

RK611

DISKLESS CONT NO. 5
MD-11-DZR6E-A

EP-DZR6E-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

DEC 1976
digital
MADE IN USA

This microfiche card contains 336 frames of document pages, arranged in a 14x24 grid. The frames are too small to read, but they appear to contain various types of text, including what might be a table of contents or a list of items. The card is dark, and the frames are light, creating a grid pattern.

1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3

- A. TESTS MULTI-SECTOR DATA TRANSFERS.
- B. TESTS MID-TRANSFER SEEKS.
- C. TESTS CYLINDER OVERFLOW CHECKING.
- D. TESTS NPR TRANSFERS TO MEMORY.
- E. TESTS ECC ERROR DETECTION AND CORRECTION, BOTH 16 AND 18 BIT MODE.
- F. TESTS WRITE CHECK BOTH 16 AND 18 BIT MODES AND FORCES WRITE CHECK 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)
- RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3 (DZR6C)
- RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 4 (DZR6D)

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

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

E01

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 1:40 MINUTES
SUBSEQUENT PASSES 3:40 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.

103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158

F01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 5
DZR6EA.P11 28-SEP-76 15:31

SEQ 0005

159
160
161
162
163
164
165
166
167

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 G (↑G) OPERATION

IF A ↑S HAS BEEN TYPED, TYPING THE ↑G CANCELS THE STALL
INITIATED BY THE ↑S.

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

5.0 PROGRAM DESCRIPTION

**MULTI-SECTOR WRITE OPERATIONS

TEST 1 MULTI-SECTOR WRITE (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR. PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION OF HEADER.

TEST 2 MULTI-SECTOR WRITE (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 23. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR. PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION OF HEADER.

TEST 3 MID-TRANSFER SEEK (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 25. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER. MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 4 MID-TRANSFER SEEK (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES

H01

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 7
DZR6EA.P11 28-SEP-76 15:31

SEG 0007

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
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275

AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 5 MID-TRANSFER SEEK (PART 3)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
CYLINDER 0, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES
AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 6 MID-TRANSFER SEEK (PART 4)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
CYLINDER 0, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK AND
DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES
IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 7 MID-TRANSFER SEEK AND 24 SECTOR FORMAT

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
DATA OF 401 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 23. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 18 BIT WORDS
AND THE 32 BIT ECC. CHECK THAT IMPLIED SEEK IS
ISSUE TO NEXT TRACK. CHECK FOR PROPER HEADER RECOGNITIO
OCCURS.

TEST 10 CYLINDER OVERFLOW (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 632, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE
TWO WORD ECC. MAKE SURE CYLINDER OVERFLOW ERROR DOES
NOT SET.

276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323

TEST 11 CYLINDER OVERFLOW (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT
CYLINDER 632, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.

TEST 12 CYLINDER OVERFLOW (PART 3)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET
CYLINDER 632, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW
ERROR DOES NOT SET.

TEST 13 CYLINDER OVERFLOW (PART 4)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET
CYLINDER 1456, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.

**NPR WRITING OF MEMORY

TEST 14 NPR WRITING OF MEMORY

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 40 DATA WORDS. VERIFY THAT THE WORDS
ARE WRITTEN PROPERLY IN MEMORY.

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
368
369
370

**ECC ERROR DETECTION/CORRECTION TESTS

TEST 15 READ DATA WITH NO ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.

TEST 16 READ DATA WITH ONE BIT BURST ERROR (PART 1)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
A ONE BIT BURST ERROR OF A 0 THAT SHOULD BE A 1.
VERIFY THE COUNTING OF THE POSITION
REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.

TEST 17 READ DATA WITH ONE BIT BURST ERROR (PART 2)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
A ONE BIT BURST ERROR OF A 1 THAT SHOULD BE A 0.
VERIFY THE COUNTING OF THE POSITION
REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.

TEST 20 READ DATA WITH 11 BIT BURST ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSJE A READ DATA
OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,

K01

CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE, A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING AN 11 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION. MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.

TEST 21 READ DATA WITH 12 BIT BURST ERROR
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE, A GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING A 12 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION REGISTER. MAKE SURE DATA CHECK, ECC HARD, AND CONTROLLER ERROR SETS.

TEST 22 READ DATA WITH 23 BIT BURST ERROR
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE, A GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING A 23 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION REGISTER. MAKE SURE DATA CHECK, ECC HARD, AND CONTROLLER ERROR SETS.

TEST 23 READ DATA WITH 1 BIT ECC WORD ERROR
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE, A GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING A 1 BIT ERROR IN ECC WORD 2, BIT 6. VERIFY THE COUNTING OF THE POSITION REGISTER. MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.

TEST 24 18 BIT NPR WRITING OF MEMORY
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK

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

L01

AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 40 DATA WORDS. VERIFY THAT THE WORDS
ARE WRITTEN PROPERLY IN MEMORY.

TEST 25 18 BIT READ DATA WITH NO ERROR
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.

TEST 26 18 BIT READ DATA WITH DCK
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
A ONE BIT BURST ERROR BEYOND THE 4128TH BIT OF THE SECTO
VERIFY THE COUNTING OF THE POSITION REGISTER AND THE
FINAL PATTERN AND POSITION. MAKE SURE DATA CHECK AND
CONTROLLER ERROR SETS.

3491 TEST 27 18 BIT READ DATA WITH ECH
CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS AND AND ECC INDICATING
A 12 BIT BURST ERROR. VERIFY THE COUNTING OF THE
POSITION REGISTER. MAKE SURE DATE CHECK, ECC HARD,
AND CONTROLLER ERROR SETS.

