ERROR
*
* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO
* AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE
* 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6.
* TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE
* AVAILABLE AND CONTROLLER ERROR SET.
*****

```

```

7091 044036 000004 000144 001200      TST53: SCOPE
7092 044040 012737 000144 001200      MOV      #100,$TIMES    ; DO 100. ITERATIONS
7093 044046 013702 001270 000010      MOV      $BASE,R2      ; LOAD RK611 BASE
7094 044052 012762 000040 000010      MOV      #SCLR,RKCS2(R2) ; CLEAR RK611 SUBSYSTEM
7095 044060 012762 000040 000026      MOV      #DMD,RKMR1(R2) ; PUT RK611 IN MAINTENANCE MODE
7096 044066 012762 000003 000020      MOV      #3,RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG
7097 044074 012762 177775 000002      MOV      #-3,RKWC(R2)  ; LOAD WORD COUNT
7098 044102 012762 067254 000004      MOV      #WRBUFF,RKBA(R2) ; LOAD BUS ADDRESS
7099 044110 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ; ISSUE WRITE HEADER
7100 044116 012700 000132 000026 1$:    MOV      #22,*4+2,RO    ; ISSUE CLOCKS UNTIL PHASE ADDRESS 6
7101 044122 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2)
7102 044130 012762 000040 000026      MOV      #DMD,RKMR1(R2)
7103 044136 005300 1$              DEC      RO
7104 044140 001370 1$              BNE     1$
7105 044142 005062 000026 2$:    CLR      RKMR1(R2)      ; FINISH COMMAND IN NORMAL MODE
7106 044146 013700 003632 2$:    MOV      WAITIM,RO     ; WAIT FOR READY
7107 044152 105762 000000 2$:    TSTB   RKCS1(R2)
7108 044156 100402 3$              BMI     3$
7109 044160 005300 4$              DEC      RO
7110 044162 001373 5$              BNE     5$

```



```

7159          .SBTTL  END OF PASS ROUTINE
7160
7161          ;*****
7162          ;INCREMENT THE PASS NUMBER ($PASS)
7163          ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7164          ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7165          ;*IF THERES A MONITOR GO TO IT
7166          ;*IF THERE ISN'T JUMP TO NEWPAS
7167
7168          $EOP:
7169          044476 000004          SCOPE
7170          044500 005037 001102          CLR          $TSTNM          ;; ZERO THE TEST NUMBER
7171          044504 005037 001200          CLR          $TIMES          ;; ZERO THE NUMBER OF ITERATIONS
7172          044510 005237 001222          INC          $PASS          ;; INCREMENT THE PASS NUMBER
7173          044514 042737 100000 001222          BIC          #100000,$PASS  ;; DON'T ALLOW A NEG. NUMBER
7174          044522 005327          DEC          (PC)+          ;; LOOP?
7175          044524 000001          $EOPCT: .WORD          1
7176          044526 003063          BGT          $DOAGN          ;; YES
7177          044530 012737          MOV          (PC)+,2(PC)+  ;; RESTORE COUNTER
7178          044532 000001          $ENDCT: .WORD          1
7179          044534 044524          $EOPCT
7180          044536 104401 044544          TYPE          65$          ;; TYPE ASCIZ STRING
7181          044542 000407          BR          64$          ;; GET OVER THE ASCIZ
7182          ;;65$: .ASCIZ <12><15>/END PASS #/
7183          64$:
7184          044562 013746 001222          MOV          $PASS,-(SP)  ;; SAVE $PASS FOR TYPEOUT
7185          7185          ;; TYPE PASS NUMBER
7186          044566 104405          TYPDS          ;; GO TYPE--DECIMAL ASCII WITH SIGN
7187          044570 104401 044576          TYPE          67$          ;; TYPE ASCIZ STRING
7188          044574 000421          BR          66$          ;; GET OVER THE ASCIZ
7189          ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
7190          66$:
7191          044640 013746 001112          MOV          $ERTTL,-(SP)  ;; SAVE $ERTTL FOR TYPEOUT
7192          7192          ;; TOTAL NUMBER OF ERRORS
7193          044644 104405          TYPDS          ;; GO TYPE--DECIMAL ASCII WITH SIGN
7194          044646 104401 001211          TYPE          $CRLF          ;; TYPE CARRIAGE RETURN, LINE FEED
7195          044652 005037 001112          CLR          $ERTTL          ;; CLEAR ERROR TOTAL
7196          044656 013700 000042          $GET42: MOV          2#42,R0          ;; GET MONITOR ADDRESS
7197          044662 001405          BEQ          $DOAGN          ;; BRANCH IF NO MONITOR
7198          044664 000005          RESET          ;; CLEAR THE WORLD
7199          044666 004710          $ENDAD: JSR          PC,(R0)  ;; GO TO MONITOR
7200          044670 000240          NOP          ;; SAVE ROOM
7201          044672 000240          NOP          ;; FOR
7202          044674 000240          NOP          ;; ACT11
7203          044676          $DOAGN:
7204          044676 000137          JMP          2(PC)+          ;; RETURN
7205          044700 004666          $RTNAD: .WORD          NEWPAS
7206          044702          377          377          000          $ENULL: .BYTE          -1,-1,0          ;; NULL CHARACTER STRING
7207          044706          .EVEN
7208
7209          .SBTTL  GENERATE BAD PARITY IN MEMORY
7210
7211          044706 013703 003640          WRTPAR: MOV          CSRPTR,R3  ;; GET ADDRESS OF CSR TO BE USED
7212          044712 012713 000004          MOV          #WR.PAR,(R3)  ;; ALLOW SETTING OF BAD PARITY
7213          044716 012737 000157 067270          MOV          #157,BADPAR  ;; WRITE BAD PARITY
7214          044724 012713 000001          MOV          #PAR.EN,(R3)  ;; ALLOW PARITY DETECTION

```

H11

```

7215 044730 000207          RTS      PC          ;RETURN
7216
7217          .SBTTL  CHECK FOR MEMORY CHECK ENABLE
7218
7219 044732 012737 045066 000004 PARCHK: MOV      #20$,ERRVEC      ;SET VECTOR FOR MEMORY PARITY CHECK
7220 044740 012737 000340 000006 MOV      #PR7,ERRVEC+2
7221 044746 012737 000000 067270 MOV      #0,BADPAR      ;LOAD GOOD PARITY
7222 044754 005037 003630 CLR      MEMPAR      ;CLEAR FLAG
7223 044760 005037 003636 CLR      PARPRE      ;CLEAR PARITY PRESENT FLAG
7224 044764 012703 172100 MOV      #MEMBAS,R3      ;LOAD REGISTER TO DETERMINE IF
7225                                ;MEMORY CHECK ENABLE AVAILABLE
7226 044770 012704 000001 MOV      #1,R4      ;INITIALIZE MASK
7227 044774 012713 000001 16$: MOV      #PAR.EN,(R3) ;ENABLE MEMORY CHECK
7228 045000 005713 TST      (R3)
7229 045002 050437 003630 BIS      R4,MEMPAR      ;SET PARITY PRESENT BIT
7230 045006 005737 003636 TST      PARPRE      ;TEST IF PARITY PRESENT ALREADY SET
7231 045012 001017 BNE      10$          ;YES - SKIP
7232 045014 012737 045072 000114 MOV      #30$,MEMVEC      ;SET UP PARITY TRAP VECTOR
7233 045022 012737 000340 000115 MOV      #PR7,MEMVEC+2
7234 045030 012713 000004 MOV      #WR.PAR,(R3) ;ALLOW SETTING OF BAD PARITY
7235 045034 012737 177777 067270 MOV      #-1,BADPAR      ;TRY SET BAD PARITY IN BADPAR
7236 045042 012713 000001 MOV      #PAR.EN,(R3) ;SET TO TRAP ON PARITY ERROR
7237 045046 005737 067270 TST      BADPAR      ;CAUSE TRAP IF BAD PARITY
7238
7239 045052 062703 000002 10$: ADD      #2,R3          ;BUMP TO NEXT CSR
7240 045056 000241 CLC
7241 045060 006104 ROL      R4          ;ROTATE MASK
7242 045062 001344 BNE      16$          ;SKIP IF MASK 0
7243 045064 000416 BR       35$          ;ELSE GO TO EXIT
7244 045066 022626 20$: CMP      (SP)+,(SP)+ ;CLEAN STACK
7245 045070 000770 BR
7246
7247 045072 022626 30$: CMP      (SP)+,(SP)+ ;CLEAR STACK
7248 045074 005013 CLR      (R3)      ;CLEAR CSR REGISTER
7249 045076 005037 067270 CLR      BADPAR      ;CLEAN UP BAD PARITY
7250 045102 012713 000001 MOV      #PAR.EN,(R3) ;SET TO DETECT PARITY ERROR
7251 045106 010337 003640 MOV      R3,CSRPTR ;STORE CSR TO BE USED IN TESTS
7252 045112 012737 177777 003636 MOV      #-1,PARPRE ;SET PARITY PRESENT FLAG
7253 045120 000754 BR       10$          ;TEST NEXT CSR
7254
7255 045122 012737 000006 000004 35$: MOV      #ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
7256 045130 005037 000006 CLR      ERRVEC+2
7257 045134 005737 003630 TST      MEMPAR      ;TEST IF ANY PARITY AVAILABLE
7258 045140 001006 BNE      37$          ;YES - SKIP
7259 045142 012737 000116 000114 MOV      #MEMVEC+2,MEMVEC ;ELSE RESTORE TRAP CATCHER
7260 045150 005037 000116 CLR      MEMVEC+2
7261 045154 000413 BR       40$          ;SKIP
7262 045156 012737 046266 000114 37$: MOV      #MEMERR,MEMVEC ;ELSE SET FOR PARITY TRAPS
7263 045164 012737 000340 000116 MOV      #PR7,MEMVEC+2
7264 045172 012746 000000 MOV      #PR0,-(SP) ;SET PRIORITY 0
7265 045176 012746 045204 MOV      #40$,-(SP)
7266 045202 000002 RTI
7267 045204 000207 40$: RTS      PC          ;RETURN
7268
7269          .SBTTL  SIMULATE ONE BIT OF WRITE DATA IN MAINTANENCE MODE
7270

```

7271	045206	052737	002400	003524	WRTBIT:	BIS	#MCLK!MEWD,E.MR1	:CREATE EXPECTED MAINT. REG. 1
7272	045214	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	:PROVIDE 1ST UPWARD TRANSITION
7273	045222	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT. REG. 1
7274	045230	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MAINT REG. 1 CORRECT
7275	045236	001416				BEQ	3\$:YES, PROVIDE DOWNWARD TRANSITION
7276	045240	012737	045260	001202		MOV	#1\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
7277	045246	012737	066374	003172		MOV	#EMW1,EMW+2	:LOAD ERROR MESSAGE
7278	045254	011646				MOV	(SP),-(SP)	:SAVE RETURN
7279	045256	000207				RTS	PC	:MR1 INCORRECT ON UPWARD TRANSITION
7280								
7281	045260	032777	001000	133652	1\$:	BIT	#SW9, \$SWR	:CHECK IF LOOP ON ERROR
7282	045266	001402				BEQ	3\$:NO, CONTINUE
7283	045270	000137	046142			JMP	63\$:YES, LOOP ON ERROR
7284								
7285	045274	042737	014400	003524	3\$:	BIC	#MCLK!PCA!PCD,E.MR1	:INITIALIZE MAINTENANCE REG. 1
7286	045302	052737	042000	003524		BIS	#MEWD!WRTGAT,E.MR1	
7287	045310	005737	003614			TST	PR.BIT	:CHECK IF ONE
7288	045314	001152				BNE	20\$:YES, SIMULATE ONE
7289	045316	005737	003616			TST	M1.BIT	:CHECK IF PREVIOUS ONE
7290	045322	001023				BNE	10\$:YES, NO TRANSITION
7291	045324	042737	002000	003524		BIC	#MEWD,E.MR1	:INDICATE TRANSITION
7292	045332	005737	003612			TST	P1.BIT	:CHECK IF NEXT BIT = 1
7293	045336	001007				BNE	5\$:YES, CHECK FOR PRECOMP ADVANCE
7294	045340	005737	003620			TST	M2.BIT	:CHECK FOR PRECOMP. ADVANCE
7295	045344	001412				BEQ	10\$:NO, CLOCK IN ZERO
7296	045346	052737	010000	003524		BIS	#PCD,E.MR1	:SET PRECOMP. DELAY
7297	045354	000406				BR	10\$:CLOCK IN ZERO
7298								
7299	045356	005737	003620		5\$:	TST	M2.BIT	:CHECK FOR PRECOMP. ADVANCE
7300	045362	001003				BNE	10\$:CLOCK IN ZERO
7301	045364	052737	004000	003524		BIS	#PCA,E.MR1	:SET PRECOMP. ADVANCE
7302	045372	012762	000040	000026	10\$:	MOV	#DMD,RKMR1(R2)	:CLOCK IN DATA BIT
7303	045400	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MR1
7304	045406	023737	003464	003524		CMP	T.MR1,E.MR1	:CHECK IF MR1 CORRECT
7305	045414	001416				BEQ	12\$:YES, CONTINUE
7306	045416	012737	045436	001202		MOV	#11\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
7307	045424	012737	066463	003172		MOV	#EMW2,EMW+2	:LOAD ERROR MESSAGE
7308	045432	011646				MOV	(SP),-(SP)	:SAVE RETURN
7309	045434	000207				RTS	PC	:MR1 INCORRECT
7310								
7311	045436	032777	001000	133474	11\$:	BIT	#SW9, \$SWR	:CHECK IF LOOP ON ERROR
7312	045444	001402				BEQ	12\$:NO, CONTINUE
7313	045446	000137	046142			JMP	63\$:YES, LOOP ON ERROR
7314								
7315	045452	052737	002400	003524	12\$:	BIS	#MCLK!MEWD,E.MR1	:CREATE EXPECTED MAINT REG 1
7316	045460	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	:PROVIDE 2ND UPWARD TRANSITION
7317	045466	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1
7318	045474	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MAINT REG. 1 CORRECT
7319	045502	001416				BEQ	15\$:YES, CONTINUE
7320	045504	012737	045524	001202		MOV	#13\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
7321	045512	012737	066554	003172		MOV	#EMW3,EMW+2	:LOAD ERROR MESSAGE
7322	045520	011646				MOV	(SP),-(SP)	:SAVE RETURN
7323	045522	000207				RTS	PC	:MR1 INCORRECT
7324								
7325	045524	032777	001000	133406	13\$:	BIT	#SW9, \$SWR	:CHECK IF LOOP ON ERROR
7326	045532	001402				BEQ	15\$:NO, CONTINUE

J11

7327	045534	000137	046142			JMP	63\$;YES, LOOP ON ERROR
7328									
7329	045540	052737	002000	003524	15\$:	BIS	#MEWD,E.MR1		;RESET TRANSITION INDICATION
7330	045546	042737	000400	003524		BIC	#MCLK,E.MR1		
7331	045554	012762	000040	000026		MOV	#DMD,RKMR1(R2)		;SUPPLY LAST PART OF DATA
7332	045562	016237	000026	003464		MOV	RKMR1(R2),T.MR1		;STORE MR1
7333	045570	023737	003464	003524		CMP	T.MR1,E.MR1		;CHECK IF MR1 CORRECT
7334	045576	001414				BEQ	18\$;YES, RETURN
7335	045600	012737	045620	001202		MOV	#17\$,SESCAPE		;LOAD ESCAPE FOR LOOP ON ERROR
7336	045606	012737	066643	003172		MOV	#EMW4,EMW+2		;LOAD ERROR MESSAGE
7337	045614	011646				MOV	(SP),-(SP)		;SAVE RETURN
7338	045616	000207				RTS	PC		;MR1 INCORRECT
7339									
7340	045620	032777	001000	133312	17\$:	BIT	#SW9,@SWR		;CHECK IF LOOP ON ERROR
7341	045626	001145				BNE	63\$;YES, LOOP ON ERROR
7342	045630	005037	001202		18\$:	CLR	SESCAPE		;CLEAR ESCAPE
7343	045634	062716	000002			ADD	#2,(SP)		;ADJUST RETURN
7344	045640	000207				RTS	PC		;RETURN
7345									
7346	045642	005737	003612		20\$:	TST	P1.BIT		;CHECK IN NEXT BIT A ONE
7347	045646	001007				BNE	30\$;YES, CHECK IF PRECOMP DELAY
7348	045650	005737	003616			TST	M1.BIT		;CHECK FOR PRECOMP ADVANCE
7349	045654	001415				BEQ	40\$;NO, CLOCK IN DATA BIT
7350	045656	052737	004000	003524		BIS	#PCA,E.MR1		;SET PRECOMP. ADVANCE
7351	045664	000411				BR	40\$;CHECK MR1
7352									
7353	045666	042737	000400	003524	30\$:	BIC	#MCLK,E.MR1		;RESET MAINT CLOCK IN EXPECTED MR1
7354	045674	005737	003616			TST	M1.BIT		;CHECK FOR PRECOMP DELAY
7355	045700	001003				BNE	40\$;NO, CHECK MR1
7356	045702	052737	010000	003524		BIS	#PCD,E.MR1		;SET SET PRECOMP DELAY
7357	045710	012762	000040	000026	40\$:	MOV	#DMD,RKMR1(R2)		;CLOCK IN DATA BIT
7358	045716	016237	000026	003464		MOV	RKMR1(R2),T.MR1		;STORE MR1
7359	045724	023737	003464	003524		CMP	T.MR1,E.MR1		;CHECK MR1 CORRECT
7360	045732	001414				BEQ	42\$;YES, CLOCK IN REST OF BIT
7361	045734	012737	045754	001202		MOV	#41\$,SESCAPE		;LOAD ESCAPE FOR LOOP ON ERROR
7362	045742	012737	066463	003172		MOV	#EMW2,EMW+2		;LOAD ERROR MESSAGE
7363	045750	011646				MOV	(SP),-(SP)		;SAVE RETURN
7364	045752	000207				RTS	PC		;MR1 INCORRECT
7365									
7366	045754	032777	001000	133156	41\$:	BIT	#SW9,@SWR		;CHECK IF LOOP ON ERROR
7367	045762	001067				BNE	63\$;YES, LOOP ON ERROR
7368	045764	052737	000400	003524	42\$:	BIS	#MCLK,E.MR1		;CREATE EXPECTED MAINT. REG. 1
7369	045772	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)		;PROVIDE 2ND UPWARD TRANSITION
7370	046000	016237	000026	003464		MOV	RKMR1(R2),T.MR1		;STORE MAINT REG 1
7371	046006	023737	003524	003464		CMP	E.MR1,T.MR1		;CHECK IF MAINT REG 1 CORRECT
7372	046014	001414				BEQ	45\$;YES, CONTINUE
7373	046016	012737	046036	001202		MOV	#43\$,SESCAPE		;LOAD ESCAPE
7374	046024	012737	066554	003172		MOV	#EMW3,EMW+2		;LOAD ERROR MESSAGE
7375	046032	011646				MOV	(SP),-(SP)		;SAVE RETURN
7376	046034	000207				RTS	PC		;MR1 INCORRECT
7377									
7378	046036	032777	001000	133074	43\$:	BIT	#SW9,@SWR		;CHECK IF LOOP ON ERROR
7379	046044	001036				BNE	63\$;YES, LOOP ON ERROR
7380	046046	042737	002400	003524	45\$:	BIC	#MEWD!MCLK,E.MR1		;SET TRANSITION
7381	046054	012762	000040	000026		MOV	#DMD,RKMR1(R2)		;CLOCK TRANSITION
7382	046062	016237	000026	003464		MOV	RKMR1(R2),T.MR1		;STORE MR1

```

7383 046070 023737 003464 003524      CMP      T.MR1,E.MR1      ;CHECK IF MR1 CORRECT
7384 046076 001414                      BEQ      50$            ;YES, RETURN
7385 046100 012737 046120 001202      MOV      #47$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
7386 046106 012737 066643 003172      MOV      #EMW4,EMW+2    ;LOAD ERROR MESSAGE
7387 046114 011646                      MOV      (SP),-(SP)     ;SAVE RETURN
7388 046116 000207                      RTS      PC             ;MR1 INCORRECT
7389
7390 046120 032777 001000 133012 47$:   BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
7391 046126 001005                      BNE      63$            ;YES, LOOP ON ERROR
7392 046130 005037 001202 50$:   CLR      $ESCAPE       ;CLEAR ESCAPE
7393 046134 062716 000002      ADD      #2,(SP)       ;ADJUST RETURN
7394 046140 000207                      RTS      PC             ;RETURN
7395
7396 046142 005037 001202 63$:   CLR      $ESCAPE       ;CLEAR ESCAPE
7397 046146 012706 001100      MOV      #STACK,SP     ;FORCE STACK
7398 046152 000177 132732      JMP      $JLPERA       ;LOOP ON ERROR
7399
7400      .SBTTL  SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE
7401
7402 046156 005737 003614  RDBIT:  TST      PR.BIT      ;CHECK IF ONE
7403 046162 001024                      BNE      10$            ;YES, SIMULATE ONE
7404 046164 005737 003616      TST      M1.BIT        ;CHECK IF PREVIOUS ONE
7405 046170 001404                      BEQ      4$             ;NO, INSERT TRANSITION
7406 046172 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2) ;YES, DO NOT INSERT TRANSITION
7407 046200 000403                      BR       5$             ;CLOCK IN ZERO
7408
7409 046202 012762 001440 000026 4$:   MOV      #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
7410 046210 012762 000040 000026 5$:   MOV      #DMD,RKMR1(R2) ;CLOCK IN ZERO
7411 046216 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2)
7412 046224 012762 000040 000026      MOV      #DMD,RKMR1(R2)
7413 046232 000207                      RTS      PC             ;RETURN
7414
7415 046234 012762 000440 000026 10$:  MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
7416 046242 012762 000040 000026      MOV      #DMD,RKMR1(R2)
7417 046250 012762 001440 000026      MOV      #DMD!MCLK!MERD,RKMR1(R2)
7418 046256 012762 000040 000026      MOV      #DMD,RKMR1(R2)
7419 046264 000207                      RTS      PC             ;RETURN
7420
7421      .SBTTL  MEMORY CHECK ENABLE TRAP
7422
7423 046266 012737 046302 001202  MEMERR: MOV      #10$, $ESCAPE ;LOAD ESCAPE
7424 046274 011637 003604                      MOV      (SP),TRAPPC  ;STORE PC
7425 046300 104147                      ERROR    147          ;REPORT MEM PARITY ERROR
7426 046302 005037 001202 10$:   CLR      $ESCAPE       ;CLEAR ESCAPE
7427 046306 032777 001000 132624      BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
7428 046314 001001                      BNE      15$            ;YES, FORCE STACK AND TRY AGAIN
7429 046316 000002                      RTI                      ;NO, RETURN
7430
7431 046320 012706 001100 15$:   MOV      #STACK,SP     ;INITIALIZE STACK
7432 046324 000177 132560      JMP      $JLPERA       ;LOOP ON ERROR
7433
7434      .SBTTL  ROUTINE TO SIZE MEMORY
7435
7436      ;*****
7437      ;CALL:
7438      ;*      JSR      PC,$SIZE

```

```

7439          *      RETURN
7440          *SLSTAD WILL CONTAIN:
7441          *      WITH KT11 OPTION          -- LAST VIRTUAL ADDRESS OF THE LAST BANK
7442          *      WITHOUT KT11 OPTION       -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
7443          *SLSTBK WILL CONTAIN THE LAST BANK AS A SAF
7444          *SKT11 IS THE MEMORY MANAGEMENT KEY
7445          *BIT07 = 0 DON'T USE MEMORY MANAGEMENT
7446          *      MUST BE SETUP BEFORE THE CALL
7447          *BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
7448          *      DETERMINED BY ROUTINE
7449
7450 046330 010046  $SIZE:  MOV    R0,-(SP)      ;; SAVE R0 ON THE STACK
7451 046332 010146      MOV    R1,-(SP)      ;; SAVE R1 ON THE STACK
7452 046334 010246      MOV    R2,-(SP)      ;; SAVE R2 ON THE STACK
7453 046336 010346      MOV    R3,-(SP)      ;; SAVE R3 ON THE STACK
7454 046340 013746 000004  MOV    @#ERRVEC,-(SP)  ;; SAVE PRESENT ERROR VECTOR PS & PC
7455 046344 013746 000006  MOV    @#ERRVEC+2,-(SP)
7456 046350 010600      MOV    SP,R0          ;; SAVE THE STACK POINTER
7457          ;;SET THE ERRVEC PS TO THE PRESENT PS
7458 046352 104400      TRAP
7459 046354 012637 000006  MOV    (SP)+,@#ERRVEC+2 ;; PUSH OLD PSW AND PC ON STACK
7460 046360 012701 003776  MOV    #3776,R1        ;; SAVE THE PSW IN @#ERRVEC+2
7461 046364 105727      MOV    (PC)+          ;; SETUP ADDRESS
7462 046366 000200      SKT11: .WORD 200      ;; USE MEMORY MANAGEMENT?
7463 046370 100062      BPL    $SCORE         ;; SET TO USE MEMORY MANAGEMENT
7464 046372 012737 046530 000004  MOV    #SKTNEX,@#ERRVEC ;; BR IF NO
7465 046400 005737 177572      TST    @#SR0         ;; SET FOR TIMEOUT
7466 046404 052737 100000 046366  BIS    #100000,SKT11  ;; KT11 ARE YOU THERE?
7467 046412 005046      CLR    -(SP)        ;; YES--SET KT11 KEY
7468 046414 012702 172340      MOV    #KIPAR0,R2    ;; INITIALIZE FOR "PAR" LOADING
7469 046420 012703 000010      MOV    #108,R3      ;; ADDRESS OF FIRST "PAR"
7470 046424 012762 077406 177740 1$:  MOV    #77406,-40(R2) ;; LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
7471 046432 011622      MOV    (SP),(R2)+    ;; PDR = 4K UP, READ/WRITE
7472 046434 062716 000200      ADD    #200,(SP)    ;; LOAD "PAR"
7473 046440 077307      SOB    R3,1$        ;; UPDATE FOR NEXT "PAR"
7474 046442 012742 177600      MOV    #177600,-(R2) ;; LOOP UNTIL ALL EIGHT ARE LOADED
7475 046446 005042      CLR    -(R2)        ;; SETUP KIPAR7 FOR I/O
7476 046450 012737 046466 000004  MOV    #25,@#ERRVEC  ;; SETUP KIPAR6 FOR TESTING
7477 046456 012737 000020 172516  MOV    #20,@#SR3    ;; CATCH TIMEOUT IF NO SR3
7478 046464 000401      BR     3$           ;; ENABLE 22 BIT MODE
7479 046466 022626      2$:  CMP    (SP)+,(SP)+  ;; THIS PDP-11 HAS A SR3 REGISTER
7480 046470 005237 177572      3$:  INC    @#SR0        ;; CLEAN OFF THE STACK--NO SR3
7481 046474 012737 046520 000004  MOV    #SKTOUT,@#ERRVEC ;; TURN ON MEMORY MANAGEMENT
7482 046502 005737 143776      4$:  TST    @#143776    ;; SET FOR TIME OUT
7483 046506 062712 000040      ADD    #40,(R2)    ;; TRAP ON NON-EX-MEM
7484 046512 023712 172356      CMP    @#KIPAR7,(R2) ;; MAKE A 1K STEP
7485 046516 101371      BHI    4$          ;; LAST ONE?
7486 046520 011202      SKTOUT: MOV    (R2),R2    ;; NO--TRY IT
7487 046522 005037 177572      CLR    @#SR0        ;; GET LAST BANK+1
7488 046526 000421      BR     $SIZEX      ;; TURN OFF MEMORY MANAGEMENT
7489 046530 042737 100000 046366  SKTNEX: BIC    #100000,SKT11 ;; KT11 NON-EXISTENT
7490 046536 012737 046566 000004  SCORE: MOV    #SCROUT,@#ERRVEC ;; SET FOR TIMEOUT
7491 046544 005002      CLR    R2          ;; SET UP BANK
7492 046546 062701 004000      1$:  ADD    #4000,R1    ;; INCREMENT BY 1K
7493 046552 062702 000040      ADD    #40,R2      ;; 1K STEP
7494 046556 005711      TST    (R1)        ;; TRAP ON TIME OUT

```

```

7495 046560 022701 177776          CMP      #177776,R1      ;;LAST ONE
7496 046564 001370          BNE      1$            ;;NO--TRY AGAIN
7497 046566 162701 004000          $CROUT: SUB     #4000,R1
7498 046572 162702 000040          $SIZEX: SUB     #40,R2      ;;DROP BACK
7499 046576 010006          MOV      RO,SP        ;;RESTORE THE STACK
7500 046600 012637 000006          MOV      (SP)+,#ERRVEC+2 ;;RESTORE ERROR VECTOR
7501 046604 012637 000004          MOV      (SP)+,#ERRVEC
7502 046610 010137 046632          MOV      R1,$LSTAD    ;;LAST ADDRESS
7503 046614 010237 046634          MOV      R2,$LSTBK    ;;LAST BANK
7504 046620 012603          MOV      (SP)+,R3     ;;RESTORE R3
7505 046622 012602          MOV      (SP)+,R2     ;;RESTORE R2
7506 046624 012601          MOV      (SP)+,R1     ;;RESTORE R1
7507 046626 012600          MOV      (SP)+,R0     ;;RESTORE R0
7508 046630 000207          RTS      PC
7509 046632 000000          $LSTAD: .WORD 0        ;;CONTAINS THE LAST ADDRESS
7510 046634 000000          $LSTBK: .WORD 0        ;;CONTAINS THE LAST BANK
7511          .SBTTL SCOPE HANDLER ROUTINE
7512
7513          ;*****
7514          ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7515          ;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7516          ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7517          ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7518          ;SW14=1      LOOP ON TEST
7519          ;SW11=1      INHIBIT ITERATIONS
7520          ;SW09=1      LOOP ON ERROR
7521          ;SW08=1      LOOP ON TEST IN SWR<7:0>
7522          ;CALL
7523          ;          SCOPE          ;;SCOPE=IOT
7524
7525          $SCOPE:
7526 046636 104407          CKSWR
7527 046640 032777 040000 132272 1$: BIT      #BIT14,#SWR    ;;TEST FOR CHANGE IN SOFT-SWR
7528 046646 001131          BNE      $OVER       ;;LOOP ON PRESENT TEST?
7529          ;#####START OF CODE FOR THE XOR TESTER#####
7530 046650 000416          $XTSTR: BR      6$    ;;YES IF SW14=1
7531          ;IF RUNNING ON THE "XOR" TESTER CHANGE
7532 046652 013746 000004          MOV      #ERRVEC,-(SP) ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7533 046656 012737 046676 000004          MOV      #SS,#ERRVEC  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7534 046664 005737 177060          TST      #177060     ;;SET FOR TIMEOUT
7535 046670 012637 000004          MOV      (SP)+,#ERRVEC ;;TIME OUT ON XOR?
7536 046674 000500          BR      $SVLAD       ;;RESTORE THE ERROR VECTOR
7537 046676 022626          5$: CMP      (SP)+,(SP)+ ;;GO TO THE NEXT TEST
7538 046700 012637 000004          MOV      (SP)+,#ERRVEC ;;CLEAR THE STACK AFTER A TIME OUT
7539 046704 000440          BR      7$          ;;RESTORE THE ERROR VECTOR
7540 046706          6$;#####END OF CODE FOR THE XOR TESTER#####
7541 046706 032777 000400 132224 7$: BIT      #BIT08,#SWR  ;;LOOP ON S. :. TEST?
7542 046714 001421          BEQ      2$          ;;BR IF NO
7543 046716 005046          CLR      -(SP)      ;;CLEAR A TEMP. LOCATION
7544 046720 117716 132214          MOV      #SWR,(SP)   ;;PICKUP THE DESIRED TEST NUMBER
7545 046724 001414          BEQ      8$          ;;BRANCH IF BAD TEST NUMBER IN SWR
7546 046726 022716 000053          CMP      #53,(SP)   ;;CHECK THE NUMBER IN THE SWR
7547 046732 002411          BLT      8$          ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
7548 046734 011637 001102          MOV      (SP),$STNM ;;UPDATE THE TEST NUMBER
7549 046740 005316          DEC      (SP)       ;;BACKUP BY ONE
7550 046742 006316          ASL      (SP)       ;;SCALE THE TEST NUMBER AS AN INDEX

```

Address	Hex	Hex	Hex	Label	Code	Comment
7551	046744	062716	047150	ADD	SSW08TBL, (SP)	FORM THE ADDRESS OF TEST POINTER
7552	046750	013637	001106	MOV	2(SP)+, \$LPADR	SET LOOP ADDRESS TO DESIRED TEST
7553	046754	000466		BR	\$OVER	GO LOOP ON THE TEST
7554	046756	005726		8\$: TST	(SP)+	CLEAN THE BAD TEST NUMBER OFF OF THE STACK
7555	046760	105737	001103	2\$: TSTB	\$ERFLG	HAS AN ERROR OCCURRED?
7556	046764	001421		BEQ	3\$	BR IF NO
7557	046766	123737	001115 001103	CMPB	\$ERMAX, \$ERFLG	MAX. ERRORS FOR THIS TEST OCCURRED?
7558	046774	101015		BHI	3\$	BR IF NO
7559	046776	032777	001000 132134	BIT	\$BIT09, 2SWR	LOOP ON ERROR?
7560	047004	001404		BEQ	4\$	BR IF NO
7561	047006	013737	001110 001106	7\$: MOV	\$LPERR, \$LPADR	SET LOOP ADDRESS TO LAST SCOPE
7562	047014	000446		BR	\$OVER	
7563	047016	105037	001103	4\$: CLRB	\$ERFLG	ZERO THE ERROR FLAG
7564	047022	005037	001200	CLR	\$TIMES	CLEAR THE NUMBER OF ITERATIONS TO MAKE
7565	047026	000415		BR	1\$	ESCAPE TO THE NEXT TEST
7566	047030	032777	004000 132102	3\$: BIT	\$BIT11, 2SWR	INHIBIT ITERATIONS?
7567	047036	001011		BNE	1\$	BR IF YES
7568	047040	005737	001222	TST	\$PASS	IF FIRST PASS OF PROGRAM
7569	047044	001406		BEQ	1\$	INHIBIT ITERATIONS
7570	047046	005237	001104	INC	\$ICNT	INCREMENT ITERATION COUNT
7571	047052	023737	001200 001104	CMP	\$TIMES, \$ICNT	CHECK THE NUMBER OF ITERATIONS MADE
7572	047060	002024		BGE	\$OVER	BR IF MORE ITERATION REQUIRED
7573	047062	012737	000001 001104	1\$: MOV	#1, \$ICNT	REINITIALIZE THE ITERATION COUNTER
7574	047070	013737	047146 001200	MOV	\$MXCNT, \$TIMES	SET NUMBER OF ITERATIONS TO DO
7575	047076	105237	001102	SSVLAD: INCB	\$STNM	COUNT TEST NUMBERS
7576	047102	113737	001102 001220	MOVB	\$STNM, \$STEN	SET TEST NUMBER IN APT MAILBOX
7577	047110	011637	001106	MOV	(SP), \$LPADR	SAVE SCOPE LOOP ADDRESS
7578	047114	011637	001110	MOV	(SP), \$LPERR	SAVE ERROR LOOP ADDRESS
7579	047120	005037	001202	CLR	\$ESCAPE	CLEAR THE ESCAPE FROM ERROR ADDRESS
7580	047124	112737	000001 001115	MOVB	#1, \$ERMAX	ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7581	047132	013777	001102 132002	SOVER: MOV	\$STNM, 2DISPLAY	DISPLAY TEST NUMBER
7582	047140	013716	001106	MOV	\$LPADR, (SP)	FUDGE RETURN ADDRESS
7583	047144	000002		RTI		FIXES PS
7584	047146	003720		\$MXCNT: 2000.		MAX. NUMBER OF ITERATIONS
7585	047150			SSW08TBL:		
7586	047150	004706		.WORD	TST1+2	STARTING ADDRESS OF TEST 1
7587	047152	005124		.WORD	TST2+2	STARTING ADDRESS OF TEST 2
7588	047154	005342		.WORD	TST3+2	STARTING ADDRESS OF TEST 3
7589	047156	005556		.WORD	TST4+2	STARTING ADDRESS OF TEST 4
7590	047160	005772		.WORD	TST5+2	STARTING ADDRESS OF TEST 5
7591	047162	006454		.WORD	TST6+2	STARTING ADDRESS OF TEST 6
7592	047164	007102		.WORD	TST7+2	STARTING ADDRESS OF TEST 7
7593	047166	007470		.WORD	TST10+2	STARTING ADDRESS OF TEST 10
7594	047170	010614		.WORD	TST11+2	STARTING ADDRESS OF TEST 11
7595	047172	011302		.WORD	TST12+2	STARTING ADDRESS OF TEST 12
7596	047174	012166		.WORD	TST13+2	STARTING ADDRESS OF TEST 13
7597	047176	013054		.WORD	TST14+2	STARTING ADDRESS OF TEST 14
7598	047200	014260		.WORD	TST15+2	STARTING ADDRESS OF TEST 15
7599	047202	015136		.WORD	TST16+2	STARTING ADDRESS OF TEST 16
7600	047204	015574		.WORD	TST17+2	STARTING ADDRESS OF TEST 17
7601	047206	016710		.WORD	TST20+2	STARTING ADDRESS OF TEST 20
7602	047210	020024		.WORD	TST21+2	STARTING ADDRESS OF TEST 21
7603	047212	021140		.WORD	TST22+2	STARTING ADDRESS OF TEST 22
7604	047214	022376		.WORD	TST23+2	STARTING ADDRESS OF TEST 23
7605	047216	023266		.WORD	TST24+2	STARTING ADDRESS OF TEST 24
7606	047220	024112		.WORD	TST25+2	STARTING ADDRESS OF TEST 25

```

7607 047222 024644 .WORD TST26+2 ;: STARTING ADDRESS OF TEST 26
7608 047224 025376 .WORD TST27+2 ;: STARTING ADDRESS OF TEST 27
7609 047226 026130 .WORD TST30+2 ;: STARTING ADDRESS OF TEST 30
7610 047230 026662 .WORD TST31+2 ;: STARTING ADDRESS OF TEST 31
7611 047232 027414 .WORD TST32+2 ;: STARTING ADDRESS OF TEST 32
7612 047234 030146 .WORD TST33+2 ;: STARTING ADDRESS OF TEST 33
7613 047236 030446 .WORD TST34+2 ;: STARTING ADDRESS OF TEST 34
7614 047240 031214 .WORD TST35+2 ;: STARTING ADDRESS OF TEST 35
7615 047242 031604 .WORD TST36+2 ;: STARTING ADDRESS OF TEST 36
7616 047244 032430 .WORD TST37+2 ;: STARTING ADDRESS OF TEST 37
7617 047246 033254 .WORD TST40+2 ;: STARTING ADDRESS OF TEST 40
7618 047250 034100 .WORD TST41+2 ;: STARTING ADDRESS OF TEST 41
7619 047252 034724 .WORD TST42+2 ;: STARTING ADDRESS OF TEST 42
7620 047254 035550 .WORD TST43+2 ;: STARTING ADDRESS OF TEST 43
7621 047256 036374 .WORD TST44+2 ;: STARTING ADDRESS OF TEST 44
7622 047260 037220 .WORD TST45+2 ;: STARTING ADDRESS OF TEST 45
7623 047262 040060 .WORD TST46+2 ;: STARTING ADDRESS OF TEST 46
7624 047264 041104 .WORD TST47+2 ;: STARTING ADDRESS OF TEST 47
7625 047266 042130 .WORD TST50+2 ;: STARTING ADDRESS OF TEST 50
7626 047270 042770 .WORD TST51+2 ;: STARTING ADDRESS OF TEST 51
7627 047272 043414 .WORD TST52+2 ;: STARTING ADDRESS OF TEST 52
7628 047274 044040 .WORD TST53+2 ;: STARTING ADDRESS OF TEST 53
7629 ;: *****
7630 ;:SBTTL LOOP ON INTERNAL ERROR
7631
7632 047276 032777 001000 131634 SCOP1$: BIT #SW9,2SWR ;:CHECK IF LOOP ON ERROR
7633 047304 001405 BEQ $$ ;:NO RETURN
7634 047306 105737 001103 TSTB $ERFLG ;:CHECK IF ERROR OCCURED
7635 047312 001402 BEQ $$ ;:NO, RETURN
7636 047314 013716 001110 MOV $LPERR,(SP) ;:GO BACK TO BEGINNING OF LOOP
7637 047320 000002 SS: RTI ;:RETURN
7638 ;:SBTTL APT COMMUNICATIONS ROUTINE
7639
7640 ;: *****
7641 047322 112737 000001 047566 $ATY1: MOVB #1,$FFLG ;: TO REPORT FATAL ERROR
7642 047330 112737 000001 047564 $ATY3: MOVB #1,$MFLG ;: TO TYPE A MESSAGE
7643 047336 000403 BR $ATYC
7644 047340 112737 000001 047566 $ATY4: MOVB #1,$FFLG ;: TO ONLY REPORT FATAL ERROR
7645 047346 $ATYC:
7646 047346 010046 MOV RO,-(SP) ;: PUSH RO ON STACK
7647 047350 010146 MOV RI,-(SP) ;: PUSH RI ON STACK
7648 047352 105737 047564 TSTB $MFLG ;: SHOULD TYPE A MESSAGE?
7649 047356 001450 BEQ $$ ;: IF NOT: BR
7650 047360 122737 000001 001234 CMPB #APTENV,$ENV ;: OPERATING UNDER APT?
7651 047366 001031 BNE 3$ ;: IF NOT: BR
7652 047370 132737 000100 001235 BITB #APTSPool,$ENV ;: SHOULD SPOOL MESSAGES?
7653 047376 001425 BEQ 3$ ;: IF NOT: BR
7654 047400 017600 000004 MOV #4(SP),RO ;: GET MESSAGE ADDR.
7655 047404 062766 000002 000004 ADD #2,4(SP) ;: BUMP RETURN ADDR.
7656 047412 005737 001214 1$: TST $MSGTYPE ;: SEE IF DONE W/ LAST XMISSION?
7657 047416 001375 BNE 1$ ;: IF NOT: WAIT
7658 047420 010037 001230 MOV RO,$MSGAD ;: PUT ADDR IN MAILBOX
7659 047424 105720 2$: TSTB (RO)+ ;: FIND END OF MESSAGE
7660 047426 001376 BNE 2$
7661 047430 163700 001230 SUB $MSGAD,RO ;: SUB START OF MESSAGE
7662 047434 006200 ASR RO ;: GET MESSAGE LNTH IN WORDS

```