42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63

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

**SPECIAL READ TESTS

TEST 30 PARTIAL SECTOR READ

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
MAKE SURE ONLY 103 WORDS GOT TRANSFERRED TO MEMORY.

TEST 31 SILO UNLOAD BEFORE HEADER ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
AFTER ECC SIMULATE A SECTOR PULSE AND A BAD SECTOR ERROR
MAKE SURE THE ENTIRE PREVIOUS SECTOR GETS LOADED INTO
MEMORY BEFORE BAD SECTOR ERROR AND CONTROLLER ERROR SETS

**WRITE CHECK TESTS

TEST 32 WRITE CHECK OF 400 WORDS

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
AND A GOOD 32 BIT ECC. MAKE SURE NO ERRORS SET.

TEST 33 WRITE CHECK WITH DCK

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
AND A 32 BIT ECC INDICATING A 1 BIT BURST ERROR.
VERIFY THAT THE POSITION REGISTER INCREMENTS PROPERLY
AND THAT THE FINAL PATTERN AND POSITION ARE CORRECT.

CHECK THAT DATA CHECK AND CONTROLLER ERROR ARE SET.

TEST 34 WRITE CHECK OF 18 BIT WORDS

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 CHECK OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
 CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK,
 AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
 A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
 AND A GOOD 32 BIT ECC. MAKE SURE NO ERRORS SET.

TEST 35 WRITE CHECK OF A PARTIAL SECTOR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
 CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
 AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
 A GOOD HEADER, TAKE ONE WORD TO BE WRITE-CHECKED, 377 WO
 NOT TO BE WRITE CHECKED, AND A GOOD 32 BIT ECC.
 MAKE SURE NO ERRORS SET.

TEST 36 WRITE CHECK ERROR (PART 1)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
 CLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
 AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
 A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 000000
 WRITE CHECKED DATA IS 000001. MAKE SURE WRITE CHECK ERR
 REPEAT USING EACH OF THE FOLLOWING CONFIGURATIONS
 AS WRITE CHECKED DATA:

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

TEST 37 WRITE CHECK ERROR (PART 2)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
 CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
 AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
 A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 177777
 WRITE CHECKED DATA IS 177776. MAKE SURE WRITE
 CHECK ERROR SETS. REPEAT USING EACH OF THE FOLLOWING

516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571

B02

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 14
DZR6EA.P1: 28-SEP-76 15:31

SEQ 0014

572

573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606

CONFIGURATION AS WRITE CHECKED DATA:

177775	177737	176777	157777
177773	177677	175777	137777
177767	177577	173777	077777
177757	177377	167777	

TEST 40 WRITE CHECK ERROR (PART 3)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT. CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE, A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 20000. WRITE CHECK DATA IS 00000. MAKE SURE WRITE CHECK ERROR SETS. REPEAT USING 40000 AS SIMULATED DATA.

TEST 41 WRITE CHECK ERROR (PART 4)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT. CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES SIMULATE A SECTOR PULSE, A GOOD HEADER, 125 GOOD WORDS, AND A BAD WORD. MAKE SURE WRITE CHECK ERROR SET. CHECK WORD COUNT AND BUS ADDRESS.

607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652

DIAGNOSTIC MODE SECTOR PULSE

THE SECTOR PULSE IS PROVIDED BY THIS ROUTINE

DIAGNOSTIC MODE IMPLIED SEEK

THE CONTROLLER IS CLOCKED THROUGH THE SEEK MESSAGES,
GETTING STATUS, AND CLEARING THE DRIVE. SINCE THIS REQUI
DIFFERENT COUNT FOR WRITE DATA AND READ DATA/WRITE CHECK
THE ROUTINE LOOKS AT THE COMMAND THAT WAS
STARTED (L.CS1) AND SETS THE CLOCK COUNT ACCORDINGLY.

AT THE END OF THE IMPLIED SEEK THE CONTENTS OF CS1, MR1,
AND MR3 ARE CHECKED TO VERIFY THE OPERATION. ALL DISCREP
ARE REPORTED IN THE ORDER LISTED.

CALL:
JSR R4,DISEEK
ERROR 4 ;CS1 MISCOMPARE ERROR
ERROR 5 ;MR1 MISCOMPARE ERROR
ERROR 6 ;MR2 MISCOMPARE ERROR
ERROR 7 ;MR3 MISCOMPARE ERRON

DIAGNOSTIC MODE READ SYNC PATTERN (PREAMBLE)

THE 255 "0" BITS AND 1 "1" BIT ARE CLOCKED THROUTH THE R
THE CONTENTS OF MR1 ARE MONITORED TO WATCH FOR AN ERROR.
AT THE END OF THE SYNC READ, CS1 IS CHECKED TO INSURE IT
DID NOT GET CHANGED.

CALL:
JSR R4,DRSYNC
ERROR 10 ;MR1 CONTENTS IN ERROR

ERROR 11 ;CS1 CONTENTS IN ERROR

IF AN ERROR IS DETECTED, THE SUBROUTINE WILL SET UP TO A
THE PRESENT TEST UNLESS LOOP ON ERROR IS SET.

653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703

DIAGNOSTIC MODE HEADER READ AND COMPARE

THIS ROUTINE SIMULATES THE THREE HEADER WORDS READ AND COMPARE THAT IS PART OF ALL DATA TRANSFER OPERATIONS.

THE EXPECTED HEADER IS CALCULATED FROM THE CYLINDER, TRA AND SECTOR SPECIFIED IN THE "L" REGISTERS. THE READING OF 3 WORDS IS DONE AND MR1 IS MONITORED TO INSURE IT DOES NOT CHANGE. AT THE END OF THE READ/COMPARE, THE CONTENTS OF RKCSI IS CHECKED AND THE INCREMENT OF RKDCYL AND RKDA VERIFIED. IF THE INCREMENT IS GOOD, THE "L" REGISTERS ARE UPDATED TO THE NEW PACK ADDRESS IN CASE OF MULTI-SECTOR OPERATIONS. THE BAD SECTOR ERROR DESIRED (BSEDES) SWITCH IS TESTED AND IF IT IS NON-ZERO, THE 2ND AND 3RD HEADER WORDS ARE CHANGED ACCORDING TO BSEDES CONTENTS. THIS CAN BE USED TO FORCE HEADER TYPE ERROR.

CALL:
JSR R4,DHDCMP
ERROR 12 ;ERROR IN MR1
ERROR 13 ;ERROR IN CSI
ERROR 14 ;ERROR IN PACK ADDRESS

DIAGNOSTIC MODE GAP READ AND SYNC WRITE

THE READING OF THE GAP AND WRITING OF THE SYNC (OR PREAM) IS HANDLED IN THIS ROUTINE.

THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE ROUTINE AND THE REMAINDER ARE PROCESSED HERE. THEN A CHECK IS MADE THAT WRITE GATE CAME ON AND 255 "0" BITS AND A SINGLE "1" BIT IS WRITTEN.

CALL:
JSR R4,DWGPSN
ERROR 15 ;MR1 IN ERROR IN READ GAP
ERROR 16 ;CSI IN ERROR AFTER READ GAP
ERROR 17 ;MR1 IN ERROR IN WRITE SYNC

ERROR 20 ;CSI IN ERROR AFTER SYNC WRITE

ROUTINE CALLED:
JSR PC,WRTBIT
RETURN POINT IF ERROR IN WRTBIT
NO ERROR RETURN

704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745

DIAGNOSTIC WRITE DATA AND ECC ROUTINE

THE DATA IN THE OUPUT BUFFER IS WRITE SIMULATED, ECC IS
GENERATED AND WRITTEN, AND THE POSTAMBLE IS WRITTEN.

CALL:
JSR R4,DWRITE
ERROR 17 ;MR1 IN ERROR IN WRITING
ERROR 21 ;ECC ERROR IN WRITING
ERROR 22 ;CS1 ERROR AFTER WRITE
NO ERROR RETURN

ROUTINE CALLED:
JSR PC,WRTBIT
RETURN POINT IF ERROR IN WRTBIT
NO ERROR RETURN FROM WRTBIT

DIAGNOSTIC MODE READ GAP AND DATA SYNC

THE READING OF THE GAP AND READING OF THE SYNC (OR PREAM
IS HANDLED IN THIS ROUTINE.

THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
ROUTINE AND THE REMAINING GAP BITS ARE PROCESSED HERE.
THEN A CHECK IS MADE THAT READ GATE CAME ON
AT THE 129TH BIT OF SYNC.:

CALL:
JSR R4,DRGPSN
ERROR 15 ;MR1 IN ERROR IN READ GAP
ERROR 16 ;CS1 IN ERROR AFTER READ GAP

ERROR 32 ;MR1 IN ERROR IN READ SYNC
ERROR 33 ;CS1 IN ERROR AFTER SYNC READ

ROUTINE CALLED:
JSR PC,RDBIT

746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801

 DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE

THIS ROUTINE IS USED TO SIMULATE THE READ OR WRITE CHECK OF DATA. THE ROUTINE WILL TRANSFER A FULL OR PARTIAL SEC

THE FIRST WORD AFTER THE CALL SPECIFIES HOW MANY WORDS ARE TO BE TRANSFERRED.

IF LESS THAN A FULL SECTOR, THE ECC AT THE END OF THE TRANSFER IS NOT CHECKED. THE ECC IS CHECKED AT TH EACH BIT IS TRANSFERED.

EITHER 16 OR 18 BIT WORDS CAN BE SIMULATED DEPENDING ON THE STATE OF CFMT BIT IN L.CS1. IF 18 BITS ARE REQUIRED, BITS 16 AND 17 ARE ASSUMED TO BE ZERO.

THE CONTENTS OF LOCATION ECPOSX MUST BE LOADED WITH THE COUNT EXPECTED AT THE END OF THE DATA TRANSFER. THIS LOADING MUST OCCUR BEFORE THE ROUTINE IS CALLED.

ANOTHER LOCATION, ECPATX, IS SET TO 0 IF THE ECC IS EXPE TO BE CORRECT AND SET TO 1 IF AN ERROR IS BEING FORCED.

A 3RD LOCATION, ECCSRC, SPECIFIES THE SOURCE OF THE ECC WORDS TO BE USED IN CHECKING THE OPERATION. IF ECCSRC IS 1, THE ECC WORDS FED INTO THE DECODER ARE TAKEN FROM THE LAST TWO LOCATIONS OF OBUFF. IF ECCSRC IS 0, THE COMPUTE ECC WORDS FROM ECCHI AND ECCLO ARE USED.

CALL:

```
JSR      R4,DREAD
LENGTH OF TRANSFER
ERROR 34      ;MRI IN ERROR READING DATA OR ECC
ERROR 35      ;ECC ERROR READING DATA
ERROR 36      ;CSI ERROR AFTER READING DATA OR ECC
ERROR 41      ;ECC REGISTER INCORRECT AFTER READ ECC
ERROR 42      ;ERR IN ECC PAT CALCULATION
ERROR 43      ;ERP IN ECC POS COUNTING
```

OR

```
JSR      R4,DWRTCK
LENGTH OF TRANSFER
ERROR 53      ;MRI IN ERROR WRITE CHK OR ECC READ
ERROR 54      ;ECC ERROR IN WRITE CHECK
ERROR 55      ;CSI ERROR AFTER IN WRT CHK OR ECC READ
ERROR 41      ;ECC REGISTER INCORRECT AFTER READ ECC
ERROR 42      ;ERR IN ECC PAT CALCULATION
ERROR 43      ;ERR IN ECC POS COUNTING
```

ROUTINES CALLED:
 RDBIT
 ECCGEN

802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844

GENERATE ECC WORD

EACH TIME THIS ROUTINE IS CALLED THE ECC IS CALCULATED F
NEXT BIT TO BE READ OR WRITTEN AND THE BITS PREVIOUSLY
READ OR WRITTEN (STORED IN ECCHI AND ECCLO). THE RESULTS
THE CALCULATION ARE PLACED IN ECCHI AND ECCLO. THE LOW 0
11 BITS OF ECCHI ARE PLACED IN E.ECPT FOR EASY COMPARI
TO RKECPT.