```

7663 047436 010037 001232      MOV      RO,$MSGLGT      ;;PUT LENGTH IN MAILBOX
7664 047442 012737 000004 001214      MOV      #4,$MSGTYPE    ;;TELL APT TO TAKE MSG.
7665 047450 000413                BR        5$
7666 047452 017637 000004 047476 3$:      MOV      24(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
7667 047460 062766 000002 000004      ADD      #2,4(SP)      ;;BUMP RETURN ADDRESS
7668 047466 013746 177776      MOV      177776,-(SP)  ;;PUSH 177776 ON STACK
7669 047472 004737 050330      JSR      PC,$TYPE      ;;CALL TYPE MACRO
7670 047476 000000                .WORD    0
7671 047500                5$:
7672 047500 105737 047566 10$:      TSTB     $FFLG          ;;SHOULD REPORT FATAL ERROR?
7673 047504 001416                BEQ      12$            ;;IF NOT: BR
7674 047506 005737 001234                TST     $ENV           ;;RUNNING UNDER APT?
7675 047512 001413                BEQ      12$            ;;IF NOT: BR
7676 047514 005737 001214 11$:      TST     $MSGTYPE      ;;FINISHED LAST MESSAGE?
7677 047520 001375                BNE     11$            ;;IF NOT: WAIT
7678 047522 017637 000004 001216      MOV      24(SP),$FATAL ;;GET ERROR #
7679 047530 062766 000002 000004      ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
7680 047536 005237 001214      INC     $MSGTYPE      ;;TELL APT TO TAKE ERROR
7681 047542 105037 047566 12$:      CLRB    $FFLG          ;;CLEAR FATAL FLAG
7682 047546 105037 047565      CLRB    $LFLG         ;;CLEAR LOG FLAG
7683 047552 105037 047564      CLRB    $MFLG         ;;CLEAR MESSAGE FLAG
7684 047556 012601      MOV     (SP)+,R1      ;;POP STACK INTO R1
7685 047560 012600      MOV     (SP)+,RO      ;;POP STACK INTO RO
7686 047562 000207      RTS     PC            ;;RETURN
7687 047564      000      $MFLG: .BYTE      0      ;;MESSG. FLAG
7688 047565      000      $LFLG: .BYTE      0      ;;LOG FLAG
7689 047566      000      $FFLG: .BYTE      0      ;;FATAL FLAG
7690                .EVEN
7691                APTSIZE=200
7692                APTENV=001
7693                APTSPool=100
7694                APTCSUP=040
7695                .SBTTL  ERROR HANDLER ROUTINE
7696
7697                ;;*****
7698                ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
7699                ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7700                ;;*AND GO TO TYPERR ON ERROR
7701                ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7702                ;;*SW15=1      HALT ON ERROR
7703                ;;*SW13=1      INHIBIT ERROR TYPEOUTS
7704                ;;*SW10=1      BELL ON ERROR
7705                ;;*SW09=1      LOOP ON ERROR
7706                ;;*CALL
7707                ;;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
7708
7709                $ERROR:
7710 047570 104407      CKSWR
7711 047572 105237 001103 7$:      INCB    $ERFLG        ;;TEST FOR CHANGE IN SOFT-SWR
7712 047576 001775                BEQ     7$            ;;SET THE ERROR FLAG
7713 047600 013777 001102 131334      MOV     $STNM,$DISPLAY ;;DON'T LET THE FLAG GO TO ZERO
7714 047606 032777 002000 131324      BIT     #BIT10,$SWR  ;;DISPLAY TEST NUMBER AND ERROR FLAG
7715 047614 001402                BEQ     1$            ;;BELL ON ERROR?
7716 047616 104401 001204                TYPE   $BELL         ;;NO - SKIP
7717 047622 005237 001112                INC     $ERTTL        ;;RING BELL
7718 047626 011637 001116 1$:      INC     $ERTTL        ;;COUNT THE NUMBER OF ERRORS
                          MOV     (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION

```

```

7719 047632 162737 000002 001116 SUB #2,$ERRPC
7720 047640 117737 131252 001114 MOVB #2,$ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
7721 047646 032777 020000 131264 BIT #BIT13,$SWR ;;SKIP TYPEOUT IF SET
7722 047654 001004 BNE 20$ ;;SKIP TYPEOUTS
7723 047656 004737 047770 JSR PC,TYPERR ;;GO TO USER ERROR ROUTINE
7724 047662 104401 001211 TYPE ,$CRLF
7725 047666 20$: 20$:
7726 047666 122737 000001 001234 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
7727 047674 001007 BNE 2$ ;;NO SKIP APT ERROR REPORT
7728 047676 113737 001114 047710 MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
7729 047704 004737 047340 JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT
7730 047710 000 .BYTE 0 21$:
7731 047711 000 .BYTE 0 21$:
7732 047712 000777 22$: BR 22$ ;;APT ERROR LOOP
7733 047714 005777 131220 2$: TST #SWR ;;HALT ON ERROR
7734 047720 100002 BPL 3$ ;;SKIP IF CONTINUE
7735 047722 000000 HALT ;;HALT ON ERROR!
7736 047724 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
7737 047726 032777 001000 131204 3$: BIT #BIT09,$SWR ;;LOOP ON ERROR SWITCH SET?
7738 047734 001402 BEQ 4$ ;;BR IF NO
7739 047736 013716 001110 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
7740 047742 005737 001202 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
7741 047746 001402 BEQ 5$ ;;BR IF NONE
7742 047750 013716 001202 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
7743 047754 5$:
7744 047754 022737 044666 000042 CMP #SENDAD,#42 ;;ACT-11 AUTO-ACCEPT?
7745 047762 001001 BNE 6$ ;;BRANCH IF NO
7746 047764 000000 HALT ;;YES
7747 047766 6$:
7748 047766 000002 6$: RTI ;;RETURN
7749
7750 ;:*****
7751 ;SBTTL TYPE ERROR ROUTINE
7752 ;*ENTRY JSR PC,TYPERR
7753 ;*RETURN RTS PC
7754 ;*
7755 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
7756 ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
7757 ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
7758 ;*THE ERROR.
7759 ;:*****
7760 TYPERR: SAVREG
7761 047772 113700 001114 MOVB $ITEMB,RO ;ENTER ERROR NUMBER
7762 047776 042700 177400 BIC #177400,RO ;CLEAR UNUSED BITS
7763 050002 005300 DEC RO ;FORM INDEX FOR ERROR TABLE
7764 050004 006300 ASL RO
7765 050006 006300 ASL RO
7766 050010 006300 ASL RO
7767 050012 062700 001300 1$: ADD #ERRTB,RO ;FORM ADDRESS OF ERROR ENTRY
7768 050016 012037 050032 MOV (RO)+,2$ ;GET EM POINTER
7769 050022 001404 BEQ 3$ ;BRANCH IF THERE ISN'T ONE
7770 050024 104401 001211 TYPE , $CRLF ;TYPE CARRIAGE RETURN LINE FEED
7771 050030 104401 TYPE ;TYPE ERROR MESSAGE (EM)
7772 050032 000000 2$: .WORD 0 ;EM POINTER GOES HERE
7773 050034 012037 050050 3$: MOV (RO)+,4$ ;GET DH POINTER
7774 050040 001404 BEQ 5$ ;BRANCH IF THERE ISN'T ONE

```

7775	050042	104401	001211		TYPE	, \$CRLF		; TYPE CR-LF
7776	050046	104401			TYPE			; TYPE DATA HEADER
7777	050050	000000		45:	.WORD	0		; DH POINTER GOES HERE
7778	050052	012001		55:	MOV	(R0)+, R1		; GET DT POINTER
7779	050054	001445			BEQ	20\$; BRANCH IF THERE ARE NONE
7780	050056	005004			CLR	R4		; RESET INDENT SWITCH
7781	050060	012000			MOV	(R0)+, R0		; GET DF POINTER
7782	050062	012002			MOV	(R0)+, R2		; STORE NUMBER OF DH'S
7783	050064	104401	001211		TYPE	, \$CRLF		
7784	050070	112003		105:	MOVW	(R0)+, R3		; GET & STORE NUMBER OF DATA WORDS
7785	050072	105720			TSTB	(R0)+		; BUMP PAST FORMAT WORD
7786	050074	005703			TST	R3		; TEST IF ANY DATA FOR THIS HEADER
7787	050076	001416			BEQ	14\$; NO - SKIP DATA PRINT
7788	050100	005704			TST	R4		; CHECK IF INDENT WORDS
7789	050102	001004			BNE	12\$; YES, GO INDENT
7790	050104	013146		115:	MOV	2(R1)+, -(SP)		; PUT FIRST DATA WORD ON STACK
7791	050106	104402			TYPOC			; TYPE IT
7792	050110	005303			DEC	R3		; MORE DATA WORDS
7793	050112	001403			BEQ	13\$; NO-BRANCH
7794	050114	104401	054714	125:	TYPE	, SPACE2		; TYPE SEPARATORS
7795	050120	000771			BR	11\$; LOOP
7796	050122	104401	001211	135:	TYPE	, \$CRLF		; TYPE <CR><LF>
7797	050126	005710			TST	(R0)		; CHECK IF NEXT HEADER AVAILABLE
7798	050130	001401			BEQ	14\$; NO, DO NOT CHANGE INDENT
7799	050132	005104			COM	R4		; CHANGE INDENT
7800	050134	005302		145:	DEC	R2		; MORE DH'S?
7801	050136	003414			BLE	20\$; NO-BRANCH
7802	050140	012037	050160	155:	MOV	(R0)+, 18\$; GET NEXT DH POINTER
7803	050144	001751			BEQ	10\$; IF NO HEADER GET DATA
7804	050146	005704			TST	R4		; INDENT?
7805	050150	001402			BEQ	17\$; NO-BRANCH
7806	050152	104401	054714		TYPE	, SPACE2		; INDENT
7807	050156	104401		175:	TYPE			; TYPE DH
7808	050160	000000		185:	.WORD	0		; DH POINTER GOES HERE
7809	050162	104401	001211		TYPE	, \$CRLF		
7810	050166	000740			BR	10\$; LOOP
7811	050170	104414		205:	RESREG			
7812	050172	005237	003610		INC	ERRCNT		; INCREMENT ERROR COUNT
7813	050176	032777	010000	130734	BIT	#SW12, 2SWR		; CHECK IF ABORT AFTER 20 ERRORS
7814	050204	001421			BEQ	25\$; NO, RETURN
7815	050206	022737	000024	003610	CMP	#20., ERRCNT		; CHECK IF ERROR THRESHOLD EXCEEDED
7816	050214	103015			BHS	25\$; NO, RETURN
7817	050216	104401	054717		TYPE	, ABORT		; TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
7818								; ERROR THRESHOLD EXCEEDED"
7819	050222	005737	000042		TST	42		; CHECK IF IN CHAIN MODE
7820	050226	001407			BEQ	30\$; NO, HALT
7821	050230	012737	000001	044524	MOV	#1, \$EOPCT		; FORCE END OF PASS COUNT TO ONE FOR ABORT
7822	050236	012706	001100		MOV	#STACK, SP		; INITIALIZE STACK
7823	050242	000137	044476		JMP	\$EOP		; BRING IN NEXT PROGRAM IN CHAIN
7824	050246	000000		305:	HALT			
7825	050250	000207		255:	RTS	PC		
7826								
7827					.SBTTL	CONTROLLED PROGRAM HALT		
7828	050252	013702	001270		CTRHLT:	MOV \$BASE, R2		; SET '611 BASE
7829	050256	012762	100000	000000	MOV	#CLR, RKCS1(R2)		; CLEAR CONTROLLER
7830	050264	012706	001100		MOV	#STACK, SP		; CLEAR STACK

```

7831 050270 104401 054650          TYPE      ,OPR007          ;TYPE HALT MESSAGE
7832 050274 005737 000042          TST        42              ;TEST IF MONITOR PRESENT
7833 050300 001410                BEQ        5$              ;NO - SKIP
7834 050302 105037 001103          CLRB      $ERFLG          ;CLEAR ERROR FLAG
7835 050306 005037 001202          CLR      $ESCAPE          ;CLEAR ESCAPE
7836 050312 005037 044524          CLR      $EOPCT          ;SET PASS COUNT TO 0
7837 050316 000137 044476          JMP      $EOP              ;GO TO END OF PASS
7838
7839 050322 000000                5$:      HALT              ;HALT PROGRAM
7840 050324 000137 003666          JMP      START1           ;DO RESTART IF CONTINUE
7841
7842          .SBTTL  TYPE ROUTINE
7843
7844          ;*****
7845          ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7846          ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7847          ;NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7848          ;NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7849          ;NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7850          ;
7851          ;CALL:
7852          ;*1) USING A TRAP INSTRUCTION
7853          ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7854          ;*OR
7855          ;*      TYPE
7856          ;*      MESADR
7857          ;*
7858
7859 050330 105737 001157                $TYPE:  TSTB      $TFPLG          ; IS THERE A TERMINAL?
7860 050334 100002                BPL      1$              ; BR IF YES
7861 050336 000000                HALT              ; HALT HERE IF NO TERMINAL
7862 050340 000430                BR        3$              ; LEAVE
7863 050342 010046                1$:      MOV      R0, -(SP)          ; SAVE R0
7864 050344 017600 000002                MOV      22(SP), R0        ; GET ADDRESS OF ASCIZ STRING
7865 050350 122737 000001 001234          CMPB     $APTENV, $ENV      ; RUNNING IN APT MODE
7866 050356 001011                BNE      62$              ; NO, GO CHECK FOR APT CONSOLE
7867 050360 132737 000100 001235          BITB     $APTPOOL, $ENVM   ; SPOOL MESSAGE TO APT
7868 050366 001405                BEQ      62$              ; NO, GO CHECK FOR CONSOLE
7869 050370 010037 050400                MOV      R0, 61$          ; SETUP MESSAGE ADDRESS FOR APT
7870 050374 004737 047330                JSR      PC, $ATY3        ; SPOOL MESSAGE TO APT
7871 050400 000000                .WORD    0                ; MESSAGE ADDRESS
7872 050402 132737 000040 001235          61$:    BITB     $APTCSUP, $ENVM ; APT CONSOLE SUPPRESSED
7873 050410 001003                BNE      60$              ; YES, SKIP TYPE OUT
7874 050412 112046                2$:      MOVB     (R0)+, -(SP)    ; PUSH CHARACTER TO BE TYPED ONTO STACK
7875 050414 001005                BNE      4$              ; BR IF IT ISN'T THE TERMINATOR
7876 050416 005726                TST     (SP)+              ; IF TERMINATOR POP IT OFF THE STACK
7877 050420 012600                60$:    MOV      (SP)+, R0        ; RESTORE R0
7878 050422 062716 000002                3$:      ADD      #2, (SP)          ; ADJUST RETURN PC
7879 050426 000002                RTI                          ; RETURN
7880 050430 122716 000011                4$:      CMPB     $HT, (SP)          ; BRANCH IF <HT>
7881 050434 001430                BEQ      8$              ;
7882 050436 122716 00020                CMPB     $CRLF, (SP)      ;; BRANCH IF NCT <CRLF>
7883 050442 001006                BNE      5$              ;
7884 050444 005726                TST     (SP)+              ; POP <CR><LF> EQUIV
7885 050446 104401                TYPE                          ; TYPE A CR AND LF
7886 050450 001211                $CRLF

```

```

7887 050452 105037 050606          CLRB  $CHARCNT      ;; CLEAR CHARACTER COUNT
7888 050456 000755                   BR      2$          ;; GET NEXT CHARACTER
7889 050460 004737 050542          5$: JSR   PC,$TYPEC  ;; GO TYPE THIS CHARACTER
7890 050464 123726 001156          6$: CMPB $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
7891 050470 001350                   BNE    2$          ;; IF NO GO GET NEXT CHAR.
7892 050472 013746 001154          MOV   $NULL,-(SP)  ;; GET # OF FILLER CHARS. NEEDED
7893                                     AND THE NULL CHAR.
7894 050476 105366 000001          7$: DECB 1(SP)     ;; DOES A NULL NEED TO BE TYPED?
7895 050502 002770                   BLT   6$          ;; BR IF NO--GO POP THE NULL OFF OF STACK
7896 050504 004737 050542          JSR   PC,$TYPEC  ;; GO TYPE A NULL
7897 050510 105337 050606          DECB  $CHARCNT    ;; DO NOT COUNT AS A COUNT
7898 050514 000770                   BR    7$          ;; LOOP

```

:HORIZONTAL TAB PROCESSOR

```

7902 050516 112716 000040          8$: MOVB #' (SP)   ;; REPLACE TAB WITH SPACE
7903 050522 004737 050542          9$: JSR   PC,$TYPEC  ;; TYPE A SPACE
7904 050526 132737 000007 050606  BITB  #7,$CHARCNT  ;; BRANCH IF NOT AT
7905 050534 001372                   BNE   9$          ;; TAB STOP
7906 050536 005726                   TST  (SP)+       ;; POP SPACE OFF STACK
7907 050540 000724                   BR    2$          ;; GET NEXT CHARACTER
7908 050542 105777 130402          $TYPEC: TSTB 2$TPS  ;; WAIT UNTIL PRINTER IS READY
7909 050546 100375                   BPL  $TYPEC
7910 050550 116677 000002 130374  MOVB  2(SP),2$TPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
7911 050556 122766 000015 000002  CMPB  #CR,2(SP)   ;; IS CHARACTER A CARRIAGE RETURN?
7912 050564 001003                   BNE  1$          ;; BRANCH IF NO
7913 050566 105037 050606          CLRB  $CHARCNT    ;; YES--CLEAR CHARACTER COUNT
7914 050572 000406                   BR   $TYPEX
7915 050574 122766 000012 000002  1$: CMPB #LF,2(SP)  ;; IS CHARACTER A LINE FEED?
7916 050602 001402                   BEQ  $TYPEX       ;; BRANCH IF YES
7917 050604 105227                   INCB (PC)+       ;; COUNT THE CHARACTER
7918 050606 000000          $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
7919 050610 000207          $TYPEX: RTS    PC

```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

7921                                     .*****
7922                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A C-DIGIT
7923                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
7924                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7925                                     ;*CALL:
7926                                     ;*
7927                                     ;*   MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
7928                                     ;*   TYPOS    ;; CALL FOR TYPEOUT
7929                                     ;*   .BYTE   N              ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
7930                                     ;*   .BYTE   M              ;; M=1 OR 0
7931                                     ;*
7932                                     ;*                                     ;; 1=TYPE LEADING ZEROS
7933                                     ;*                                     ;; 0=SUPPRESS LEADING ZEROS
7934                                     ;*
7935                                     ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7936                                     ;*$TYPOS OR $TYPOC
7937                                     ;*CALL:
7938                                     ;*   MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
7939                                     ;*   TYPON    ;; CALL FOR TYPEOUT
7940                                     ;*
7941                                     ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7942                                     ;*CALL:

```

H12

```

7943 ;* MOV NUM,-(SP) ;:NUMBER TO BE TYPED
7944 ;* TYPOC ;:CALL FOR TYPEOUT
7945
7946 050612 017646 000000 $TYPOS: MOV 2(SP),-(SP) ;:PICKUP THE MODE
7947 050616 116637 000001 051035 MOVB 1(SP),$OFILL ;:LOAD ZERO FILL SWITCH
7948 050624 112637 051037 MOVB (SP)+,$OMODE+1 ;:NUMBER OF DIGITS TO TYPE
7949 050630 062716 000002 ADD #2,(SP) ;:ADJUST RETURN ADDRESS
7950 050634 000406 $TYPON BR $TYPON
7951 050636 112737 000001 051035 $TYPOC: MOVB #1,$OFILL ;:SET THE ZERO FILL SWITCH
7952 050644 112737 000006 051037 MOVB #6,$OMODE+1 ;:SET FOR SIX(6) DIGITS
7953 050652 112737 000005 051034 $TYPON: MOVB #5,$OCNT ;:SET THE ITERATION COUNT
7954 050660 010346 MOV R3,-(SP) ;:SAVE R3
7955 050662 010446 MOV R4,-(SP) ;:SAVE R4
7956 050664 010546 MOV R5,-(SP) ;:SAVE R5
7957 050666 113704 051037 MOVB $OMODE+1,R4 ;:GET THE NUMBER OF DIGITS TO TYPE
7958 050672 005404 NEG R4
7959 050674 062704 000006 ADD #6,R4 ;:SUBTRACT IT FOR MAX. ALLOWED
7960 050700 110437 051036 MOVB R4,$OMODE ;:SAVE IT FOR USE
7961 050704 113704 051035 MOVB $OFILL,R4 ;:GET THE ZERO FILL SWITCH
7962 050710 016605 000012 MOV 12(SP),R5 ;:PICKUP THE INPUT NUMBER
7963 050714 005003 CLR R3 ;:CLEAR THE OUTPUT WORD
7964 050716 006105 1$: ROL R5 ;:ROTATE MSB INTO "C"
7965 050720 000404 BR 3$ ;:GO DO MSB
7966 050722 006105 2$: ROL R5 ;:FORM THIS DIGIT
7967 050724 006105 ROL R5
7968 050726 006105 ROL R5
7969 050730 010503 MOV R5,R3
7970 050732 006103 3$: ROL R3 ;:GET LSB OF THIS DIGIT
7971 050734 105337 051036 DECB $OMODE ;:TYPE THIS DIGIT?
7972 050740 100016 BPL 7$ ;:BR IF NO
7973 050742 042703 177770 BIC #177770,R3 ;:GET RID OF JUNK
7974 050746 001002 BNE 4$ ;:TEST FOR 0
7975 050750 005704 TST R4 ;:SUPPRESS THIS 0?
7976 050752 001403 BEQ 5$ ;:BR IF YES
7977 050754 005204 4$: INC R4 ;:DON'T SUPPRESS ANYMORE 0'S
7978 050756 052703 000060 BIS #'0,R3 ;:MAKE THIS DIGIT ASCII
7979 050762 052703 000040 5$: BIS #' ,R3 ;:MAKE ASCII IF NOT ALREADY
7980 050766 110337 051032 MOVB R3,8$ ;:SAVE FOR TYPING
7981 050772 104401 051032 TYPE 8$ ;:GO TYPE THIS DIGIT
7982 050776 105337 051034 7$: DECB $OCNT ;:COUNT BY 1
7983 051002 003347 BGT 2$ ;:BR IF MORE TO DO
7984 051004 002402 BLT 6$ ;:BR IF DONE
7985 051006 005204 INC R4 ;:INSURE LAST DIGIT ISN'T A BLANK
7986 051010 000744 BR 2$ ;:GO DO THE LAST DIGIT
7987 051012 012605 6$: MOV (SP)+,R5 ;:RESTORE R5
7988 051014 012604 MOV (SP)+,R4 ;:RESTORE R4
7989 051016 012603 MOV (SP)+,R3 ;:RESTORE R3
7990 051020 016666 000002 000004 MOV 2(SP),4(SP) ;:SET THE STACK FOR RETURNING
7991 051026 012616 MOV (SP)+,(SP)
7992 051030 000002 RTI ;:RETURN
7993 051032 000 8$: .BYTE 0 ;:STORAGE FOR ASCII DIGIT
7994 051033 000 .BYTE 0 ;:TERMINATOR FOR TYPE ROUTINE
7995 051034 000 $OCNT: .BYTE 0 ;:OCTAL DIGIT COUNTER
7996 051035 000 $OFILL: .BYTE 0 ;:ZERO FILL SWITCH
7997 051036 000000 $OMODE: .WORD 0 ;:NUMBER OF DIGITS TO TYPE
7998 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
  