CALL: JSR PC,ECCGEN
RETURN: RTS R4

SIMULATE ONE BIT OF WRITE DATA IN MAINTENANCE MODE

5835

ONE BIT OF DATA IS WRITE SIMULATED. THE CONTENTS OF
RKMRI IS COMPUTED (PRECOMP ADVANCE AND DELAY, MAINTENANC
ENCODED WRITE DATA) FOR THE PROPER TRANSITION OF THE
MAINTENANCE CLOCK AND CHECKED AT EACH TRANSITION. IF MRI
IS INCORRECT AT ANY TRANSITION, AN ERROR IS SET UP TO
CONTAIN A MESSAGE THAT IDENTIFIES THE TRANSITION WHEN TH
ERROR OCCURRED.

CALL: JSR PC,WRTBIT
RETURN: RTS PC+2 FOR NO ERROR RETURN
RTS PC FOR ERROR IN MRI

SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE

ONE BIT OF DATA IS READ SIMULATED. NO CHECKING IS
DONE BY THIS ROUTINE.

CALL: JSR PC,RDBIT
RETURN: RTS PC

845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884

MEMORY CHECK ENABLE TRAP

IF THE PROCESSOR IN USE HAS MEMORY PARITY OPTION,
IT WILL BE ENABLED. THIS ROUTINE WILL PROCESS TRAPS CAUS
BY MEMORY PARITY ERRORS.

CLEAR INPUT BUFFER

BUILD DATA BUFFER

THE PATTERN SPECIFIED IN THE CALL IS LOADED INTO THE DAT
BUFFER. THE ENTIRE BUFFER IS ALWAYS LOADED (400(8) WORDS

LOAD "L" REGISTERS

THE "L" REGISTERS ARE LOADED FORM THE PARAMETERS FOLLOWI
THE SUBROUTINE CALL. DIAGNOSTIC MODE IS SET IN L.MR1
AND L.CS1 IS LOADED WITH THE COMMAND.

CALL:
JSR R4,LOADRK
.WORD CYLINDER
.BYTE ;SECTOR
.BYTE ;TRACK
.WORD ;BUFFER ADDRESS
.WORD ;WORD COUNT
.WORD ;COMMAND,FORMAT,DRIVE TYPE,&UPPER BUS AD

RETURN:
RTS R4

885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918

START THE OPERATION

THE INFORMATION IN THE "L" REGISTERS ARE LOADED INTO THE
RK611 REGISTERS IN A STRAIGHT TRANSFER.

GET THE RK611 REGISTERS

ALL THE RK611 REGISTERS EXCEPT THE DATA BUFFER ARE
STORED IN THE "T" REGISTERS.

"FIRST SHALL BE LAST, LAST SHALL BE FIRST" SUBROUTINE

THE CONTENTS OF R3 IS SWAPPED END FOR END, I.E. BIT 15
BECOMES BIT 0 AND VICE VERSA, BIT 14 BECOMES BIT
1 AND VICE VERSA, ETC.

CALL:
JSR R4,FSBLVV
WITH R3 LOADED WITH THE WORD TO BE SWAPPED

RETURN:
RTS R4
WITH R3 SWAPPED.

CHECK FOR MEMORY CHECK ENABLE

```

919                                     %
920      .NLIST   CND,MD,MC,TOC
921      .LIST    ME
922      .ENABL   ABS,AMA
923      $SWR=    167400
924      $TN=     1
925      .TITLE   RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A
926      ;*COPYRIGHT (C) 1976
927      ;*DIGITAL EQUIPMENT CORP.
928      ;*MAYNARD, MASS. 01754
929      ;*
930      ;*PROGRAM BY MARV TEGROTHUIS
931      ;*
932      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
933      ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
934      ;*
935      .SBTTL   OPERATIONAL SWITCH SETTINGS
936      ;*
937      ;*      SWITCH                USE
938      ;*      -----                -----
939      ;*      15                    HALT ON ERROR
940      ;*      14                    LOOP ON TEST
941      ;*      13                    INHIBIT ERROR TYPEOUTS
942      ;*      12                    ABORT PROGRAM AFTER 20 ERRORS
943      ;*      11                    INHIBIT ITERATIONS
944      ;*      10                    BELL ON ERROR
945      ;*      9                     LOOP ON ERROR
946      ;*      8                     LOOP ON TEST IN SWR<7:0>
947      .SBTTL   BASIC DEFINITIONS
948      ;*
949      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
950      STACK=   1100
951      .EQUIV   EMT,ERROR           ;;BASIC DEFINITION OF ERROR CALL
952      .EQUIV   IOT,SCOPE           ;;BASIC DEFINITION OF SCOPE CALL
953      ;*
954      ;*MISCELLANEOUS DEFINITIONS
955      AT=      11                   ;;CODE FOR HORIZONTAL TAB
956      LF=      12                   ;;CODE FOR LINE FEED
957      CR=      15                   ;;CODE FOR CARRIAGE RETURN
958      CRLF=    200                  ;;CODE FOR CARRIAGE RETURN-LINE FEED
959      PS=      177776               ;;PROCESSOR STATUS WORD
960      .EQUIV   PS,PSW
961      STKLMT=  177774               ;;STACK LIMIT REGISTER
962      PIRQ=    177772               ;;PROGRAM INTERRUPT REQUEST REGISTER
963      DSWR=    177570               ;;HARDWARE SWITCH REGISTER
964      DDISP=   177570               ;;HARDWARE DISPLAY REGISTER
965      ;*
966      ;*GENERAL PURPOSE REGISTER DEFINITIONS
967      R0=      %0                   ;;GENERAL REGISTER
968      R1=      %1                   ;;GENERAL REGISTER
969      R2=      %2                   ;;GENERAL REGISTER
970      R3=      %3                   ;;GENERAL REGISTER
971      R4=      %4                   ;;GENERAL REGISTER
972      R5=      %5                   ;;GENERAL REGISTER
973      R6=      %6                   ;;GENERAL REGISTER
974      R7=      %7                   ;;GENERAL REGISTER