```

```

7999
8000
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010 051040
8011 051040 010046
8012 051042 010146
8013 051044 010246
8014 051046 010346
8015 051050 010546
8016 051052 012746 020200
8017 051056 016605 000020
8018 051062 100004
8019 051064 005405
8020 051066 112766 000055 000001
8021 051074 005000
8022 051076 012703 051254
8023 051102 112723 000040
8024 051106 005002
8025 051110 016001 051244
8026 051114 160105
8027 051116 002402
8028 051120 005202
8029 051122 000774
8030 051124 060105
8031 051126 005702
8032 051130 001002
8033 051132 105716
8034 051134 100407
8035 051136 106316
8036 051140 103003
8037 051142 116663 000001 177777
8038 051150 052702 000060
8039 051154 052702 000040
8040 051160 110223
8041 051162 005720
8042 051164 020027 000010
8043 051170 002746
8044 051172 003002
8045 051174 010502
8046 051176 000764
8047 051200 105726
8048 051202 100003
8049 051204 116663 177777 177776
8050 051212 105013
8051 051214 012605
8052 051216 012603
8053 051220 012602
8054 051222 012601

```

```

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

$TYPDS:
MOV     R0,-(SP)      ;;PUSH R0 ON STACK
MOV     R1,-(SP)      ;;PUSH R1 ON STACK
MOV     R2,-(SP)      ;;PUSH R2 ON STACK
MOV     R3,-(SP)      ;;PUSH R3 ON STACK
MOV     R5,-(SP)      ;;PUSH R5 ON STACK
MOV     #20200,-(SP)   ;;SET BLANK SWITCH AND SIGN
MOV     20(SP),R5      ;;GET THE INPUT NUMBER
BPL     1$             ;;BR IF INPUT IS POS.
NEG     R5             ;;MAKE THE BINARY NUMBER POS.
MOVB    #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
1$:    CLR     R0      ;;ZERO THE CONSTANTS INDEX
MOV     #SDBLK,R3     ;;SETUP THE OUTPUT POINTER
MOVB    #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
2$:    CLR     R2      ;;CLEAR THE BCD NUMBER
MOV     $DTBL(R0),R1  ;;GET THE CONSTANT
3$:    SUB     R1,R5   ;;FORM THIS BCD DIGIT
BLT     4$           ;;BR IF DONE
INC     R2           ;;INCREASE THE BCD DIGIT BY 1
4$:    ADD     R1,R5   ;;ADD BACK THE CONSTANT
TST     R2           ;;CHECK IF BCD DIGIT=0
BNE     5$           ;;FALL THROUGH IF 0
TSTB   (SP)         ;;STILL DOING LEADING 0'S?
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
MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
9$:    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

```

```

8055 051224 012600          MOV      (SP)+,RO          ;;POP STACK INTO RO
8056 051226 104401 051254   TYPE      $DBLK           ;;NOW TYPE THE NUMBER
8057 051232 016666 000002 000004 MOV      2(SP),4(SP)      ;;ADJUST THE STACK
8058 051240 012616          MOV      (SP)+,(SP)
8059 051242 000002          RTI                       ;;RETURN TO USER
8060 051244 023420          $OTBL: 10000.
8061 051246 001750          1000.
8062 051250 000144          100.
8063 051252 000012          10.
8064 051254 000004          $DBLK: .BLKW 4
8065                                .SBTTL TTY INPUT ROUTINE
8066
8067                                ;:*****
8068                                ;ENABL LSB
8069 051264 000000          $TKCNT: .WORD 0          ;;NUMBER OF ITEMS IN QUEUE
8070 051266 000000          $TKQIN: .WORD 0         ;;INPUT POINTER
8071 051270 000000          $TKQOUT: .WORD 0        ;;OUTPUT POINTER
8072 051272 000001          $TKQSRT: .BLKB 1       ;;TTY KEYBOARD QUEUE
8073                                $TKQEND=.
8074                                .EVEN
8075
8076                                ;*TK INITIALIZE ROUTINE
8077                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
8078                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
8079
8080                                ;*CALL:
8081                                ;* JSR PC,$TKINT
8082                                ;* RETURN
8083
8084 051274 005037 051264          $TKINT: CLR      $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
8085 051300 012737 051272 051266 MOV      $TKQSRT,$TKQIN     ;;MOVE THE STARTING ADDRESS OF THE
8086 051306 013737 051266 051270 MOV      $TKQIN,$TKQOUT    ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
8087 051314 012737 051344 000060 MOV      $TKSRV,$TKVEC    ;;INITIALIZE THE KEYBOARD VECTOR
8088 051322 012737 000200 000062 MOV      #200,$TKVEC+2    ;;"BR" LEVEL 4
8089 051330 005777 127612          TST      $TKB            ;;CLEAR DONE FLAG
8090 051334 012777 000100 127602 MOV      #100,$TKS        ;;ENABLE TTY KEYBOARD INTERRUPT
8091 051342 000207          RTS      PC              ;;RETURN TO CALLER
8092
8093                                ;*TK SERVICE ROUTINE
8094                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
8095                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
8096                                ;*IT IN THE QUEUE.
8097                                ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
8098                                ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
8099
8100 051344 117746 127576          $TKSRV: MOV      $TKB,-(SP)  ;;PICKUP THE CHARACTER
8101 051350 042716 177600          BIC      #↑C17,(SP)      ;;STRIP THE JUNK
8102 051354 021627 000003          CMP      (SP),#3         ;;IS IT A CONTROL C?
8103 051360 001007          BNE      IS              ;;BRANCH IF NO
8104 051362 104401 052460          TYPE      $CNTLC        ;;TYPE A CONTROL-C (↑C)
8105 051366 004737 051274          JSR      PC,$TKINT       ;;INIT THE KEYBOARD
8106 051372 005726          TST      (SP)+          ;;CLEAN UP STACK
8107 051374 000137 050252          JMP      CTRHLT          ;;CONTROL C RESTART
8108 051400 021627 000007          IS:  CMP      (SP),#7    ;;IS IT A CONTROL G?
8109 051404 001004          BNE      2$             ;;BRANCH IF NO
8110 051406 022737 000176 001140          CMP      $SWREG,$SWR     ;;IS SOF" SWR SELECTED?

```

```

8111 051414 001500          BEQ      6$          ;;GO TO SWR CHANGE
8112
8113 051416          2$:
8114 051416 022737 000001 051264  CMP      #1,STKCNT      ;; IS THE QUEUE FULL?
8115 051424 001004          BNE      3$          ;; BRANCH IF NO
8116 051426 104401 001204  TYPE     $BELL        ;; RING THE TTY BELL
8117 051432 005726          TST     (SP)+         ;; CLEAN CHARACTER OFF OF STACK
8118 051434 000451          BR       5$          ;; EXIT
8119 051436 021627 000023          3$:  CMP     (SP),#23      ;; IS IT A CONTROL-S?
8120 051442 001021          BNE     32$         ;; BRANCH IF NO
8121 051444 005077 127474  CLR      2$TKS       ;; DISABLE TTY KEYBOARD INTERRUPTS
8122 051450 005726          TST     (SP)+         ;; CLEAN CHAR OFF STACK
8123 051452 105777 127466          31$:  TSTB   2$TKS       ;; WAIT FOR A CHAR
8124 051456 100375          BPL     31$        ;; LOOP UNTIL ITS THERE
8125 051460 117746 127462  MOVB    2$TKB, -(SP)  ;; GET THE CHARACTER
8126 051464 042716 177600  BIC     #1C17?, (SP)  ;; MAKE IT 7-BIT ASCII
8127 051470 022627 000021  CMP     (SP)+, #21    ;; IS IT A CONTROL-Q?
8128 051474 001366          BNE     31$        ;; BRANCH IF NO
8129 051476 012777 000100 127440  MOV     #100, 2$TKS  ;; REENABLE TTY KEYBOARD INTERRUPTS
8130 051504 000002          RTI
8131 051506 005237 051264          32$:  INC     $TKCNT       ;; COUNT THIS CHARACTER
8132 051512 021627 000140  CMP     (SP), #140   ;; IS IT UPPER CASE?
8133 051516 002405          BLT     4$          ;; BRANCH IF YES
8134 051520 021627 000175  CMP     (SP), #175   ;; IS IT A SPECIAL CHAR?
8135 051524 003002          BGT     4$          ;; BRANCH IF YES
8136 051526 042716 000040  BIC     #40, (SP)    ;; MAKE IT UPPER CASE
8137 051532 112677 177530          4$:  MOVB   (SP)+, 2$TKQIN  ;; AND PUT IT IN QUEUE
8138 051536 005237 051266  INC     $TKQIN       ;; UPDATE THE POINTER
8139 051542 023727 051266 051273  CMP     $TKQIN, #2$TKQEND  ;; GO OFF THE END?
8140 051550 001003          BNE     5$          ;; BRANCH IF NO
8141 051552 012737 051272 051266  MOV     #2$TKQSR, $TKQIN  ;; RESET THE POINTER
8142 051560 000002          5$:  RTI
8143
8144
8145
8146
8147
8148
8149 051562 022737 000176 001140  $CKSWR:  CMP     #SWREG, SWR  ;; IS THE SOFT-SWR SELECTED
8150 051570 001124          BNE     15$         ;; EXIT IF NOT
8151 051572 105777 127346  TSTB   2$TKS       ;; IS A CHAR WAITING?
8152 051576 100121          BPL     15$        ;; IF NOT, EXIT
8153 051600 117746 127342  MOVB   2$TKB, -(SP)  ;; YES
8154 051604 042716 177600  BIC     #1C17?, (SP)  ;; MAKE IT 7-BIT ASCII
8155 051610 021627 000007  CMP     (SP), #7     ;; IS IT A CONTROL-G?
8156 051614 001300          BNE     25$        ;; IF NOT, PUT IT IN THE TTY QUEUE
8157
8158
8159
8160
8161
8162
8163 051616 123727 001134 000001  6$:  CMPB   $AUTOB, #1    ;; ARE WE RUNNING IN AUTO-MODE?
8164 051624 001674          BEQ     25$        ;; BRANCH IF YES
8165 051626 005726          TST     (SP)+         ;; CLEAR CONTROL-G OFF STACK
8166 051630 004737 051274          JSR     PC, $TKINT   ;; FLUSH THE TTY INPUT QUEUE

```

```

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

```

```

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

```

8167	051634	005077	127304		CLR	2STKS	:: DISABLE TTY KEYBOARD INTERRUPTS
8168	051640	112737	000001	001135	MOVB	#1, \$INTAG	:: SET INTERRUPT MODE INDICATOR
8169							
8170	051646	104401	052472		TYPE	, \$CNTLG	:: ECHO THE CONTROL-G (↑G)
8171	051652	104401	052477		TYPE	, \$MSWR	:: TYPE CURRENT CONTENTS
8172	051656	013746	000176		MOV	\$WREG, -(SP)	:: SAVE SWREG FOR TYPEOUT
8173	051662	104402			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
8174	051664	104401	052510		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR
8175	051670	005046			CLR	-(SP)	:: CLEAR COUNTER
8176	051672	005046			CLR	-(SP)	:: THE NEW SWR
8177	051674	105777	127244		TSTB	2STKS	:: CHAR THERE?
8178	051700	100375			BPL	7S	:: IF NOT TRY AGAIN
8179							
8180	051702	117746	127240		MOVB	2STKB, -(SP)	:: PICK UP CHAR
8181	051706	042716	177600		BIC	#1C177, (SP)	:: MAKE IT 7-BIT ASCII
8182							
8183	051712	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?
8184	051716	001015			BNE	9S	:: BRANCH IF NOT
8185	051720	104401	052460		TYPE	, \$CNTLC	:: YES, ECHO CONTROL-C (↑C)
8186	051724	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
8187	051730	123727	001135	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?
8188	051736	001003			BNE	8S	:: BRANCH IF NO
8189	051740	012777	000100	127176	MOV	#100, 2STKS	:: ALLOW TTY KEYBOARD INTERRUPTS
8190	051746	000137	050252		JMP	CTRHLT	:: CONTROL-C RESTART
8191							
8192							
8193	051752	021627	000025		CMP	(SP), #25	:: IS IT A CONTROL-U?
8194	051756	001005			BNE	10S	:: BRANCH IF NOT
8195	051760	104401	052465		TYPE	, \$CNTLU	:: YES, ECHO CONTROL-U (↑U)
8196	051764	062706	000006		ADD	#6, SP	:: IGNORE PREVIOUS INPUT
8197	051770	000737			BR	19S	:: LET'S TRY IT AGAIN
8198							
8199							
8200	051772	021627	000015		CMP	(SP), #15	:: IS IT A <CR>?
8201	051776	001022			BNE	16S	:: BRANCH IF NO
8202	052000	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
8203	052004	001403			BEQ	11S	:: BRANCH IF YES
8204	052006	016677	000002	127124	MOV	2(SP), 2SWR	:: SAVE NEW SWR
8205	052014	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
8206	052020	104401	001211		TYPE	, \$CRLF	:: ECHO <CR> AND <LF>
8207	052024	123727	001135	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
8208	052032	001003			BNE	15S	:: BRANCH IF NOT
8209	052034	012777	000100	127102	MOV	#100, 2STKS	:: RE-ENABLE TTY KBD INTERRUPTS
8210	052042	000002			RTI		:: RETURN
8211	052044	004737	050542		JSR	PC, \$TYPEC	:: ECHO CHAR
8212	052050	021627	000060		CMP	(SP), #60	:: CHAR < 0?
8213	052054	002420			BLT	18S	:: BRANCH IF YES
8214	052056	021627	000067		CMP	(SP), #67	:: CHAR > 7?
8215	052062	003015			BGT	18S	:: BRANCH IF YES
8216	052064	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
8217	052070	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
8218	052074	001403			BEQ	17S	:: BRANCH IF YES
8219	052076	006316			ASL	(SP)	:: NO, SHIFT PRESENT
8220	052100	006316			ASL	(SP)	:: CHAR OVER TO MAKE
8221	052102	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
8222	052104	005266	000002		INC	2(SP)	:: KEEP COUNT OF CHAR

```

8223 052110 056616 177776          BIS      -2(SP), (SP)      ;; SET IN NEW CHAR
8224 052114 000667                   BR        7$              ;; GET THE NEXT ONE
8225 052116 104401 001210      18$:    TYPE      $QUES      ;; TYPE ?(CR)(LF)
8226 052122 000720                   BR        20$              ;; SIMULATE CONTROL-U
8227                                     .DSABL  LSB
8228
8229
8230                                     ;*****
8231                                     ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8232                                     ;CALL:
8233                                     ;   RDCHR                      ;; GET A CHARACTER FROM THE QUEUE
8234                                     ;   RETURN HERE                ;; CHARACTER IS ON THE STACK
8235                                     ;
8236                                     ;
8237
8238 052124 011646          $RDCHR:  MOV      (SP), -(SP)      ;; PUSH DOWN THE PC AND
8239 052126 016666 000004 000002    MOV      4(SP), 2(SP)      ;; THE PS
8240 052134 005066 000004          CLR      4(SP)              ;; GET READY FOR A CHARACTER
8241 052140 005046          CLR      -(SP)             ;; PUT NEW PS ON STACK
8242 052142 012746 052150          MOV      #64$, -(SP)      ;; PUT NEW PC ON STACK
8243 052146 000002          RTI                    ;; POP NEW PC AND PS
8244 052150
8245 052150 005737 051264      64$:    TST      $STKCNT          ;; WAIT ON A CHARACTER
8246 052154 001775          BEQ      1$
8247 052156 005337 051264      1$:    DEC      $STKCNT          ;; DECREMENT THE COUNTER
8248 052162 117766 177102 000004    MOVB    2($STKQOUT, 4(SP)) ;; GET ONE CHARACTER
8249 052170 005237 051270          INC      $STKQOUT         ;; UPDATE THE POINTER
8250 052174 023727 051270 051273    CMP     $STKQOUT, #($STKQEND) ;; DID IT GO OFF OF THE END?
8251 052202 001003          BNE     2$              ;; BRANCH IF NO
8252 052204 012737 051272 051270    MOV     #($STKQSR, $STKQOUT) ;; RESET THE POINTER
8253 052212 000002          RTI                    ;; RETURN
8254                                     ;*****
8255                                     ;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
8256                                     ;CALL:
8257                                     ;   RDLIN                      ;; INPUT A STRING FROM THE TTY
8258                                     ;   RETURN HERE                ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8259                                     ;   TERMINATOR WILL BE A BYTE OF ALL 0'S
8260
8261 052214 010346          $RDLIN:  MOV      R3, -(SP)      ;; SAVE R3
8262 052216 005046          CLR      -(SP)           ;; CLEAR THE RUBOUT KEY
8263 052220 012703 052450      1$:    MOV      #($TTYIN, R3)      ;; GET ADDRESS
8264 052224 022703 052460      2$:    CMP     #($TTYIN+8., R3)    ;; BUFFER FULL?
8265 052230 101456          BLOS    4$              ;; BR IF YES
8266 052232 104410          RDCHR    ;; GO READ ONE CHARACTER FROM THE TTY
8267 052234 112613          MOVB    (SP)+, (R3)      ;; GET CHARACTER
8268 052236 122713 000177      10$:   CMPB    #177, (R3)        ;; IS IT A RUBOUT
8269 052242 001022          BNE     5$              ;; BR IF NO
8270 052244 005716          TST     (SP)             ;; IS THIS THE FIRST RUBOUT?
8271 052246 001007          BNE     6$              ;; BR IF NO
8272 052250 112737 000134 052446    MOVB    #' \, 9$         ;; TYPE A BACK SLASH
8273 052256 104401 052446          TYPE    9$
8274 052262 012716 177777          MOV     #-1, (SP)        ;; SET THE RUBOUT KEY
8275 052266 005303      6$:    DEC     R3              ;; BACKUP BY ONE
8276 052270 020327 052450          CMP     R3, #($TTYIN)    ;; STACK EMPTY?
8277 052274 103434          BLO    4$              ;; BR IF YES
8278 052276 111337 052446          MOVB    (R3), 9$        ;; SETUP TO TYPEOUT THE DELETED CHAR.