```



```

975      000006      SP=      %6      ;;STACK POINTER
976      000007      PC=      %7      ;;PROGRAM COUNTER
977
978      ;*PRIORITY LEVEL DEFINITIONS
979      000000      PR0=      0      ;;PRIORITY LEVEL 0
980      000040      PR1=      40     ;;PRIORITY LEVEL 1
981      000100      PR2=      100    ;;PRIORITY LEVEL 2
982      000140      PR3=      140    ;;PRIORITY LEVEL 3
983      000200      PR4=      200    ;;PRIORITY LEVEL 4
984      000240      PR5=      240    ;;PRIORITY LEVEL 5
985      000300      PR6=      300    ;;PRIORITY LEVEL 6
986      000340      PR7=      340    ;;PRIORITY LEVEL 7
987
988      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
989      100000      SW15=     100000
990      040000      SW14=     40000
991      020000      SW13=     20000
992      010000      SW12=     10000
993      004000      SW11=     4000
994      002000      SW10=     2000
995      001000      SW09=     1000
996      000400      SW08=     400
997      000200      SW07=     200
998      000100      SW06=     100
999      000040      SW05=     40
1000     000020      SW04=     20
1001     000010      SW03=     10
1002     000004      SW02=     4
1003     000002      SW01=     2
1004     000001      SW00=     1
1005     .EQUIV      SW09,SW9
1006     .EQUIV      SW08,SW8
1007     .EQUIV      SW07,SW7
1008     .EQUIV      SW06,SW6
1009     .EQUIV      SW05,SW5
1010     .EQUIV      SW04,SW4
1011     .EQUIV      SW03,SW3
1012     .EQUIV      SW02,SW2
1013     .EQUIV      SW01,SW1
1014     .EQUIV      SW00,SW0
1015
1016     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1017     100000      BIT15=    100000
1018     040000      BIT14=    40000
1019     020000      BIT13=    20000
1020     010000      BIT12=    10000
1021     004000      BIT11=    4000
1022     002000      BIT10=    2000
1023     001000      BIT09=    1000
1024     000400      BIT08=    400
1025     000200      BIT07=    200
1026     000100      BIT06=    100
1027     000040      BIT05=    40
1028     000020      BIT04=    20
1029     000010      BIT03=    10
1030     000004      BIT02=    4

```

```

1031      000002      BIT01= 2
1032      000001      BIT00= 1
1033      .EQUIV      BIT09,BIT9
1034      .EQUIV      BIT08,BIT8
1035      .EQUIV      BIT07,BIT7
1036      .EQUIV      BIT06,BIT6
1037      .EQUIV      BIT05,BIT5
1038      .EQUIV      BIT04,BIT4
1039      .EQUIV      BIT03,BIT3
1040      .EQUIV      BIT02,BIT2
1041      .EQUIV      BIT01,BIT1
1042      .EQUIV      BIT00,BIT0
1043
1044      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1045      000004      ERRVEC= 4          ;: TIME OUT AND OTHER ERRORS
1046      000010      RESVEC= 10         ;: RESERVED AND ILLEGAL INSTRUCTIONS
1047      000014      TBITVEC=14        ;: "T" BIT
1048      000014      TRTVEC= 14         ;: TRACE TRAP
1049      000014      BPTVEC= 14         ;: BREAKPOINT TRAP (BPT)
1050      000020      IOTVEC= 20         ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
1051      000024      PWRVEC= 24         ;: POWER FAIL
1052      000030      EMTVEC= 30         ;: EMULATOR TRAP (EMT) **ERROR**
1053      000034      TRAPVEC=34        ;: "TRAP" TRAP
1054      000060      TKVEC= 60          ;: TTY KEYBOARD VECTOR
1055      000064      TPVEC= 64          ;: TTY PRINTER VECTOR
1056      000240      PIRQVEC=240        ;: PROGRAM INTERRUPT REQUEST VECTOR
1057      .SBTTL      MEMORY MANAGEMENT DEFINITIONS
1058
1059      ;*KT11 VECTOR ADDRESS
1060
1061      000250      MMVEC= 250
1062
1063      ;*KT11 STATUS REGISTER ADDRESSES
1064
1065      177572      SR0= 177572
1066      177574      SR1= 177574
1067      177576      SR2= 177576
1068      172516      SR3= 172516
1069
1070      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1071
1072      172300      KIPDR0= 172300
1073      172302      KIPDR1= 172302
1074      172304      KIPDR2= 172304
1075      172306      KIPDR3= 172306
1076      172310      KIPDR4= 172310
1077      172312      KIPDR5= 172312
1078      172314      KIPDR6= 172314
1079      172316      KIPDR7= 172316
1080
1081      ;*KERNEL "I" PAGE ADDRESS REGISTERS
1082
1083      172340      KIPAR0= 172340
1084      172342      KIPAR1= 172342
1085      172344      KIPAR2= 172344
1086      172346      KIPAR3= 172346

```

```

1087      172350      KIPAR4= 172350
1088      172352      KIPAR5= 172352
1089      172354      KIPAR6= 172354
1090      172356      KIPAR7= 172356
1091
1092      000114      MEMVEC= 114      ;MEMORY PARITY VECTOR
1093      172100      MEMBAS= 172100  ;MEM PARITY OPTION
1094      000004      WR.PAR= 4       ;WRITE BAD PARITY
1095      000001      PAR.EN= 1       ;ENABLE PARITY ENABLE
1096      120210      AVECT1= 120210 ;DEFINE RK611 VECTOR ADDRESS
1097      000005      APRIOR= 5       ;DEFINE RK611 PRIORITY
1098      177440      ABASE= 177440  ;DEFINE BASE OF RK611 REGISTERS
1099
1100      .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1101
1102      00000C      RKCS1= 0        ;CONTROL AND STATUS REGISTER 1
1103      000002      RKWC= 2         ;WORD COUNT REGISTER
1104      000004      RKBA= 4         ;BUS ADDRESS REGISTER
1105      000006      RKDA= 6         ;DESIRED TRACK SECTOR REGISTER
1106      000010      RKCS2= 10       ;CONTROL AND STATUS REGISTER 2
1107      000012      RKDS= 12        ;DRIVE STATUS REGISTER
1108      000014      RKER= 14        ;ERROR REGISTER
1109      000016      RKASOF= 16      ;ATTENTION SUMMARY AND OFFSET REGISTER
1110      000020      RKDCYL=20       ;DESIRED CYLINDER REGISTER
1111      000024      RKDB= 24        ;DATA BUFFER
1112      000026      RKMR1= 26       ;MAINTENANCE REGISTER 1
1113      000034      RKMR2= 34       ;MAINTENANCE REGISTER 2
1114      000036      RKMR3= 36       ;MAINTENANCE REGISTER 3
1115      000030      RKECPS= 30     ;ECC POSITION INFORMATION
1116      000032      RKECPT= 32     ;ECC PATTERN INFORMATION
1117      000022      RKSPAR= 22     ;SPARE REGISTER
1118
1119      .SBTTL  DRIVE COMMANDS
1120
1121      000001      SELDRV= 01     ;SELECT DRIVE
1122      000003      PACK= 03       ;PACK ACKNOWLEDGE
1123      000005      CLEAR= 05      ;DRIVE CLEAR
1124      000007      UNLOAD= 07     ;UNLOAD
1125      000011      SRTSPL= 11     ;START SPINDLE
1126      000013      RECAL= 13      ;RECALIBRATE
1127      000015      OFFSET= 15     ;OFFSET
1128      000017      SEEK= 17       ;SEEK
1129      000021      RDDATA= 21    ;READ DATA
1130      000023      WRDATA= 23    ;WRITE DATA
1131      000025      RDHEAD= 25    ;READ HEADER
1132      000027      WRHEAD= 27    ;WRITE HEADER AND DATA
1133      000031      WRTCHK= 31    ;WRITE CHECK
1134      000300      INTR= 300     ;GENERATE INTERRUPT TO CPU
1135
1136      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1137
1138      000001      GO= BIT0       ;GO BIT
1139      000100      IE= BIT6       ;INTERRUPT ENABLE
1140      000200      RDY= BIT7      ;CONTROLLER READY
1141      000400      BA16= BIT8     ;BUS ADDRESS BIT 16
1142      001000      BA17= BIT9     ;BUS ADDRESS BIT 17
    
```