```

```

8279 052302 104401 052446          TYPE      9S          :: GO TYPE
8280 052306 000746          BR        2S          :: GO READ ANOTHER CHAR.
8281 052310 005716          5S: TST      (SP)      :: RUBOUT KEY SET?
8282 052312 001406          BEQ      7S          :: BR IF NO
8283 052314 112737 000134 052446      MOVB     #' \, 9S     :: TYPE A BACK SLASH
8284 052322 104401 052446          TYPE      9S          ::
8285 052326 005016          CLR      (SP)        :: CLEAR THE RUBOUT KEY
8286 052330 122713 000025          7S: CMPB     #25, (R3)  :: IS CHARACTER A CTRL U?
8287 052334 001003          BNE     8S          :: BR IF NO
8288 052336 104401 052465          TYPE     $CNTLU      :: TYPE A CONTROL "U"
8289 052342 000726          BR        1S          :: GO START OVER
8290 052344 122713 000022          8S: CMPB     #22, (R3)  :: IS CHARACTER A "r"?
8291 052350 001011          BNE     3S          :: BRANCH IF NO
8292 052352 105013          CLRB     (R3)        :: CLEAR THE CHARACTER
8293 052354 104401 001211          TYPE     , $CRLF      :: TYPE A "CR" & "LF"
8294 052360 104401 052450          TYPE     $TTYIN      :: TYPE THE INPUT STRING
8295 052364 000717          BR        2S          :: GO PICKUP ANOTHER CHARACTER
8296 052366 104401 001210          4S: TYPE     $QUES      :: TYPE A '?'
8297 052372 000712          BR        1S          :: CLEAR THE BUFFER AND LOOP
8298 052374 111337 052446          3S: MOVB     (R3), 9S   :: ECHO THE CHARACTER
8299 052400 104401 052446          TYPE      9S          ::
8300 052404 122723 000015          CMPB     #15, (R3)+  :: CHECK FOR RETURN
8301 052410 001305          BNE     2S          :: LOOP IF NOT RETURN
8302 052412 105063 177777          CLRB     -1(R3)     :: CLEAR RETURN (THE 15)
8303 052416 104401 001212          TYPE     , $LF       :: TYPE A LINE FEED
8304 052422 005726          TST     (SP)+       :: CLEAN RUBOUT KEY FROM THE STACK
8305 052424 012603          MOV     (SP)+, R3   :: RESTORE R3
8306 052426 011646          MOV     (SP), -(SP) :: ADJUST THE STACK AND PUT ADDRESS OF THE
8307 052430 016666 000004 000002          MOV     4(SP), 2(SP) :: FIRST ASCII CHARACTER ON IT
8308 052436 012766 052450 000004          MOV     # $TTYIN, 4(SP)
8309 052444 000002          RTI          :: RETURN
8310 052446 000          9S: .BYTE     0          :: STORAGE FOR ASCII CHAR. TO TYPE
8311 052447 000          .BYTE     0          :: TERMINATOR
8312 052450 000010          $TTYIN: .BLKB     8.  :: RESERVE 8 BYTES FOR TTY INPUT
8313 052460 041536 005015 000          $CNTLC: .ASCIZ  /?C/<15><12>  :: CONTROL "C"
8314 052465 136 006525 000012          $CNTLU: .ASCIZ  /?U/<15><12>  :: CONTROL "U"
8315 052472 043536 005015 000          $CNTLG: .ASCIZ  /?G/<15><12>  :: CONTROL "G"
8316 052477 015 051412 051127          $MSWR: .ASCIZ  <15><12>/SWR = /
8317 052504 036440 000040          $MNEW: .ASCIZ  / NEW = /
8318 052510 020040 042516 020127
8319 052516 020075 000
8320 052522
8321 .EVEN
8322 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
8323 *****
8324 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
8325 *CHANGE IT TO BINARY.
8326 *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
8327 *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
8328 *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
8329 *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
8330 *CALL:
8331 *      ROOCT          :: READ AN OCTAL NUMBER
8332 *      RETURN HERE   :: LOW ORDER BITS ARE ON TOP OF THE STACK
8333 *                   :: HIGH ORDER BITS ARE IN $HIOCT
8334

```

```

8335 052522 011646          SRDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE
8336 052524 016666 000004 000002 MOV      4(SP), 2(SP)      ;; INPUT NUMBER
8337 052532 010046          MOV      RO, -(SP)        ;; PUSH RO ON STACK
8338 052534 010146          MOV      R1, -(SP)        ;; PUSH R1 ON STACK
8339 052536 010246          MOV      R2, -(SP)        ;; PUSH R2 ON STACK
8340 052540 104411          1$:  RDLIN          ;; READ AN ASCII LINE
8341 052542 012600          MOV      (SP)+, RO        ;; GET ADDRESS OF 1ST CHARACTER
8342 052544 010037 052650 MOV      RO, 5$          ;; AND SAVE IT
8343 052550 005001          CLR      R1              ;; CLEAR DATA WORD
8344 052552 005002          CLR      R2
8345 052554 112046          2$:  MOVVB      (RO)+, -(SP)  ;; PICKUP THIS CHARACTER
8346 052556 001420          BEQ      3$              ;; IF ZERO GET OUT
8347 052560 122716 000060 CMPB     #'0', (SP)      ;; MAKE SURE THIS CHARACTER
8348 052564 003026          BGT      4$              ;; IS AN OCTAL DIGIT
8349 052566 122716 000067 CMPB     #'7', (SP)
8350 052572 002423          BLT      4$
8351 052574 006301          ASL      R1              ;; *2
8352 052576 006102          ROL      R2
8353 052600 006301          ASL      R1              ;; *4
8354 052602 006102          ROL      R2
8355 052604 006301          ASL      R1              ;; *8
8356 052606 006102          ROL      R2
8357 052610 042716 177770 BIC     #'C7', (SP)     ;; STRIP THE ASCII JUNK
8358 052614 062601          ADD     (SP)+, R1       ;; ADD IN THIS DIGIT
8359 052616 000756          BR      2$              ;; LOOP
8360 052620 005726          3$:  TST      (SP)+        ;; CLEAN TERMINATOR FROM STACK
8361 052622 010166 000012 MOV      R1, 12(SP)     ;; SAVE THE RESULT
8362 052626 010237 052660 MOV      R2, $HIOCT
8363 052632 012602          MOV     (SP)+, R2      ;; POP STACK INTO R2
8364 052634 012601          MOV     (SP)+, R1      ;; POP STACK INTO R1
8365 052636 012600          MOV     (SP)+, RO      ;; POP STACK INTO RO
8366 052640 000002          RTI                    ;; RETURN
8367 052642 005726          4$:  TST      (SP)+        ;; CLEAN PARTIAL FROM STACK
8368 052644 105010          CLRB    (RO)           ;; SET A TERMINATOR
8369 052646 104401          TYPE                    ;; TYPE UP THRU THE BAD CHAR.
8370 052650 000000          5$:  .WORD     0
8371 052652 104401 001210 TYPE     $QUES          ;; "?" "CR" & "LF"
8372 052656 000730          BR      1$              ;; TRY AGAIN
8373 052660 000000          $HIOCT: .WORD     0    ;; HIGH ORDER BITS GO HERE
8374          .SBTTL  SAVE AND RESTORE RO-R5 ROUTINES
8375
8376          ;; *****
8377          ;; *SAVE RO-R5
8378          ;; *CALL:
8379          ;; * SAVREG
8380          ;; *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
8381          ;; *
8382          ;; *TOP---(+16)
8383          ;; * +2---(+18)
8384          ;; * +4---R5
8385          ;; * +6---R4
8386          ;; * +8---R3
8387          ;; *+10---R2
8388          ;; *+12---R1
8389          ;; *+14---RO
8390

```

8391 052662
8392 052662 010046
8393 052664 010146
8394 052666 010246
8395 052670 010346
8396 052672 010446
8397 052674 010546
8398 052676 016646 000022
8399 052702 016646 000022
8400 052706 016646 000022
8401 052712 016646 000022
8402 052716 000002
8403
8404
8405
8406
8407 052720
8408 052720 012666 000022
8409 052724 012666 000022
8410 052730 012666 000022
8411 052734 012666 000022
8412 052740 012605
8413 052742 012604
8414 052744 012603
8415 052746 012602
8416 052750 012601
8417 052752 012600
8418 052754 000002
8419
8420
8421
8422
8423
8424
8425 052756 017737 126156 003634
8426 052764 012737 053004 000024
8427 052772 012737 000340 000026
8428 053000 000000
8429 053002 000776
8430
8431
8432
8433
8434 053004 005037 053100
8435 053010 012737 000144 053102
8436 053016 005237 053100
8437 053022 001375
8438 053024 005337 053102
8439 053030 001372
8440 053032 012737 052756 000024
8441 053040 012737 000340 000026
8442 053046 012706 001100
8443 053052 104401 053104
8444 053056 004737 044732
8445 053062 013777 003634 126050
8446 053070 013702 001270

```

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

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

.SBTTL  POWER DOWN AND UP ROUTINE

;*****

:POWER DOWN ROUTINE
$PWRDN: MOV    @SWR,SAVSWR  ;;SAVE SWITCH REGISTER
MOV    @SPWRUP,PWRVEC    ;;SET UP VECTOR
MOV    @PR7,PWRVEC+2
HALT
BR     -2      ;HANG UP

;*****

:POWER UP ROUTINE
$PWRUP: CLR    $PWRCT      ;LOAD WAIT COUNT
MOV    #100,$PWRCT+2
IS:    INC    $PWRCT      ;WAIT FOR TELETYPE
BNE    IS
DEC    $PWRCT+2
BNE    IS
MOV    @SPWRDN,PWRVEC    ;SET UP FOR POWER DOWN VECTOR
MOV    @PR7,PWRVEC+2
MOV    @STACK,SP        ;FORCE STACK
TYPE   $POWER           ;TYPE POWER
JSR    PC,PARCHK        ;REINITIALIZE MEMORY CHECK ENABLE
MOV    SAVSWR,@SWR      ;RESTORE SWITCH REGISTER
MOV    $BASE,R2         ;REINITIALISE R2 FOR '611 BASE

```

```

8447 053074 000177 126006 JMP 2$SLPADR ;GO BACK TO LAST TEST
8448
8449 053100 000000 000000 $PWRT: .WORD 0,0 ;TELETYPE TIME OUT
8450 053104 047520 042527 000122 $POWER: .ASCIZ /POWER/
8451 .EVEN
8452 .SBTTL TRAP DECODER
8453
8454 ;*****
8455 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8456 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8457 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8458 ;*GO TO THAT ROUTINE.
8459
8460 053112 010046 $TRAP: MOV RO,-(SP) ;;SAVE RO
8461 053114 016600 000002 MOV 2(SP),RO ;;GET TRAP ADDRESS
8462 053120 005740 TST -(RO) ;;BACKUP BY 2
8463 053122 111000 MOVB (RO),RO ;;GET RIGHT BYTE OF TRAP
8464 053124 006300 ASL RO ;;POSITION FOR INDEXING
8465 053126 016000 053146 MOV $TRPAD(RO),RO ;;INDEX TO TABLE
8466 053132 000200 RTS RO ;;GO TO ROUTINE
8467
8468
8469 ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8470
8471 053134 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
8472 053136 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
8473 053144 000002 RTI ;;RESTORE THE PSW
8474
8475 .SBTTL TRAP TABLE
8476
8477 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8478 ;*BY THE "TRAP" INSTRUCTION.
8479
8480 : ROUTINE
8481 :-----
8482 053146 053134 $TRPAD: .WORD $TRAP2
8483 053150 050330 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
8484 053152 050636 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8485 053154 050612 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
8486 053156 050652 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
8487 053160 051040 $TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
8488
8489 053162 051652 $GTSWR ;;CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
8490
8491 053164 051562 $CKSWR ;;CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
8492 053166 052124 $RDCHR ;;CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
8493 053170 052214 $RDLIN ;;CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
8494 053172 052522 $RDOCT ;;CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
8495 053174 052662 $SAVREG ;;CALL=SAVREG TRAP+13(104413) SAVE RO-R5 ROUTINE
8496 053176 052720 $RESREG ;;CALL=RESREG TRAP+14(104414) RESTORE RO-R5 ROUTINE
8497 053200 047276 $SCOPI$ ;;CALL=SCOPI TRAP+15(104415) INTERNAL LOOP ON ERROR

```

```

      .SBTTL DATA TABLE FOR PRINT OUT
8498
8499
8500 053202 001220 003604 DT000: .WORD $TESTN,TRAPPC
8501 053206 001220 001116 003500 DT001: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MP3
8502 053214 003440 003526 003466
8503 053222 003530 003470
8504 053226 001220 001116 003500 DT015: .WORD $TESTN,$ERRPC,E.CS1,T.CS1
8505 053234 003440
8506 053236 001220 001116 003524 DT017: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
8507 053244 003464 003614 003616
8508 053252 003622
8509 053254 001220 001116 003500 DT022: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC
8510 053262 003440 003504 003444
8511 053270 003502 003442
8512 053274 001220 001116 003524 DT025: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,BITCNT
8513 053302 003464 003622
8514 053306 001220 001116 003500 DT031: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC,E.MR1,T.MR1
8515 053314 003440 003504 003444
8516 053322 003502 003442 003524
8517 053330 003464
8518 053332 001220 001116 003500 DT035: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.BA,T.BA
8519 053340 003440 003510 003450
8520 053346 003504 003444
8521 053352 003502 003442
8522 053356 001220 001116 003522 DT041: .WORD $TESTN,$ERRPC,E.DB,T.DB
8523 053364 003462
8524 053366 001220 001116 003500 DT042: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8525 053374 003440 003510 003450
8526 053402 001220 001116 003522 DT054: .WORD $TESTN,$ERRPC,E.DB,T.DB,WRDCNT
8527 053410 003462 003624
8528 053414 001220 001116 003500 DT055: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,WRDCNT
8529 053422 003440 003510 003450
8530 053430 003624
8531 053432 001220 001116 003500 DT072: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
8532 053440 003440 003510 003450
8533 053446 003514 003454
8534 053452 003504 003444 003502 .WORD E.BA,T.BA,E.WC,T.WC
8535 053460 003442
8536 053462 001220 001116 003500 DT077: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8537 053470 003440 003510 003450
8538 053476 001220 001116 003500 DT150: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,WRDCNT,BITCNT
8539 053504 003440 003624 003622
8540 053512 001220 001116 003500 DT164: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS1,BITCNT
8541 053520 003440 003510 003440
8542 053526 003622
8543 053530 001220 001116 003524 DT170: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT,SECCNT
8544 053536 003464 003612 003614
8545 053544 003616 003620 003622
8546 053552 003626
8547 053554 001220 001116 003524 DT171: .WORD $TESTN,$ERRPC,E.MR1,T.MR1
8548 053562 003464
8549 053564 001220 001116 003500 DT175: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER
8550 053572 003440 003510 003450
8551 053600 003512 003452 003514
8552 053606 003454
8553 053610 001220 001116 003500 DT211: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER

```

F13

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 160
DZR6CB.P11 28-JAN-77 14:04 DATA TABLE FOR PRINT OUT

SEQ 0160

8554	053616	003440	003510	003450
8555	053624	003512	003452	003514
8556	053632	003454		
8557	053634	003540	003550	003552
8558	053642	003554		

.WORD P.CS1,P.CS2,P.DS,P.ER

```

8559          .SBTTL DATA FORMAT FOR PRINT OUT
8560
8561 053644 000001          DF000: .WORD 1
8562 053646 002          .BYTE 2,0
8563 053650 000005          DF001: .WORD 5          ;ERRORS 1-14
8564 053652 000          .BYTE 0,0
8565 053654 055030          .WORD 04000A
8566 053656 000          .BYTE 0,0
8567 053660 055046          .WORD 04000B
8568 053662 002          .BYTE 2,0
8569 053664 055112          .WORD 04001A
8570 053666 000          .BYTE 0,0
8571 053670 055171          .WORD 04001B
8572 053672 006          .BYTE 6,0
8573 053674 000005          DF015: .WORD 5          ;ERRORS 15-16
8574 053676 000          .BYTE 0,0
8575 053700 055030          .WORD 04000A
8576 053702 000          .BYTE 0,0
8577 053704 055046          .WORD 04000B
8578 053706 002          .BYTE 2,0
8579 053710 055250          .WORD 04015A
8580 053712 000          .BYTE 0,0
8581 053714 055267          .WORD 04015B
8582 053716 002          .BYTE 2,0
8583 053720 000005          DF017: .WORD 5          ;ERROR 17-21
8584 053722 000          .BYTE 0,0
8585 053724 055030          .WORD 04000A
8586 053726 000          .BYTE 0,0
8587 053730 055046          .WORD 04000B
8588 053732 002          .BYTE 2,0
8589 053734 055305          .WORD 04017A
8590 053736 000          .BYTE 0,0
8591 053740 055351          .WORD 04017B
8592 053742 005          .BYTE 5,0
8593 053744 000005          DF022: .WORD 5          ;ERROR 22-24
8594 053746 000          .BYTE 0,0
8595 053750 055030          .WORD 04000A
8596 053752 000          .BYTE 0,0
8597 053754 055046          .WORD 04000B
8598 053756 002          .BYTE 2,0
8599 053760 055417          .WORD 04022A
8600 053762 000          .BYTE 0,0
8601 053764 055476          .WORD 04022B
8602 053766 006          .BYTE 6,0
8603 053770 000005          DF025: .WORD 5          ;ERROR 25
8604 053772 000          .BYTE 0,0
8605 053774 055030          .WORD 04000A
8606 053776 000          .BYTE 0,0
8607 054000 055046          .WORD 04000B
8608 054002 002          .BYTE 2,0
8609 054004 055553          .WORD 04025A
8610 054006 000          .BYTE 0,0
8611 054010 055577          .WORD 04025B
8612 054012 003          .BYTE 3,0
8613 054014 000005          DF031: .WORD 5          ;ERROR 31-34
8614 054016 000          .BYTE 0,0

```

H13

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 162
DZR6CB.P11 28-JAN-77 14:04 DATA FORMAT FOR PRINT OUT

SEQ 0162

8615	054020	055030		.WORD	DH000A	
8616	054022	000	000	.BYTE	0,0	
8617	054024	055046		.WORD	DH000B	
8618	054026	002	000	.BYTE	2,0	
8619	054030	055625		.WORD	DH031A	
8620	054032	000	000	.BYTE	0,0	
8621	054034	055724		.WORD	DH031B	
8622	054036	010	000	.BYTE	8.,0	
8623	054040	000005		.WORD	5	:ERRORS 35-40
8624	054042	000	000	.BYTE	0,0	
8625	054044	055030		.WORD	DH000A	
8626	054046	000	000	.BYTE	0,0	
8627	054050	055046		.WORD	DH000B	
8628	054052	002	000	.BYTE	2,0	
8629	054054	056022		.WORD	DH035A	
8630	054056	000	000	.BYTE	0,0	
8631	054060	056121		.WORD	DH035B	
8632	054062	010	000	.BYTE	8.,0	
8633	054064	000005		.WORD	5	:ERROR 41
8634	054066	000	000	.BYTE	0,0	
8635	054070	055030		.WORD	DH000A	
8636	054072	000	000	.BYTE	0,0	
8637	054074	055046		.WORD	DH000B	
8638	054076	002	000	.BYTE	2,0	
8639	054100	056216		.WORD	DH041A	
8640	054102	000	000	.BYTE	0,0	
8641	054104	056233		.WORD	DH041B	
8642	054106	002	000	.BYTE	2,0	
8643	054110	000005		.WORD	5	:ERRORS 42-43
8644	054112	000	000	.BYTE	0,0	
8645	054114	055030		.WORD	DH000A	
8646	054116	000	000	.BYTE	0,0	
8647	054120	055046		.WORD	DH000B	
8648	054122	002	000	.BYTE	2,0	
8649	054124	056250		.WORD	DH042A	
8650	054126	000	000	.BYTE	0,0	
8651	054130	056307		.WORD	DH042B	
8652	054132	004	000	.BYTE	4,0	
8653	054134	000005		.WORD	5	:ERROR 54
8654	054136	000	000	.BYTE	0,0	
8655	054140	055030		.WORD	DH000A	
8656	054142	000	000	.BYTE	0,0	
8657	054144	055046		.WORD	DH000B	
8658	054146	002	000	.BYTE	2,0	
8659	054150	056345		.WORD	DH054A	
8660	054152	000	000	.BYTE	0,0	
8661	054154	056372		.WORD	DH054B	
8662	054156	003	000	.BYTE	3,0	
8663	054160	000005		.WORD	5	:ERROR 55
8664	054162	000	000	.BYTE	0,0	
8665	054164	055030		.WORD	DH000A	
8666	054166	000	000	.BYTE	0,0	
8667	054170	055046		.WORD	DH000B	
8668	054172	002	000	.BYTE	2,0	
8669	054174	056420		.WORD	DH055A	
8670	054176	000	000	.BYTE	0,0	

8671	054200	056465		.WORD	DH055B	
8672	054202	005	000	.BYTE	5,0	
8673	054204	000007		DF072: .WORD	7	
8674	054206	000	000	.BYTE	0,0	
8675	054210	055030		.WORD	DH000A	
8676	054212	000	000	.BYTE	0,0	
8677	054214	055046		.WORD	DH000B	
8678	054216	002	000	.BYTE	2,0	
8679	054220	056533		.WORD	DH072A	
8680	054222	000	000	.BYTE	0,0	
8681	054224	056572		.WORD	DH072B	
8682	054226	004	000	.BYTE	4,0	
8683	054230	056630		.WORD	DH072C	
8684	054232	000	000	.BYTE	0,0	
8685	054234	056707		.WORD	DH072D	
8686	054236	006	000	.BYTE	6,0	
8687	054240	000007		DF077: .WORD	7	
8688	054242	000	000	.BYTE	0,0	
8689	054244	055030		.WORD	DH000A	
8690	054246	000	000	.BYTE	0,0	
8691	054250	055046		.WORD	DH000B	
8692	054252	002	000	.BYTE	2,0	
8693	054254	056533		.WORD	DH072A	
8694	054256	000	000	.BYTE	0,0	
8695	054260	056572		.WORD	DH072B	
8696	054262	004	000	.BYTE	4,0	
8697	054264	000005		DF150: .WORD	5	:ERROR 150
8698	054266	000	000	.BYTE	0,0	
8699	054270	055030		.WORD	DH000A	
8700	054272	000	000	.BYTE	0,0	
8701	054274	055046		.WORD	DH000B	
8702	054276	002	000	.BYTE	2,0	
8703	054300	056764		.WORD	DH150A	
8704	054302	000	000	.BYTE	0,0	
8705	054304	057020		.WORD	DH150B	
8706	054306	004	000	.BYTE	4,0	
8707	054310	000005		DF164: .WORD	5	:ERROR 164
8708	054312	000	000	.BYTE	0,0	
8709	054314	055030		.WORD	DH000A	
8710	054316	000	000	.BYTE	0,0	
8711	054320	055046		.WORD	DH000B	
8712	054322	002	000	.BYTE	2,0	
8713	054324	057056		.WORD	DH164A	
8714	054326	000	000	.BYTE	0,0	
8715	054330	057122		.WORD	DH164B	
8716	054332	005	000	.BYTE	5,0	
8717	054334	000005		DF170: .WORD	5	:ERROR 170
8718	054336	000	000	.BYTE	0,0	
8719	054340	055030		.WORD	DH000A	
8720	054342	000	000	.BYTE	0,0	
8721	054344	055046		.WORD	DH000B	
8722	054346	002	000	.BYTE	2,0	
8723	054350	057170		.WORD	DH170A	
8724	054352	000	000	.BYTE	0,0	
8725	054354	057267		.WORD	DH170B	
8726	054356	010	000	.BYTE	8,0	