1143	002000	CDT=	BIT10	: CONTROLLER DRIVE TYPE (0=RK06,
1144	004000	CTC=	BIT11	: CONTROLLER TIMED OUT WAITING FOR
1145				: DRIVE RESPONSE
1146	010000	CFMT=	BIT12	: CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR),
1147	020000	SPAR=	BIT13	: DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1148	040000	DI=	BIT14	: DRIVE INTERRUPT
1149	100000	CERR=	BIT15	: CONTROLLER ERROR
1150	100000	CCLR=	BIT15	: CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1151	000007	DRVMSK=	7	: MASK FOR DRIVE SELECTION CODE
1152	000010	RLS=	BIT3	: DESELECT OR RELEASE DRIVE IN BITS 0-2
1153	000020	BAI=	BIT4	: BUS ADDRESS INCREMENT INHIBIT
1154	000040	SCLR=	BITS	: CLEAR CONTROLLER AND ALL DRIVES
1155	000100	IR=	BIT6	: INPUT READY
1156	000200	OR=	BIT7	: OUTPUT READY
1157	000400	UFE=	BIT8	: UNIT FIELD ERROR
1158	001000	MDS=	BIT9	: MULTIPLE DRIVE SELECT
1159	002000	PGE=	BIT10	: PROGRAMMING ERROR
1160	004000	NEM=	BIT11	: NON-EXISTENT MEMORY
1161	010000	NED=	BIT12	: NON-EXISTENT DRIVE
1162	020000	UPE=	BIT13	: UNIBUS PARITY ERROR
1163	040000	WCE=	BIT14	: WRITE CHECK ERROR
1164	100000	DLT=	BIT15	: DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1170	000001	ILF=	BIT0	: ILLEGAL FUNCTION CODE
1171	000002	SKI=	BIT1	: SEEK INCOMPLETE
1172	000004	NXF=	BIT2	: NON-EXECUTABLE DRIVE FUNCTION
1173	000010	DRPAR=	BIT3	: DRIVE DETECTED DRIVE BUS PARITY ERROR
1174	000020	FMTE=	BIT4	: FORMAT ERROR
1175	000040	DTYPE=	BIT5	: DRIVE TYPE ERROR
1176	000100	ECH=	BIT6	: ECC HARD
1177	000200	BSE=	BIT7	: BAD SECTOR ERROR
1178	000400	HVRC=	BIT8	: HEADER VRC ERROR
1179	001000	COE=	BIT9	: CYLINDER ADDRESS OVERFLOW ERROR
1180	002000	IDAE=	BIT10	: INVALID DISK ADDRESS ERROR
1181	004000	WLE=	BIT11	: WRITE LOCK ERROR
1182	010000	DTE=	BIT12	: DRIVE TIMING ERROR
1183	020000	OPI=	BIT13	: OPERATION (SEARCH) INCOMPLETE
1184	040000	UNS=	BIT14	: DRIVE UNSAFE
1185	100000	DCK=	BIT15	: DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1188	000001	DRA=	BIT0	: DRIVE AVAILABLE (CONTROLLER IS SET IF
1189				: THIS BIT IS RESET)
1190	000004	OFST=	BIT2	: DRIVE OFFSET
1191	000010	ACLO=	BIT3	: AC LOW
1192	000020	SPDLSS=	BIT4	: SPEED LOSS
1193	000040	DROT=	BIT5	: DRIVE OFF TRACK
1194	000100	VV=	BIT6	: VOLUME VALID
1195	000200	DRDY=	BIT7	: DRIVE READY
1196	000400	DDT=	BIT8	: DRIVE TYPE (0=RK06)

```

1199      004000      WRL=      BIT11      ;WRITE LOCK
1200      020000      PIP=      BIT13      ;POSITIONING IN PROGRESS
1201      040000      DSC=      BIT14      ;DRIVE STATUS CHANGE
1202      100000      SVAL=     BIT15      ;STATUS VALID
1203
1204      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1205
1206      000017      MESMSK= 17      ;MESSAGE MASK
1207
1208      000020      PAT=      BIT4      ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1209      000040      DMD=      BITS      ;DIAGNOSTIC MODE
1210      000100      MSP=      BIT6      ;MAINTENANCE SECTOR PULSE
1211      000200      MIND=     BIT7      ;MAINTENANCE INDEX
1212      000400      MCLK=     BIT8      ;MAINTENANCE CLOCK
1213      001000      MERD=     BIT9      ;MAINTENANCE ENCODED READ DATA
1214      002000      MEWD=     BIT10     ;MAINTENANCE ENCODED WRITE DATA
1215      004000      PCA=      BIT11     ;PRECOMPENSATION ADVANCE
1216      010000      PCD=      BIT12     ;PRECOMPENSATION DELAY
1217      020000      ECCW=     BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1218      040000      WRTGAT=  BIT14     ;WRITE GATE
1219      100000      RDGATE=  BIT15     ;READ GATE
1220
1221      .SBTTL  TRANSMITTED MESSAGE A
1222
1223      000020      S. SEEK=  BIT4      ;SEEK COMMAND
1224      000040      S. RECL=  BITS      ;RECALIBRATE COMMAND
1225      000100      S. STSP=  BIT6      ;START SPINDLE COMMAND
1226      000200      S. RTC=   BIT7      ;DRIVE RETURN TO CENTERLINE COMMAND
1227      000400      S. CLR=   BIT8      ;CLEAR ERROR AND DSC
1228      001000      S. FMT=   BIT9      ;FORMAT
1229      002000      S. UNLD=  BIT10     ;UNLOAD
1230      004000      S. PACK=  BIT11     ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1231
1232      .SBTTL  TRAP CATCHER
1233
1234      000000      . = 0
1235      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1236      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1237      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1238      000174      . = 174
1239      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1240      000176      000000      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
1241      000200      000137      002400      .SBTTL  STARTING ADDRESS(ES)
1242      000204      000137      002370      JMP      @*START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1243      000214      000214      JMP      RESTRT  ;JUMP TO RESTART ROUTINE
1244      000214      000137      002360      . = 214
1245      000214      000137      002360      JMP      PARM    ;JUMP TO OPERATOR ASSIGNED PARMETERS
1246
1247      .SBTTL  ACT11 HOOKS
1248
1249      ;*****
1250      ;HOOKS REQUIRED BY ACT11
1251      000220      000046      $SVPC= .      ;SAVE PC
1252      000046      025042      . = 46
1253      000046      025042      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1254      000052      000052      . = 52
1255      000052      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
1256      000220      000220      .=$SVPC      ;; RESTORE PC

```



```

1255      000114      000114
1256      000114      033524
1257      000116      000340
1258      000116      001000
1259
1260
1261
1262
1263
1264      001000
1265      000024      000024
1266      000024      000200
1267      000044      000044
1268      000044      001000
1269      001000
1270
1271
1272
1273
1274      001000
1275      001000      000000
1276      001002      001250
1277      001004      000132
1278      001006      001604
1279      001010      000144
1280      001012      000030