8727	054360	000005		DF171:	.WORD	5		:ERRORS 171-174
8728	054362	000	000		.BYTE	0,0		
8729	054364	055030			.WORD	04000A		
8730	054366	000	000		.BYTE	0,0		
8731	054370	055046			.WORD	04000B		
8732	054372	002	000		.BYTE	2,0		
8733	054374	057365			.WORD	04171A		
8734	054376	000	000		.BYTE	0,0		
8735	054400	057404			.WORD	04171B		
8736	054402	002	000		.BYTE	2,0		
8737	054404	000005		DF175:	.WORD	5		:ERRORS 175-210
8738	054406	000	000		.BYTE	0,0		
8739	054410	055030			.WORD	04000A		
8740	054412	000	000		.BYTE	0,0		
8741	054414	055046			.WORD	04000B		
8742	054416	002	000		.BYTE	2,0		
8743	054420	057422			.WORD	04175A		
8744	054422	000	000		.BYTE	0,0		
8745	054424	057521			.WORD	04175B		
8746	054426	010	000		.BYTE	8,0		
8747	054430	000007		DF211:	.WORD	7		:ERRORS 211-214
8748	054432	000	000		.BYTE	0,0		
8749	054434	055030			.WORD	04000A		
8750	054436	000	000		.BYTE	0,0		
8751	054440	055046			.WORD	04000B		
8752	054442	002	JC		.BYTE	2,0		
8753	054444	057422			.WORD	04175A		
8754	054446	000	000		.BYTE	0,0		
8755	054450	057521			.WORD	04175B		
8756	054452	010	000		.BYTE	8,0		
8757	054454	057616			.WORD	04211A		
8758	054456	000	000		.BYTE	0,0		
8759	054460	057644			.WORD	04211B		
8760	054462	004	000		.BYTE	4,0		

```

8761 .SBTTL ASCII MESSAGES
8762
8763 054464 005015 045522 030466 OPRO01: .ASCIZ <15><12>/RK611 VECTOR ADDRESS ( /
8764 054472 020061 042526 052103
8765 054500 051117 040440 042104
8766 054506 042522 051523 024040
8767 054514 000040
8768 054516 024440 036440 000040 OPRO02: .ASCIZ / ) = /
8769 054524 045522 030466 020061 OPRO03: .ASCIZ /RK611 VECTOR ADDRESS (
8770 054532 042526 052103 051117
8771 054540 040440 042104 042522
8772 054546 051523 024040 000040
8773 054554 045522 030466 020061 OPRO04: .ASCIZ /RK611 PRIORITY ( /
8774 054562 051120 047511 044522
8775 054570 054524 024040 000040
8776 054576 005015 047062 020104 OPRO06: .ASCIZ <15><12>/2ND PASS RUN TIME IS APPROX 8 MINUTES/<15><12>
8777 054604 040520 051523 051040
8778 054612 047125 052040 046511
8779 054620 020105 051511 040440
8780 054626 050120 047522 020130
8781 054634 020070 044515 052516
8782 054642 042524 006523 000012
8783 054650 005015 025052 025052 OPRO07: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
8784 054656 020052 020040 051120
8785 054664 043517 040522 020115
8786 054672 040510 052114 042105
8787 054700 020040 025040 025052
8788 054706 025052 006452 000012
8789 054714 020040 000
8790 054717 015 050012 047522
8791 054724 051107 046501 040440
8792 054732 047502 052122 042105
8793 054740 041040 041505 052501
8794 054746 042523 042440 051122
8795 054754 051117 052040 051110
8796 054762 051505 047510 042114
8797 054770 042440 041530 042505
8798 054776 042504 006504 000012
8799 055004 005015 042524 052123 TSTBY1: .ASCIZ <15><12>/TEST /
8800 055012 000040
8801 055014 041040 050131 051501 TSTBY2: .ASCIZ / BYPASSED/<15><12>
8802 055022 042523 006504 000012

```

DATA HEADERS

				.SBTTL DATA HEADERS			
8803							
8804							
8805	055030	042524	052123	020040	DH000A: .ASCIZ	/TEST	ERROR/
8806	055036	020040	051105	047522			
8807	055044	000122					
8808	055046	052516	020115	020040	DH000B: .ASCIZ	/NUM	PC/
8809	055054	020040	041520	000			
8810	055061	124	051505	020124	DH000C: .ASCII	/TEST	TRAP/(<15><12>)
8811	055066	020040	052040	040522			
8812	055074	006520	012				
8813	055077	116	046525	020040		.ASCIZ	/NUM PC/
8814	055104	020040	050040	000103			
8815	055112	054105	042520	052103	DH001A: .ASCIZ	/EXPECT	ACUTAL EXPECT ACTUAL EXPECT ACTUAL
8816	055120	020040	041501	052125			
8817	055126	046101	020040	054105			
8818	055134	042520	052103	020040			
8819	055142	041501	052524	046101			
8820	055150	020040	054105	042520			
8821	055156	052103	020040	041501			
8822	055164	052524	046101	000			
8823	055171	122	041513	030523	DH001B: .ASCIZ	/RKCS1	RKCS1 MESS A MESS A MESS B MESS B/
8824	055176	020040	051040	041513			
8825	055204	030523	020040	046440			
8826	055212	051505	020123	020101			
8827	055220	046440	051505	020123			
8828	055226	020101	046440	051505			
8829	055234	020123	020102	046440			
8830	055242	051505	020123	000102			
8831	055250	054105	042520	052103	DH015A: .ASCIZ	/EXPECT	ACTUAL/
8832	055256	020040	041501	052524			
8833	055264	046101	000				
8834	055267	122	041513	030523	DH015B: .ASCIZ	/RKCS1	RKCS1/
8835	055274	020040	051040	041513			
8836	055302	030523	000				
8837	055305	105	050130	041505	DH017A: .ASCIZ	/EXPECT	ACTUAL PRESENT PREVIOUS BIT/
8838	055312	020124	040440	052103			
8839	055320	040525	020114	050040			
8840	055326	042522	042523	052116			
8841	055334	050040	042522	047526			
8842	055342	051525	041040	052111			
8843	055350	000					
8844	055351	122	046513	030522	DH017B: .ASCIZ	/RKMR1	RKMR1 BIT BIT COUNT/
8845	055356	020040	051040	046513			
8846	055364	030522	020040	041040			
8847	055372	052111	020040	020040			
8848	055400	041040	052111	020040			
8849	055406	020040	041440	052517			
8850	055414	052116	000				
8851	055417	105	050130	041505	DH022A: .ASCIZ	/EXPECT	ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
8852	055424	020124	040440	052103			
8853	055432	040525	020114	042440			
8854	055440	050130	041505	020124			
8855	055446	040440	052103	040525			
8856	055454	020114	042440	050130			
8857	055462	041505	020124	040440			
8858	055470	052103	040525	000114			

8915	056172	041113	101									
8916	056175	040	020040	051040	.ASCIZ	/	RKWC	RKWC/				
8917	056202	053513	020103	020040								
8918	056210	051040	053513	000103								
8919	056216	054105	042520	052103	DH041A:	.ASCIZ	/EXPECT	WORD/				
8920	056224	020040	047527	042122								
8921	056232	000										
8922	056233	127	051117	020104	DH041B:	.ASCIZ	/WORD	READ/				
8923	056240	020040	051040	040505								
8924	056246	000104										
8925	056250	054105	042520	052103	DH042A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/		
8926	056256	020040	041501	052524								
8927	056264	046101	020040	054105								
8928	056272	042520	052103	020040								
8929	056300	041501	052524	046101								
8930	056306	000										
8931	056307	122	041513	030523	DH042B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2/		
8932	056314	020040	051040	041513								
8933	056322	030523	020040	051040								
8934	056330	041513	031123	020040								
8935	056336	051040	041513	031123								
8936	056344	000										
8937	056345	105	050130	041505	DH054A:	.ASCIZ	/EXPECT	WORD	WORD/			
8938	056352	020124	053440	051117								
8939	056360	020104	020040	053440								
8940	056366	051117	000104									
8941	056372	047527	042122	020040	DH054B:	.ASCIZ	/WORD	READ	COUNT/			
8942	056400	020040	042522	042101								
8943	056406	020040	020040	047503								
8944	056414	047125	000124									
8945	056420	054105	042520	052103	DH055A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	WORD/	
8946	056426	020040	041501	052524								
8947	056434	046101	020040	054105								
8948	056442	042520	052103	020040								
8949	056450	041501	052524	046101								
8950	056456	020040	047527	042122								
8951	056464	000										
8952	056465	122	041513	030523	DH055B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2	COUNT/	
8953	056472	020040	051040	041513								
8954	056500	030523	020040	051040								
8955	056506	041513	031123	020040								
8956	056514	051040	041513	031123								
8957	056522	020040	041440	052517								
8958	056530	052116	000									
8959	056533	105	050130	041505	DH072A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/		
8960	056540	020124	040440	052103								
8961	056546	040525	020114	042440								
8962	056554	050130	041505	020124								
8963	056562	040440	052103	040525								
8964	056570	000114										
8965	056572	045522	051503	020061	DH072B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2/		
8966	056600	020040	045522	051503								
8967	056606	020061	020040	045522								
8968	056614	051503	020062	020040								
8969	056622	045522	051503	000062								
8970	056630	054105	042520	052103	DH072C:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT	ACTUAL

.SBTTL ERROR MESSAGES

9068						
9069						
9070	057701	125	042516	050130	EM000:	.ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
9071	057706	041505	042524	020104		
9072	057714	042515	047515	054522		
9073	057722	050040	051101	052111		
9074	057730	020131	047105	041101		
9075	057736	042514	052040	040522		
9076	057744	000120				
9077	057746	052101	042524	050115	EM200:	.ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER/
9078	057754	044524	043516	052040		
9079	057762	020117	044103	041505		
9080	057770	020113	042523	045505		
9081	057776	046440	051505	040523		
9082	060004	042507	043040	047522		
9083	060012	020115	042522	042101		
9084	060020	044040	040505	042504		
9085	060026	000122				
9086	060030	052101	042524	050115	EM201:	ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER/
9087	060036	044524	043516	052040		
9088	060044	020117	044103	041505		
9089	060052	020113	042523	045505		
9090	060060	046440	051505	040523		
9091	060066	042507	043040	047522		
9092	060074	020115	051127	052111		
9093	060102	020105	042510	042101		
9094	060110	051105	000			
9095	060113	101	052124	046505	EM202:	.ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM READ HEADER/
9096	060120	052120	047111	020107		
9097	060126	047524	041440	042510		
9098	060134	045503	041440	042514		
9099	060142	051101	042040	044522		
9100	060150	042526	046440	051505		
9101	060156	040523	042507	043040		
9102	060164	047522	020115	042522		
9103	060172	042101	044040	040505		
9104	060200	042504	000122			
9105	060204	052101	042524	050115	EM203:	.ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM WRITE HEADER/
9106	060212	044524	043516	052040		
9107	060220	020117	044103	041505		
9108	060226	020113	046103	040505		
9109	060234	020122	051104	053111		
9110	060242	020105	042515	051523		
9111	060250	043501	020105	051106		
9112	060256	046517	053440	044522		
9113	060264	042524	044040	040505		
9114	060272	042504	000122			
9115	060276	052101	042524	050115	EM204:	.ASCIZ /ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT/
9116	060304	044524	043516	040440		
9117	060312	051040	040505	020104		
9118	060320	042510	042101	051105		
9119	060326	052040	020117	044103		
9120	060334	041505	020113	042523		
9121	060342	052103	051117	050040		
9122	060350	046125	042523	042040		
9123	060356	052105	041505	000124		

E14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04

MACY11 27(1006) 28-JAN-77 14:16 PAGE 172

SEQ 0172

ERROR MESSAGES

9124	060364	052101	042524	050115	EM205: .ASCIZ /ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT/
9125	060372	044524	043516	040440	
9126	060400	053440	044522	042524	
9127	060406	044040	040505	042504	
9128	060414	020122	047524	041440	
9129	060422	042510	045503	044440	
9130	060430	042116	054105	050040	
9131	060436	046125	042523	042040	
9132	060444	052105	041505	000124	
9133	060452	052101	042524	050115	EM206: .ASCIZ /ATTEMPTING AN NPR READ OF ONE WORD/
9134	060460	044524	043516	040440	
9135	060466	020116	050116	020122	
9136	060474	042522	042101	047440	
9137	060502	020106	047117	020105	
9138	060510	047527	042122	000	
9139	060515	101	052124	046505	EM207: .ASCIZ /ATTEMPTING AN NPR READ/
9140	060522	052120	047111	020107	
9141	060530	047101	047040	051120	
9142	060536	051040	040505	000104	
9143	060544	052101	042524	050115	EM208: .ASCIZ /ATTEMPTING NPR READ CHECKING ZERO DETECT/
9144	060552	044524	043516	047040	
9145	060560	051120	051040	040505	
9146	060566	020104	044103	041505	
9147	060574	044513	043516	055040	
9148	060602	051105	020117	042504	
9149	060610	042524	052103	000	
9150	060615	101	052124	046505	EM209: .ASCIZ /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
9151	060622	052120	047111	020107	
9152	060630	050116	020122	042522	
9153	060636	042101	053440	052111	
9154	060644	020110	052502	020123	
9155	060652	042101	051104	051505	
9156	060660	020123	047111	051103	
9157	060666	046505	047105	020124	
9158	060674	047111	044510	044502	
9159	060702	000124			
9160	060704	052101	042524	050115	EM210: .ASCII /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
9161	060712	044524	043516	047040	
9162	060720	051120	051040	040505	
9163	060726	020104	044527	044124	
9164	060734	041040	051525	040440	
9165	060742	042104	042522	051523	
9166	060750	044440	041516	042522	
9167	060756	042515	052116	044440	
9168	060764	044116	041111	052111	
9169	060772	041440	042510	045503	.ASCIZ / CHECKING ZERO DETECT/
9170	061000	047111	020107	042532	
9171	061006	047522	042040	052105	
9172	061014	041505	000124		
9173	061020	052101	042524	050115	EM211: .ASCIZ /ATTEMPTING TO FORCE NON-EXISTENT MEMORY/
9174	061026	044524	043516	052040	
9175	061034	020117	047506	041522	
9176	061042	020105	047516	026516	
9177	061050	054105	051511	042524	
9178	061056	052116	046440	046505	
9179	061064	051117	000131		

9180	06107C	051101	042524	050115	EM212: .ASCIZ /ATTEMPTING TO CLEAR NON-EXISTENT MEMORY/
9181	061076	044524	043516	052040	
9182	061104	020117	046103	040505	
9183	061112	020122	047516	026516	
9184	061120	054105	051511	042524	
9185	061126	052116	046440	046505	
9186	061134	051117	000131		
9187	061140	042524	052123	047111	EM213: .ASCIZ /TESTING EXTENDED MEMORY ADDRESSING BITS/
9188	061146	020107	054105	042524	
9189	061154	042116	042105	046440	
9190	061162	046505	051117	020131	
9191	061170	042101	051104	051505	
9192	061176	044523	043516	041040	
9193	061204	052111	000123		
9194	061210	052101	042524	050115	EM214: .ASCIZ /ATTEMPTING TO FORCE UNIBUS PARITY ERROR.
9195	061216	044524	043516	052040	
9196	061224	020117	047506	041522	
9197	061232	020105	047125	041111	
9198	061240	051525	050040	051101	
9199	061246	052111	020131	051105	
9200	061254	047522	000122		
9201	061260	052101	042524	050115	EM215: .ASCIZ /ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY/
9202	061266	044524	043516	047040	
9203	061274	051120	051040	040505	
9204	061302	020104	043117	046040	
9205	061310	041517	052101	047511	
9206	061316	020116	051120	047511	
9207	061324	020122	047524	041040	
9208	061332	042101	050040	051101	
9209	061340	052111	000131		
9210	061344	052101	042524	050115	EM216: .ASCIZ /ATTEMPTING TO CLEAR UNIBUS PARITY ERROR/
9211	061352	044524	043516	052040	
9212	061360	020117	046103	040505	
9213	061366	020122	047125	041111	
9214	061374	051525	050040	051101	
9215	061402	052111	020131	051105	
9216	061410	047522	000122		
9217	061414	052101	042524	050115	EM217: .ASCIZ /ATTEMPTING 18 BIT NPR READ/
9218	061422	044524	043516	030440	
9219	061430	020070	044502	020124	
9220	061436	050116	020122	042522	
9221	061444	042101	000		
9222	061447	101	052124	046505	EM218: .ASCIZ /ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT/
9223	061454	052120	047111	020107	
9224	061462	034061	041040	052111	
9225	061470	047040	051120	051040	
9226	061476	040505	020104	044103	
9227	061504	041505	044513	043516	
9228	061512	055040	051105	020117	
9229	061520	042504	042524	052103	
9230	061526	000			
9231	061527	101	052124	046505	EM219: .ASCIZ /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
9232	061534	052120	047111	020107	
9233	061542	034061	041040	052111	
9234	061550	047040	051120	051040	
9235	061556	040505	020104	044527	

ERROR MESSAGES

9236	061564	044124	041040	052111	
9237	061572	030440	020066	050050	
9238	061600	024501	051440	052105	
9239	061606	000			
9240	061607	101	052124	046505	EM220: .ASCII /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
9241	061614	052120	047111	020107	
9242	061622	034061	041040	052111	
9243	061630	047040	051120	051040	
9244	061636	040505	020104	044527	
9245	061644	044124	041040	052111	
9246	061652	030440	020066	050050	
9247	061660	024501	051440	052105	
9248	061666	052101	042524	050115	EM221: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER/
9249	061674	044524	043516	051440	
9250	061702	046511	046125	052101	
9251	061710	047511	020116	043117	
9252	061716	042040	052101	020101	
9253	061724	047111	051040	040505	
9254	061732	020104	042510	042101	
9255	061740	051105	000		
9256	061743	101	052124	046505	EM222: .ASCIZ /ATTEMPTING READ HEADER IN MAINTANENCE MODE/
9257	061750	052120	047111	020107	
9258	061756	042522	042101	044040	
9259	061764	040505	042504	020122	
9260	061772	047111	046440	044501	
9261	062000	052116	047101	047105	
9262	062006	042503	046440	042117	
9263	062014	000105			
9264	062016	052101	042524	050115	EM223: .ASCIZ /ATTEMPTING DATA BUFFER READ AFTER READ HEADER/
9265	062024	044524	043516	042040	
9266	062032	052101	020101	052502	
9267	062040	043106	051105	051040	
9268	062046	040505	020104	043101	
9269	062054	042524	020122	042522	
9270	062062	04210	044040	040505	
9271	062070	042504	000122		
9272	062074	052101	042524	050115	EM224: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER (18 BIT FORMAT)/
9273	062102	044524	043516	051440	
9274	062110	046511	046125	052101	
9275	062116	047511	020116	043117	
9276	062124	042040	052101	020101	
9277	062132	047111	051040	040505	
9278	062140	020104	042510	042101	
9279	062146	051105	024040	034061	
9280	062154	041040	052111	043040	
9281	062162	051117	040515	024524	
9282	062170	000			
9283	062171	101	052124	046505	EM225: .ASCIZ /ATTEMPTING READ HEADER (18 BIT FORMAT) IN MAINT MODE/
9284	062176	052120	047111	020107	
9285	062204	042522	042101	044040	
9286	062212	040505	042504	020122	
9287	062220	030450	020070	044502	
9288	062226	020124	047506	046522	
9289	062234	052101	020051	047111	
9290	062242	046440	044501	052116	
9291	062250	046440	042117	000105	

9292	062256	052101	042524	050115	EM226: .ASCII /ATTEMPTING DATA BUFFER READ AFTER/<12><15>
9293	062264	044524	043516	042040	
9294	062272	052101	020101	052502	
9295	062300	043106	051105	051040	
9296	062306	040505	020104	043101	
9297	062314	042524	005122	015	
9298	062321	122	040505	020104	.ASCIZ /READ HEADER IN 18 BIT FORMAT/
9299	062326	042510	042101	051105	
9300	062334	044440	020116	034061	
9301	062342	041040	052111	043040	
9302	062350	051117	040515	000124	
9303	062356	052101	042524	050115	EM227: .ASCII /ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER/<15><12>
9304	062364	044524	043516	052040	
9305	062372	020117	044103	041505	
9306	062400	020113	054523	041516	
9307	062406	020110	042504	042524	
9308	062414	052103	047440	020116	
9309	062422	042522	042101	044040	
9310	062430	040505	042504	006522	
9311	062436	012			
9312	062437	103	042510	045503	.ASCIZ /CHECKING ZERO DETECT/
9313	062444	047111	020107	042532	
9314	062452	047522	042040	052105	
9315	062460	041505	000124		
9316	062464	052101	042524	050115	EM230: .ASCII /ATTEMPTING TO CHECK WRITING OF ZEROES BETWEEN INDEX/<15><12>
9317	062472	044524	043516	052040	
9318	062500	020117	044103	041505	
9319	062506	020113	051127	052111	
9320	062514	047111	020107	043117	
9321	062522	055040	051105	042517	
9322	062530	020123	042502	053524	
9323	062536	042505	020116	047111	
9324	062544	042504	006530	012	
9325	062551	101	042116	051440	.ASCIZ /AND SECTOR PULSE/
9326	062556	041505	047524	020122	
9327	062564	052520	051514	000105	
9328	062572	052101	042524	050115	EM231: .ASCII /ATTEMPTING TO RESET WRITE GATE BY SETTING/<15><12>
9329	062600	044524	043516	052040	
9330	062606	020117	042522	042523	
9331	062614	020124	051127	052111	
9332	062622	020105	040507	042524	
9333	062630	041040	020131	042523	
9334	062636	052124	047111	006507	
9335	062644	012			
9336	062645	123	041505	047524	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
9337	062652	020122	052520	051514	
9338	062660	020105	047111	040440	
9339	062666	053440	044522	042524	
9340	062674	044040	040505	042504	
9341	062702	020122	047503	046515	
9342	062710	047101	000104		
9343	062714	052101	042524	050115	EM232: .ASCII /ATTEMPTING TO SET WRITE GATE BY RESETTING/<15><12>
9344	062722	044524	043516	052040	
9345	062730	020117	042523	020124	
9346	062736	051127	052111	020105	
9347	062744	040507	042524	041040	

9348	062752	020131	042522	042523	
9349	062760	052124	047111	006507	
9350	062766	012			
9351	062767	123	041505	047524	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
9352	062774	020122	052520	051514	
9353	063002	020105	047111	040440	
9354	063010	053440	044522	042524	
9355	063016	044040	040505	042504	
9356	063024	020122	047503	046515	
9357	063032	047101	000104		
9358	063036	052101	042524	050115	EM233: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER/
9359	063044	044524	043516	052040	
9360	063052	020117	051127	052111	
9361	063060	020105	054523	041516	
9362	063066	020110	043117	044040	
9363	063074	040505	042504	000122	
9364	063102	052101	042524	050115	EM234: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA/
9365	063110	044524	043516	052040	
9366	063116	020117	051127	052111	
9367	063124	020105	042510	042101	
9368	063132	051105	042040	052101	
9369	063140	000101			
9370	063142	052101	042524	050115	EM235: .ASCII /ATTEMPTING TO RESET WRITE GATE WITH SECOND/<15><12>
9371	063150	044524	043516	052040	
9372	063156	020117	042522	042523	
9373	063164	020124	051127	052111	
9374	063172	020105	040507	042524	
9375	063200	053440	052111	020110	
9376	063206	042523	047503	042116	
9377	063214	005015			
9378	063216	047111	042504	020130	.ASCIZ /INDEX PULSE OF WRITE HEADER/
9379	063224	052520	051514	020105	
9380	063232	043117	053440	044522	
9381	063240	042524	044040	040505	
9382	063246	042504	000122		
9383	063252	052101	042524	050115	EM236: .ASCIZ /ATTEMPTING TO COMPLETE WRITE HEADER IN MAINT MODE/
9384	063260	044524	043516	052040	
9385	063266	020117	047503	050115	
9386	063274	042514	042524	053440	
9387	063302	044522	042524	044040	
9388	063310	040505	042504	020122	
9389	063316	047111	046440	044501	
9390	063324	052116	046440	042117	
9391	063332	020105			
9392	063334	052101	042524	050115	EM237: .ASCIZ /ATTEMPTING TO WRITE GAP OR DATA SYNCH/
9393	063342	044524	043516	052040	
9394	063350	020117	051127	052111	
9395	063356	020105	040507	020120	
9396	063364	051117	042040	052101	
9397	063372	020101	054523	041516	
9398	063400	000110			
9399	063402	052101	042524	050115	EM238: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD/
9400	063410	044524	043516	052040	
9401	063416	020117	051127	052111	
9402	063424	020105	040504	040524	
9403	063432	043040	042511	042114	

ERROR MESSAGES

9404	063440	000			
9405	063441	101	052124	046505	EM239: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER USING 24 SECTOR FORMAT/
9406	063446	044520	043516	052040	
9407	063454	020117	051127	052111	
9408	063462	020105	054523	041516	
9409	063470	020110	043117	044040	
9410	063476	040505	042504	020122	
9411	063504	051525	047111	020107	
9412	063512	032062	051440	041505	
9413	063520	047524	020122	047506	
9414	063526	046522	052101	000	
9415	063533	101	052124	046505	EM240: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA USING 24 SECTOR FORMAT/
9416	063540	052120	047111	020107	
9417	063546	047524	053440	044522	
9418	063554	042524	044040	040505	
9419	063562	042504	020122	040504	
9420	063570	040524	052440	044523	
9421	063576	043516	031040	020064	
9422	063604	042523	052103	051117	
9423	063612	043040	051117	040515	
9424	063620	000124			
9425	063622	052101	042524	050115	EM241: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26 SECTOR)/
9426	063630	044524	043516	052040	
9427	063636	020117	047506	041522	
9428	063644	020105	047506	046522	
9429	063652	052101	042440	051122	
9430	063660	051117	024040	043103	
9431	063666	052115	036440	031040	
9432	063674	020066	042523	052103	
9433	063702	051117	000051		
9434	063706	052101	042524	050115	EM242: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24 SECTOR)/
9435	063714	044524	043516	052040	
9436	063722	020117	047506	041522	
9437	063730	020105	047506	046522	
9438	063736	052101	042440	051122	
9439	063744	051117	024040	043103	
9440	063752	052115	036440	031040	
9441	063760	020064	042523	052103	
9442	063766	051117	000051		
9443	063772	052101	042524	050115	EM243: .ASCIZ /ATTEMPTING TO FORCE CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS/
9444	064000	047111	020107	047524	
9445	064006	043040	051117	042503	
9446	064014	041440	047117	051124	
9447	064022	046117	042514	020122	
9448	064030	051105	047522	020122	
9449	064036	044527	044124	043040	
9450	064044	052501	052114	041040	
9451	064052	052111	044440	020116	
9452	064060	051104	053111	020105	
9453	064066	042515	051523	000	
9454	064073	101	052124	046505	EM244: .ASCIZ /ATTEMPTING TO CLEAR ERROR/
9455	064100	052120	047111	020107	
9456	064106	047524	041440	042514	
9457	064114	051101	042440	051122	
9458	064122	051117	000		
9459	064125	103	046517	040515	EM3000: .ASCIZ /COMMAND AND STATUS REG 1 INCORRECT

960	064132	042116	040440	042116	
961	064140	051440	040524	052524	
962	064146	020123	042522	020107	
963	064154	020061	047111	047503	
964	064162	051122	041505	000124	
965	064170	042515	051523	043501	EM3001: .ASCIZ /MESSAGE A INCORRECT/
966	064176	020105	020101	047111	
967	064204	047503	051122	041505	
968	064212	000124			
969	064214	042515	051523	043501	EM3002: .ASCIZ /MESSAGE B INCORRECT/
970	064222	020105	020102	047111	
971	064230	047503	051122	041505	
972	064236	000124			
973	064240	051503	020061	047111	EM3003: .ASCIZ /CSI INCORRECT AFTER SENDING DRIVE CLEAR
974	064246	047503	051122	041505	
975	064254	020124	043101	042524	
976	064262	020122	042523	042116	
977	064270	047111	020107	051104	
978	064276	053111	020105	046103	
979	064304	040505	000122		
980	064310	051503	020061	047111	EM3004: .ASCIZ /CSI INCORRECT AFTER AFTER DATA SIMULATION/
981	064316	047503	051122	041505	
982	064324	020124	043101	042524	
983	064332	020122	043101	042524	
984	064340	020122	040504	040524	
985	064346	051440	046511	046125	
986	064354	052101	047511	000116	
987	064362	040515	047111	020124	EM3005: ASCII /MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
988	064370	042522	027107	030440	
989	064376	044440	041516	051117	
990	064404	042522	052103	042040	
991	064412	051125	047111	020107	
992	064420	040504	040524	051440	
993	064426	046511	046125	052101	
994	064434	047511	116		
995	064437	015	040412	052106	.ASCIZ <15><12>/AFTER SECTOR PULSE/
996	064444	051105	051440	041505	
997	064452	047524	020122	052520	
998	064460	051514	000105		
999	064464	040515	047111	020124	EM3006: .ASCII MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9500	064472	042522	027107	030440	
9501	064500	044440	041516	051117	
9502	064506	042522	052103	042040	
9503	064514	051125	047111	020107	
9504	064522	040504	040524	051440	
9505	064530	046511	046125	052101	
9506	064536	047511	116		
9507	064544	015	040412	052106	.ASCIZ <15><12>/AFTER INDEX PULSE/
9508	064552	051105	044440	042116	
9509	064554	054105	050040	046125	
9510	064562	042523	000		
9511	064565	115	044501	052116	EM3007: .ASCII /MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9512	064572	051040	043505	020056	
9513	064600	020061	047111	047503	
9514	064606	051122	041505	020124	
9515	064614	052504	044522	043516	

L14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04

MACY11 27(1006) 28-JAN-77 14:16 PAGE 179

SEQ 0179

ERROR MESSAGES

9516	064622	042040	052101	020101	
9517	064630	044523	052515	040514	
9518	064636	044524	047117		
9519	064642	005015	047516	051440	.ASCIZ <15><12>/NO SECTOR OR INDEX PULSE SUPPLIED/
9520	064650	041505	047524	020122	
9521	064656	051117	044440	042116	
9522	064664	054105	050040	046125	
9523	064672	042523	051440	050125	
9524	064700	046120	042511	000104	
9525	064706	052502	020123	042101	EM3008: .ASCIZ /BUS ADDRESS INCORRECT AFTER SENDING DRIVE CLEAR/
9526	064714	051104	051505	020123	
9527	064722	047111	047503	051122	
9528	064730	041505	020124	043101	
9529	064736	042524	020122	042523	
9530	064744	042116	047111	020107	
9531	064752	051104	053111	020105	
9532	064760	046103	040505	000122	
9533	064766	047527	042122	041440	EM3009: .ASCIZ /WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR/
9534	064774	052517	052116	044440	
9535	065002	041516	051117	042522	
9536	065010	052103	040440	052106	
9537	065016	051105	051440	047105	
9538	065024	044504	043516	042040	
9539	065032	044522	042526	041440	
9540	065040	042514	051101	000	
9541	065045	103	030523	041440	EM3010: .ASCIZ /CSI CHANGED DURING COMMAND EXECUTION/
9542	065052	040510	043516	042105	
9543	065060	042040	051125	047111	
9544	065066	020107	047503	046515	
9545	065074	047101	020104	054105	
9546	065102	041505	052125	047511	
9547	065110	000116			
9548	065112	052502	020123	042101	EM3011: .ASCIZ /BUS ADDRESS CHANGED BEFORE INDEX PULSE/
9549	065120	051104	051505	020123	
9550	065126	044103	047101	042507	
9551	065134	020104	042502	047506	
9552	065142	042522	044440	042116	
9553	065150	054105	050040	046125	
9554	065156	042523	000		
9555	065161	127	051117	020104	EM3012: .ASCIZ /WORD COUNT CHANGED BEFORE INDEX PULSE/
9556	065166	047503	047125	020124	
9557	065174	044103	047101	042507	
9558	065202	020104	042502	047506	
9559	065210	042522	044440	042116	
9560	065216	054105	050040	046125	
9561	065224	042523	000		
9562	065227	103	030523	041440	EM3013: .ASCIZ /CSI CHANGE AFTER INDEX PULSE/
9563	065234	040510	043516	020105	
9564	065242	043101	042524	020122	
9565	065250	047111	042504	020130	
9566	065256	052520	051514	000105	
9567	065264	040515	047111	020124	EM3014: .ASCIZ /MAINT REG 1 INCORRECT BEFORE INDEX PULSE/
9568	065272	042522	020107	020061	
9569	065300	047111	047503	051122	
9570	065306	041505	020124	042502	
9571	065314	047506	042522	044440	

M14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04 ERROR MESSAGES

MACY11 27(1006) 28-JAN-77 14:16 PAGE 180

SEQ 0180

9572	065322	042116	054105	050040	
9573	065330	046125	042523	000	
9574	065335	102	051525	040440	EM3015: .ASCIZ /BUS ADDRESS CHANGED AFTER INDEX PULSE/
9575	065342	042104	042522	051523	
9576	065350	041440	040510	043516	
9577	065356	042105	040440	052106	
9578	065364	051105	044440	042116	
9579	065372	054105	050040	046125	
9580	065400	042523	000		
9581	065403	127	051117	020104	EM3016: .ASCIZ /WORD COUNT CHANGED AFTER INDEX PULSE/
9582	065410	047503	047125	020124	
9583	065416	044103	047101	042507	
9584	065424	020104	043101	042524	
9585	065432	020122	047111	042504	
9586	065440	020130	052520	051514	
9587	065446	000105			
9588	065450	047503	046515	047101	EM3018: .ASCIZ /COMMAND AND STATUS REG. 2 INCORRECT/
9589	065456	020104	047101	020104	
9590	065464	052123	052101	051525	
9591	065472	051040	043505	020056	
9592	065500	020062	047111	047503	
9593	065506	051122	041505	000124	
9594	065514	052502	020123	042101	EM3019: .ASCIZ /BUS ADD REG INCORRECT/
9595	065522	020104	042522	020107	
9596	065530	047111	047503	051122	
9597	065536	041505	000124		
9598	065542	047527	042122	041440	EM3020: .ASCIZ /WORD COUNT REG INCORRECT/
9599	065550	052517	052116	051040	
9600	065556	043505	044440	041516	
9601	065564	051117	042522	052103	
9602	065572	000			
9603	065573	104	052101	020101	EM3021: .ASCIZ /DATA READ INCORRECT/
9604	065600	042522	042101	044440	
9605	065606	041516	051117	042522	
9606	065614	052103	000		
9607	065617	103	030523	044440	EM3022: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
9608	065624	041516	051117	042522	
9609	065632	052103	040440	052106	
9610	065640	051105	051040	040505	
9611	065646	044504	043516	042040	
9612	065654	052101	020101	052502	
9613	065662	043106	051105	000	
9614	065667	103	031123	044440	EM3023: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
9615	065674	041516	051117	042522	
9616	065702	052103	040440	052106	
9617	065710	051105	051040	040505	
9618	065716	044504	043516	042040	
9619	065724	052101	020101	052502	
9620	065732	043106	051105	000	
9621	065737	105	051122	051117	EM3024: .ASCIZ /ERROR REG INCORRECT/
9622	065744	051040	043505	044440	
9623	065752	041516	051117	042522	
9624	065760	052103	000		
9625	065763	115	044501	052116	EM3025: .ASCIZ /MAINT REG 1 INCORRECT AFTER INDEX PULSE/
9626	065770	051040	043505	030440	
9627	065776	044440	041516	051117	

N14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA
DZR6CB.P11 28-JAN-77 14:04 ERROR MESSAGES

MACY11 27(1006) 28-JAN-77 14:16 PAGE 181

SEQ 0181

9628	066004	042522	052103	040440	
9629	066012	052106	051105	044440	
9630	066020	042116	054105	050040	
9631	066026	046125	042523	000	
9632	066033	103	030523	044440	EM3026: .ASCIZ /CS1 INCORRECT AFTER COMMAND COMPLETION/
9633	066040	041516	051117	042522	
9634	066046	052103	040440	052106	
9635	066054	051105	041440	046517	
9636	066062	040515	042116	041440	
9637	066070	046517	046120	052105	
9638	066076	047511	000116		
9639	066102	051503	020062	047111	EM3027: .ASCIZ /CS2 INCORRECT AFTER COMMAND COMPLETION/
9640	066110	047503	051122	041505	
9641	066116	020124	043101	042524	
9642	066124	020122	047503	046515	
9643	066132	047101	020104	047503	
9644	066140	050115	042514	044524	
9645	066146	047117	000		
9646	066151	103	030523	044440	EM3028: .ASCIZ /CS1 INCORRECT AFTER READING EMPTY SILO/
9647	066156	041516	051117	042522	
9648	066164	052103	040440	052106	
9649	066172	051105	051040	040505	
9650	066200	044504	043516	042440	
9651	066206	050115	054524	051440	
9652	066214	046111	000117		
9653	066220	051503	020062	047111	EM3029: .ASCIZ /CS2 INCORRECT AFTER READING EMPTY SILO/
9654	066226	047503	051122	041505	
9655	066234	020124	043101	042524	
9656	066242	020122	042522	042101	
9657	066250	047111	020107	046505	
9658	066256	052120	020131	044523	
9659	066264	047514	000		
9660	066267	115	044501	052116	EM3030: .ASCIZ /MAINT REG 1 INCORRECT/
9661	066274	051040	043505	030440	
9662	066302	044440	041516	051117	
9663	066310	042522	052103	000	
9664	066315	104	044522	042526	EM3031: .ASCIZ /DRIVE STATUS REG INCORRECT/
9665	066322	051440	040524	052524	
9666	066330	020123	042522	020107	
9667	066336	047111	047503	051122	
9668	066344	041505	000124		
9669	066350	051105	047522	020122	EM3032: .ASCIZ /ERROR REG INCORRECT/
9670	066356	042522	020107	047111	
9671	066364	047503	051122	041505	
9672	066372	000124			
9673	066374	051115	020061	047111	EMW1: .ASCIZ /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
9674	066402	047503	051122	041505	
9675	066410	020124	047117	030440	
9676	066416	052123	052440	053520	
9677	066424	051101	020104	051124	
9678	066432	047101	044523	052123	
9679	066440	047511	020116	043117	
9680	066446	046440	044501	052116	
9681	066454	041440	047514	045503	
9682	066462	000			
9683	066463	115	030522	044440	EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/

9684	066470	041516	051117	042522
9685	066476	052103	047440	020116
9686	066504	051461	020124	047504
9687	066512	047127	040527	042122
9688	066520	052040	040522	051516
9689	066526	051511	044524	047117
9690	066534	047440	020106	040515
9691	066542	047111	020124	046103
9692	066550	041517	000113	
9693	066554	051115	020061	047111
9694	066562	047503	051122	041505
9695	066570	020124	047117	031040
9696	066576	042116	052440	053520
9697	066604	051101	020104	051124
9698	066612	047101	044523	052123
9699	066620	047511	020116	043117
9700	066626	046440	044501	052116
9701	066634	041440	047514	045503
9702	066642	000		
9703	066643	115	030522	044440
9704	066650	041516	051117	042522
9705	066656	052103	047440	020116
9706	066664	047062	020104	047504
9707	066672	047127	040527	042122
9708	066700	052040	040522	051516
9709	066706	051511	044524	047117
9710	066714	047440	020106	040515
9711	066722	047111	020124	046103
9712	066730	041517	000113	

EMW3: .ASCIZ /MRI INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/

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

Address	Hex Data	Hex Data	Hex Data	Label	Hex Data	Hex Data
9713				.SBTTL	DATA PATTERNS	
9714						
9715				.EVEN		
9716	066734	000000		PATTERN:	.WORD	000000
9717	066736	177777			.WORD	177777
9718	066740	125252			.WORD	125252
9719	066742	052515			.WORD	052515
9720	066744	101706			.WORD	101706
9721	066746	060422			.WORD	060422
9722	066750	130715			.WORD	130715
9723	066752	177777		HEAD1:	.WORD	177777
9724	066754	000000			.WORD	000000
9725	066756	177777			.WORD	177777
9726	066760	000000		HEAD2:	.WORD	000000
9727	066762	177777			.WORD	177777
9728	066764	000000			.WORD	000000
9729	066766	125252		HEAD3:	.WORD	125252
9730	066770	052525			.WORD	052525
9731	066772	125252			.WORD	125252
9732	066774	052525			.WORD	052525
9733	066776	125252			.WORD	125252
9734	067000	052525			.WORD	052525
9735	067002	044444		HEAD4:	.WORD	044444
9736	067004	022222			.WORD	022222
9737	067006	111111			.WORD	111111
9738	067010	052012		HEAD5:	.WORD	052012
9739	067012	100520			.WORD	100520
9740	067014	052012			.WORD	052012
9741	067016	155555		HEAD6:	.WORD	155555
9742	067020	066666			.WORD	066666
9743	067022	155555			.WORD	155555
9744	067024	104210		HEAD7:	.WORD	104210
9745	067026	104210			.WORD	104210
9746	067030	104210			.WORD	104210
9747	067032	100	100	NPRBUF:	.BYTE	100,100
9748	067034	101	101		.BYTE	101,101
9749	067036	102	102		.BYTE	102,102
9750	067040	103	103		.BYTE	103,103
9751	067042	104	104		.BYTE	104,104
9752	067044	105	105		.BYTE	105,105
9753	067046	106	106		.BYTE	106,106
9754	067050	107	107		.BYTE	107,107
9755	067052	110	110		.BYTE	110,110
9756	067054	111	111		.BYTE	111,111
9757	067056	112	112		.BYTE	112,112
9758	067060	113	113		.BYTE	113,113
9759	067062	114	114		.BYTE	114,114
9760	067064	115	115		.BYTE	115,115
9761	067066	116	116		.BYTE	116,116
9762	067070	117	117		.BYTE	117,117
9763	067072	120	120		.BYTE	120,120
9764	067074	121	121		.BYTE	121,121
9765	067076	122	122		.BYTE	122,122
9766	067100	123	123		.BYTE	123,123
9767	067102	124	124		.BYTE	124,124
9768	067104	125	125		.BYTE	125,125

9769	067106	126	126	.BYTE	126,126
9770	067110	127	127	.BYTE	127,127
9771	067112	130	130	.BYTE	130,130
9772	067114	131	131	.BYTE	131,131
9773	067116	132	132	.BYTE	132,132
9774	067120	133	133	.BYTE	133,133
9775	067122	134	134	.BYTE	134,134
9776	067124	135	135	.BYTE	135,135
9777	067126	136	136	.BYTE	136,136
9778	067130	137	137	.BYTE	137,137
9779	067132	140	140	.BYTE	140,140
9780	067134	141	141	.BYTE	141,141
9781	067136	142	142	.BYTE	142,142
9782	067140	143	143	.BYTE	143,143
9783	067142	144	144	.BYTE	144,144
9784	067144	145	145	.BYTE	145,145
9785	067146	146	146	.BYTE	146,146
9786	067150	147	147	.BYTE	147,147
9787	067152	150	150	.BYTE	150,150
9788	067154	151	151	.BYTE	151,151
9789	067156	152	152	.BYTE	152,152
9790	067160	153	153	.BYTE	153,153
9791	067162	154	154	.BYTE	154,154
9792	067164	155	155	.BYTE	155,155
9793	067166	156	156	.BYTE	156,156
9794	067170	157	157	.BYTE	157,157
9795	067172	160	160	.BYTE	160,160
9796	067174	161	161	.BYTE	161,161
9797	067176	162	162	.BYTE	162,162
9798	067200	163	163	.BYTE	163,163
9799	067202	164	164	.BYTE	164,164
9800	067204	165	165	.BYTE	165,165
9801	067206	166	166	.BYTE	166,166
9802	067210	167	167	.BYTE	167,167
9803	067212	170	170	.BYTE	170,170
9804	067214	171	171	.BYTE	171,171
9805	067216	172	172	.BYTE	172,172
9806	067220	173	173	.BYTE	173,173
9807	067222	174	174	.BYTE	174,174
9808	067224	175	175	.BYTE	175,175
9809	067226	176	176	.BYTE	176,176
9810	067230	177	177	.BYTE	177,177
9811	067232	200	200	.BYTE	200,200
9812	067234	201	201	.BYTE	201,201
9813	067236	202	202	.BYTE	202,202
9814	067240	203	203	.BYTE	203,203
9815	067242	204	204	.BYTE	204,204
9816	067244	205	205	.BYTE	205,205
9817	067246	206	206	.BYTE	206,206
9818	067250	207	207	.BYTE	207,207
9819	067252	210	210	.BYTE	210,210
9820	067254	000102			
9821		067270			
9822		000001			

WRBUFF: .BLKW 66.
 BADPAR= WRBUFF+14
 .END

		6901	6903	6904	6907	6909	6914	6915	6941	6945	6946	7018	7022	7023
		7095	7101	7102	7272	7302	7316	7331	7357	7369	7381	7406	7409	7410
		7411	7412	7415	7416	7417	7418							
DRA	= 000001	1138#	6961	7038	7117									
DRDY	= 000200	1145#												
DRDT	= 000040	1143#												
DRPAR	= 000010	1122#	7039											
DRVMSK	= 000007	1102#												
DSC	= 040000	1149#												
DSMR	= 177570	911#	1256	2306										
DTE	= 010000	1131#												
DTYE	= 000040	1124#												
DT000	053202	1965	8500#											
DT001	053206	1350	1356	1362	1368	1374	1380	1386	1392	1398	1404	1410	1416	8501#
DT015	053226	1422	1428	2086	8504#									
DT017	053236	1434	1440	1446	8506#									
DT022	053254	1452	1458	1464	1476	1482	1488	8509#						
DT025	053274	1470	8512#											
DT031	053306	1494	1500	1506	1512	8514#								
DT035	053332	1518	1524	1530	1536	1560	1566	1572	1578	1584	1590	1596	1602	1626
		1632	1638	1644	1650	1656	1662	1668	1734	1740	1746	1752	1788	1794
		1800	1806	1830	1836	1842	1848	1854	1860	1866	1872	1896	1902	1908
		1914	1921	1928	1935	1942	8518#							
DT041	053356	1542	1758	1948	8522#									
DT042	053366	1548	1554	1878	1977	1983	2013	2019	8524#					
DT054	053402	1608	1674	2001	2037	8526#								
DT055	053414	1614	1620	1680	1686	1884	1890	1989	1995	2025	2031	8528#		
DT072	053432	1692	1698	1704	1710	1716	1764	1770	1776	1782	1812	8531#		
DT077	053462	1722	1728	1818	1824	1954	1960	8536#						
DT150	053476	1971	2007	8538#										
DT164	053512	2043	2049	2055	2061	8540#								
DT170	053530	2066	8543#											
DT171	053554	2071	2076	2081	8547#									
DT175	053564	2092	2098	2104	2110	2116	2122	2128	2134	2140	2146	2152	2158	8549#
DT211	053610	2164	2170	2176	2182	8553#								
ECCW	= 020000	1165#	2624	2640	2719	2793	2883	2990	5504	5587	5643	5707	5763	5826
		5882	5945	6001	6064	6120	6183	6239	6302	6358	6430	6488	6555	6618
		6624	6644	6701	6764	6770	6790	6851	6909					
ECH	= 000100	1125#												
EMTVEC	= 000030	1000#	2290*	2291*										
EMW	003170	2064#	5517*	5586*	5604*	5706*	5724*	5825*	5843*	5944*	5962*	6063*	6081*	6182*
		6200#	6301*	6319*	6429*	6447*	6554*	6572*	6592*	6605*	6700*	6718*	6738*	6751*
		6850#	6868*	7277*	7307*	7321*	7336*	7362*	7374*	7386*				
EMW1	066374	7277	9673#											
EMW2	066463	7307	7362	9683#										
EMW3	066554	7321	7374	9693#										
EMW4	066643	7336	7386	9703#										
EM000	057701	1963	9070#											
EM200	057746	1348	1354	1360	9077#									
EM201	060030	1366	1372	1378	9086#									
EM202	060113	1384	1390	1396	9095#									
EM203	060204	1402	1408	1414	9105#									
EM204	060276	1420	1426	1432	1438	1444	9115#							
EM205	060364	1450	1456	1462	1468	1474	1480	1486	1492	1498	1504	1510	9124#	
EM206	060452	1516	1522	1528	1534	1540	1546	1552	9133#					
EM207	060515	1558	1564	1570	1576	1606	1612	1618	9139#					

P.DA	003546	2227#												
P.DB	003562	2233#												
P.DCYL	003560	2232#												
P.DS	003552	2229#	6977*	7054*	7133*	8557								
P.ECPS	003572	2237#												
P.ECPT	003574	2238#												
P.ER	003554	2230#	6978*	7055*	7134*	8557								
P.MR1	003564	2234#												
P.MR2	003566	2235#												
P.MR3	003570	2236#												
P.SPAA	003576	2239#												
P.WC	003542	2225#												
PI.BIT	003612	2250#	5512*	5589*	5600*	5611	5614*	5617*	5628	5629*	5709*	5720*	5731	5734*
		5737#	5748	5749*	5828*	5839*	5850	5853*	5856*	5867	5868*	5947*	5958*	5969
		5972#	5975*	5986	5987*	6066*	6077*	6088	6091*	6094*	6105	6106*	6185*	6196*
		6207	6210*	6213*	6224	6225*	6304*	6315*	6326	6329*	6332*	6343	6344*	6432*
		6443*	6453	6456*	6459*	6470	6471*	6557*	6568*	6578	6581*	6584*	6596	6597*
		6602*	6610	6611*	6703*	6714*	6724	6727*	6730*	6742	6743*	6748*	6756	6757*
		6853*	6864*	6874	6877*	6880*	6891	6892*	7292	7346	8543			
RDBIT	046156	2629	2642	2654	2664	2723	2735	2746	2797	2809	2820	4674	4678	4691
		4710	4784	4788	4801	4820	4894	4898	4911	4930	5004	5008	5021	5040
		5114	5118	5131	5150	5223	5227	5240	5259	5328	5392	5396	5409	5428
		7402#												
RDCHR =	104410	8266	8492#											
RDDATA=	000021	1077#												
RDGATE=	100000	1167#	2640											
RDHEAD=	000025	1079#	2429	2439	2517	2527	2608	2615	2673	2697	2704	2754	2778	2785
		2628	4663	4681	4715	4773	4791	4825	4883	4901	4935	4993	5011	5045
		5103	5121	5155	5212	5230	5264	5315	5325	5346	5379	5399	5433	6943
		6959	7020	7036										
RDLIN =	104411	8340	8493#											
RDOCT =	104412	2354	2365	2382	8494#									
RDY =	000200	1088#	3665	3713	4263	4310	4378	4715	4825	4935	5045	5155	5264	5346
		5433	5653	5773	5892	6011	6130	6249	6368	6498	6655	6801	6919	6959
		6985	7036	7062	7115	7141								
RECAL =	000013	1074#												
RESREG=	104414	7811	8496#											
RESTRT	003652	1190	2268#											
RESVEC=	000010	994#												
RKASOF=	000016	1057#												
RKBA =	000004	1052#	2854*	2863	2899	2924	2956	2988	3037*	3060	3124*	3150	3181	3241*
		3265	3301	3370*	3394	3434	3477	3544*	3569	3603	3662*	3687	3785	3826*
		3848	3948	3989*	4011	4111	4152*	4174	4261*	4283	4327*	4349	4375	4420*
		4445	4481	4557*	4581	4606	5488*	5562*	5682*	5801*	5920*	6039*	6158*	6277*
		6403*	6528*	6674*	6824*	7098*								
RKCS1 =	000000	1050#	2424*	2429*	2436	2445*	2469*	2474*	2481	2490*	2512*	2517*	2524	2533*
		2556*	2561*	2568	2577*	2604*	2608*	2614	2619*	2634*	2647*	2659*	2669*	2672
		2693*	2694*	2697*	2703	2708*	2728*	2740*	2750*	2753	2774*	2775*	2778*	2784
		2789*	2802*	2814*	2824*	2827	2850*	2856*	2862	2898	2923	2955	2987	3003*
		3032*	3040*	3059	3084	3122*	3129*	3149	3179	3203	3238*	3244*	3264	3299
		3324	3367*	3373*	3393	3432	3454	3475	3505	3541*	3548*	3568	3601	3625
		3659*	3666*	3685	3710*	3711	3762*	3766*	3784	3821*	3828*	3847	3883*	3925*
		3929*	3947	3984*	3991*	4010	4046*	4088*	4092*	4110	4147*	4154*	4173	4209*
		4257*	4264*	4282	4307*	4308	4323*	4330*	4348	4373	4399*	4416*	4423*	4444
		4479	4507	4553*	4560*	4580	4604	4625	4635*	4661*	4663*	4692	4700*	4713
		4727	4771*	4773*	4802	4810*	4823	4837	4881*	4883*	4912	4920*	4933	4947

	4991*	4993*	5022	5030*	5043	5057	5101*	5103*	5132	5140*	5153	5167	5210*
	5212*	5241	5249*	5262	5276	5313*	5315*	5329	5344	5354*	5377*	5379*	5410
	5418*	5431	5445	5484*	5490*	5560*	5564*	5652	5680*	5684*	5772	5799*	5803*
	5891	5918*	5922*	6010	6037*	6041*	6129	6156*	6160*	6248	6275*	6279*	6367
	6401*	6406*	6497	6526*	6531*	6654	6672*	6677*	6800	6822*	6827*	6918	6943*
	6951	6955	6980*	6981	7020*	7028	7032	7057*	7058	7099*	7107	7111	7136*
	7137	7829*											
RKCS2 = 000010	1054*	2428*	2473*	2516*	2560*	3038*	3065	3085	3157	3180	3204	3272	3300
	3330	3401	3433	3458	3476	3511	3547*	3576	3602	3631	3686	3712	3790
	3855	3953	4018	4116	4181	4289	4309	4354	4374	4452	4480	4508	4586
	4605	4626	4714	4728	4824	4838	4934	4948	5044	5058	5154	5168	5263
	5277	5330	5345	5432	5446	6940*	6956	6982	7017*	7033	7059	7094*	7112
	7138												
RKDA = 000006	1053*	2427*	2472*	2515*	2559*	3036*							
RKDB = 000024	1059*	3079	3198	3319	3449	3500	3620	3806	3875	3969	4038	4132	4201
	4499	4620	4726	4836	4946	5056	5166	5275	5343	5444			
RKDCYL = 000020	1058*	2426*	2471*	2514*	2558*	3035*	6942*	7019*	7096*				
RKDS = 000012	1055*	6957	6983	7034	7060	7113	7139						
RKECPS = 000030	1063*												
RKECPT = 000032	1064*												
RKER = 000014	1056*	3689	4285	4377	6958	6984	7035	7061	7114	7140			
RKMR1 = 000026	1060*	2425*	2432*	2433*	2470*	2477*	2478*	2513*	2520*	2521*	2557*	2564*	2565*
	2605*	2606*	2607*	2610*	2611*	2622*	2623*	2630	2643	2655	2665	2695*	2696*
	2699*	2700*	2711*	2713*	2714*	2717*	2724	2736	2747	2776*	2777*	2780*	2781*
	2798	2810	2821	2851*	2852*	2853*	2858*	2859*	2886*	2887*	2888*	2889*	2890
	2917*	2918*	2919*	2922*	2943*	2944*	2945*	2946*	2947	2973*	2975*	2976*	2979*
	2982*	2983*	2986	3034*	3043*	3044*	3048*	3049*	3050*	3053*	3055*	3056*	3123*
	3132*	3133*	3137*	3138*	3139*	3142*	3145*	3146*	3175*	3176*	3239*	3247*	3248*
	3252*	3253*	3254*	3257*	3260*	3261*	3295*	3296*	3368*	3376*	3377*	3381*	3382*
	3383*	3386*	3389*	3390*	3428*	3429*	3466*	3467*	3542*	3551*	3552*	3556*	3557*
	3558*	3561*	3564*	3565*	3597*	3598*	3660*	3669*	3670*	3674*	3675*	3676*	3679*
	3681*	3682*	3763*	3768*	3769*	3773*	3774*	3775*	3778*	3780*	3781*	3822*	3830*
	3831*	3835*	3836*	3837*	3840*	3843*	3844*	3926*	3931*	3932*	3936*	3937*	3938*
	3941*	3943*	3944*	3985*	3993*	3994*	3998*	3999*	4000*	4003*	4006*	4007*	4089*
	4094*	4095*	4099*	4100*	4101*	4104*	4106*	4107*	4148*	4156*	4157*	4161*	4162*
	4163*	4166*	4169*	4170*	4258*	4266*	4267*	4271*	4272*	4273*	4276*	4278*	4279*
	4324*	4332*	4333*	4337*	4338*	4339*	4342*	4344*	4345*	4369*	4370*	4417*	4426*
	4427*	4431*	4432*	4433*	4436*	4440*	4441*	4473*	4474*	4554*	4563*	4564*	4568*
	4569*	4570*	4573*	4576*	4577*	4600*	4601*	4662*	4665*	4666*	4669*	4670*	4772*
	4775*	4776*	4779*	4780*	4882*	4885*	4886*	4889*	4890*	4992*	4995*	4996*	4999*
	5000*	5102*	5105*	5106*	5109*	5110*	5211*	5214*	5215*	5218*	5219*	5314*	5317*
	5318*	5321*	5322*	5378*	5381*	5382*	5385*	5386*	5387*	5388*	5485*	5486*	5487*
	5493*	5494*	5498*	5499*	5500*	5503*	5507*	5508*	5526*	5527	5533*	5534	5561*
	5567*	5568*	5572*	5573*	5574*	5577*	5579*	5580*	5583*	5584*	5635*	5637*	5638*
	5641*	5642	5648*	5649*	5681*	5687*	5688*	5692*	5693*	5694*	5697*	5699*	5700*
	5703*	5704*	5755*	5757*	5758*	5761*	5762	5768*	5769*	5800*	5806*	5807*	5811*
	5812*	5813*	5816*	5818*	5819*	5822*	5823*	5874*	5876*	5877*	5880*	5881	5887*
	5888*	5919*	5925*	5926*	5930*	5931*	5932*	5935*	5937*	5938*	5941*	5942*	5993*
	5995*	5996*	5999*	6000	6006*	6007*	6038*	6044*	6045*	6049*	6050*	6051*	6054*
	6056*	6057*	6060*	6061*	6112*	6114*	6115*	6118*	6119	6125*	6126*	6157*	6163*
	6164*	6168*	6169*	6170*	6173*	6175*	6176*	6179*	6180*	6231*	6233*	6234*	6237*
	6238	6244*	6245*	6276*	6282*	6283*	6287*	6288*	6289*	6292*	6294*	6295*	6298*
	6299*	6350*	6352*	6353*	6356*	6357	6363*	6364*	6402*	6409*	6410*	6414*	6415*
	6416*	6420*	6423*	6424*	6427*	6428*	6480*	6482*	6483*	6486*	6487	6493*	6494*
	6527*	6534*	6535*	6539*	6540*	6541*	6544*	6546*	6547*	6552*	6553*	6636*	6638*
	6639*	6642*	6643	6650*	6651*	6673*	6680*	6681*	6685*	6686*	6687*	6690*	6692*

BYPASS	1229#	3743	3906	4069	4242	4542											
CLMSG	1229#	2511	2555														
CLRPSM	1229#																
COMMEN	1005#																
ENDCOM	1005#																
ERROR	899#	2444	2449	2452	2489	2494	2497	2532	2537	2540	2576	2581	2584	2618	2633		
	2646	2658	2668	2676	2707	2727	2739	2757	2788	2801	2813	2831	2870	2875	2890		
	2893	2903	2908	2913	2928	2933	2938	2950	2960	2965	2970	2993	2996	2999	3002		
	3069	3072	3075	3078	3083	3089	3092	3162	3165	3168	3171	3185	3188	3191	3194		
	3202	3210	3213	3275	3278	3281	3284	3305	3308	3311	3314	3323	3336	3339	3404		
	3407	3410	3413	3438	3441	3444	3447	3453	3461	3464	3481	3484	3487	3490	3504		
	3517	3520	3579	3582	3585	3588	3607	3610	3613	3616	3624	3637	3640	3697	3700		
	3703	3706	3709	3717	3720	3796	3799	3802	3805	3810	3859	3862	3865	3868	3879		
	3959	3962	3965	3968	3973	4022	4025	4028	4031	4042	4122	4125	4128	4131	4136		
	4185	4188	4191	4194	4205	4294	4297	4300	4303	4306	4314	4317	4358	4361	4364		
	4367	4385	4388	4391	4394	4397	4458	4461	4464	4467	4485	4488	4491	4494	4503		
	4514	4517	4589	4592	4595	4598	4610	4613	4616	4619	4624	4630	4633	4699	4719		
	4722	4734	4739	4744	4809	4829	4832	4844	4849	4854	4919	4939	4942	4954	4959		
	4964	5029	5049	5052	5064	5069	5074	5139	5159	5162	5174	5179	5184	5248	5268		
	5271	5283	5288	5293	5333	5338	5350	5353	5417	5437	5440	5452	5457	5462	5520		
	5530	5537	5596	5602	5619	5631	5646	5656	5716	5722	5739	5751	5766	5776	5835		
	5841	5858	5870	5885	5895	5954	5960	5977	5989	6004	6014	6073	6079	6096	6108		
	6123	6133	6192	6198	6215	6227	6242	6252	6311	6317	6334	6346	6361	6371	6439		
	6445	6461	6473	6491	6501	6564	6570	6586	6595	6604	6613	6620	6627	6648	6658		
	6710	6716	6732	6745	6750	6759	6766	6773	6794	6804	6860	6866	6882	6894	6912		
	6922	6965	6968	6971	6974	6991	6994	6997	7000	7042	7045	7048	7051	7068	7071		
	7074	7077	7121	7124	7127	7130	7147	7150	7153	7156	7425						
ESCAPE	1005#																
FORERR	1229#	6939	7016	7053													
GETPRI	1005#	7458															
GETSWR	1005#	2331#															
LDLPER	1229#	3028	3118	3758	3817	3921	3980	4084	4143	4253	4319						
MSG	2409#	2411	2454#	2456	2499#	2501	2542#	2544	2589#	2591	2678#	2680	2759#	2761	2833#		
	2835	3007#	3009	3098#	3100	3223#	3225	3344#	3346	3525#	3527	3645#	3647	3722#	3724		
	3885#	3887	4048#	4050	4211#	4213	4402#	4404	4522#	4524	4639#	4641	4749#	4751	4859#		
	4861	4969#	4971	7079#	5081	5189#	5191	5298#	5300	5356#	5358	5469#	5471	5539#	5541		
	5658#	5660	5778#	5780	5897#	5899	6016#	6018	6135#	6137	6254#	6256	6373#	6375	6503#		
	6505	6660#	6662	6806#	6808	6926#	6928	7003#	7005	7080#	7082						
MULT	1005#																
NEWTST	1005#	2409	2454	2499	2542	2589	2678	2759	2833	3007	3098	3223	3344	3525	3645		
	3722	3985	4048	4211	4402	4522	4639	4749	4859	4969	5079	5189	5298	5356	5469		
	5539	5658	5778	5897	6016	6135	6254	6373	6503	6660	6806	6926	7003	7080			
POP	1005#	7684	7685	8051	8363	8412											
PUSH	1005#	7645	7647	7668	8010	8337	8392										
RDL00P	1229#	4660	4770	4880	4990	5100	5376										
REPORT	1005#																
SCOPE	900#	2421	2466	2509	2553	2601	2690	2771	2847	3020	3112	3234	3363	3537	3656		
	3737	3900	4063	4238	4413	4538	4658	4768	4878	4988	5098	5207	5310	5374	5481		
	5557	5677	5796	5915	6034	6153	6272	6398	6523	6669	6819	6937	7014	7091	7169		
SEKMSG	1229#	2423	2468														
SETPRI	1005#	8241															
SETTRA	8475#	8484	8485	8486	8487	8489	8491	8492	8493	8494	8495	8496	8497				
SETUP	1005#	2279															
SKIP	1005#	2446	2451	2491	2496	2534	2539	2578	2583	2620	2635	2648	2660	2670	2675		
	2709	2729	2741	2751	2756	2790	2803	2815	2825	2830	3096	3220	3719	4701	4735		
	4740	4811	4845	4850	4921	4955	4960	5031	5065	5070	5141	5175	5180	5250	5284		

B01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P3 MD-11-DZR6CA MACY11 27(1006) 28-JAN-77 14:16 PAGE 209
DZR6CB.P11 28-JAN-77 14:04 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0207

.SREAD	8718	8065
.SSAVE	8718	8374
.SSCOP	8718	7511
.SSIZE	8718	7434
.STRAP	8718	8452
.STYPO	8718	7998
.STYPE	8718	7842
.STYPO	8718	7921

. ABS. 067460 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DZR6CB.BIN, DSKZ:DZR6CB.SEQ/CRF/SOL/DOC=DZR6CB
RUN-TIME: 29 28 3 SECONDS
RUN-TIME RATIO: 211/61=3.4
CORE USED: 35K (70 PAGES)

DOCUMENT PAGES: 207

EOF1DZR6CBSEQ

00010000

770323

PDP10 411