```

```

.=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.
$MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT:  .WORD 90.    ;;RUN TIM OF LONGEST TEST
$PASTM: .WORD 900.   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 100.   ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

E03

1281
1282
1283
1284
1285
1286
1287
1288 001100
1289 001100 000000
1290 001102 000
1291 001103 000
1292 001104 000000
1293 001106 000000
1294 001110 000000
1295 001112 000000
1296 001114 000
1297 001115 001
1298 001116 000000
1299 001120 000000
1300 001122 000000
1301 001124 000000
1302 001126 000000
1303 001130 000000
1304 001132 000000
1305 001134 000
1306 001135 000
1307 001136 000000
1308 001140 177570
1309 001142 177570
1310 001144 177560
1311 001146 177562
1312 001150 177564
1313 001152 177566
1314 001154 000
1315 001155 002
1316 001156 012
1317 001157 000
1318 001160 000000
1319
1320 001162 000000
1321 001164 000000
1322 001166 000000
1323 001170 000000
1324 001172 000000
1325 001174 000000
1326 001176 000000
1327 001200 000000
1328 001202 000000
1329 001204 000000
1330 001206 000000
1331 001210 000000
1332 001212 000000
1333 001214 000000
1334 001216 000000
1335 001220 000000
1336 001222 000000

.SBTTL COMMON TAGS

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

SCMTAG: . =1100

:: START OF COMMON TAGS
\$STNM: .WORD 0 :: CONTAINS THE TEST NUMBER
\$ERFLG: .BYTE 00 :: CONTAINS ERROR FLAG
\$ICNT: .WORD 000000 :: CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 000000 :: CONTAINS SCOPE LOOP ADDRESS
\$LPERR: .WORD 000000 :: CONTAINS SCOPE RETURN FOR ERRORS
\$ERTTL: .WORD 000000 :: CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 001 :: CONTAINS ITEM CONTROL BYTE
\$ERMAX: .BYTE 001 :: CONTAINS MAX. ERRORS PER TEST
\$ERRPC: .WORD 000000 :: CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 000000 :: CONTAINS ADDRESS OF 'GOOD' DATA
\$BDADR: .WORD 000000 :: CONTAINS ADDRESS OF 'BAD' DATA
\$GDDAT: .WORD 000000 :: CONTAINS 'GOOD' DATA
\$BDDAT: .WORD 000000 :: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED
\$AUTOB: .BYTE 000 :: AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 000 :: INTERRUPT MODE INDICATOR
\$SWR: .WORD 0DSWR :: ADDRESS OF SWITCH REGISTER
\$DISP: .WORD 0DISP :: ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 :: TTY KBD STATUS
\$TKB: 177562 :: TTY KBD BUFFER
\$TPS: 177564 :: TTY PRINTER STATUS REG. ADDRESS
\$TPB: 177566 :: TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0 :: CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2 :: CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12 :: INSERT FILL CHARS. AFTER A "LINE FEED"
\$TPFLG: .BYTE 0 :: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$REGAD: .WORD 0 :: CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
\$REGO: .WORD 0 :: CONTAINS ((\$REGAD)+0)
\$REG1: .WORD 000 :: CONTAINS ((\$REGAD)+2)
\$REG2: .WORD 000 :: CONTAINS ((\$REGAD)+4)
\$REG3: .WORD 000 :: CONTAINS ((\$REGAD)+6)
\$REG4: .WORD 000 :: CONTAINS ((\$REGAD)+10)
\$REG5: .WORD 000 :: CONTAINS ((\$REGAD)+12)
\$REG6: .WORD 000 :: CONTAINS ((\$REGAD)+14)
\$REG7: .WORD 000 :: CONTAINS ((\$REGAD)+16)
\$TMP0: .WORD 000 :: USER DEFINED
\$TMP1: .WORD 000 :: USER DEFINED
\$TMP2: .WORD 000 :: USER DEFINED
\$TMP3: .WORD 000 :: USER DEFINED
\$TMP4: .WORD 000 :: USER DEFINED
\$TMP5: .WORD 000 :: USER DEFINED
\$TMP6: .WORD 000 :: USER DEFINED
\$TMP7: .WORD 000 :: USER DEFINED
\$TMPIC: .WORD 000 :: USER DEFINED

1337	001224	000000	\$TMP11: .WORD	0	::USER DEFINED
1338	001226	000000	\$TMP12: .WORD	0	::USER DEFINED
1339	001230	000000	\$TMP13: .WORD	0	::USER DEFINED
1340	001232	000000	\$TMP14: .WORD	0	::USER DEFINED
1341	001234	000000	\$TIMES: 0		::MAX. NUMBER OF ITERATIONS
1342	001236	000000	\$ESCAPE: 0		::ESCAPE ON ERROR ADDRESS
1343	001240	177607	\$BELL: .ASCIZ	<207><377><377>	::CODE FOR BELL
1344	001244	077	\$QUES: .ASCII	??	::QUESTION MARK
1345	001245	015	\$CRLF: .ASCII	<15>	::CARRIAGE RETURN
1346	001246	000012	\$LF: .ASCIZ	<12>	::LINE FEED
1347			:*****		
1348			:SBTTL APT MAILBOX-ETABLE		
1349			:*****		
1350			:EVEN		
1351			:MAIL: .WORD		
1352	001250		\$MSGTY: .WORD	AMSGTY	::APT MAILBOX
1353	001250	000000	\$FATAL: .WORD	AFATAL	::MESSAGE TYPE CODE
1354	001252	000000	\$TESTN: .WORD	ATESTN	::FATAL ERROR NUMBER
1355	001254	000000	\$PASS: .WORD	APASS	::TEST NUMBER
1356	001256	000000	\$DEVCT: .WORD	ADEVCT	::PASS COUNT
1357	001260	000000	\$UNIT: .WORD	AUNIT	::DEVICE COUNT
1358	001262	000000	\$MSGAD: .WORD	AMSGAD	::I/O UNIT NUMBER
1359	001264	000000	\$MSGLG: .WORD	AMSGLG	::MESSAGE ADDRESS
1360	001266	000000	\$ETABLE: .WORD		::MESSAGE LENGTH
1361	001270		\$ENV: .BYTE	AENV	::APT ENVIRONMENT TABLE
1362	001270	000	\$ENVM: .BYTE	AENVM	::ENVIRONMENT BYTE
1363	001271	000	\$SWREG: .WORD	ASWREG	::ENVIRONMENT MODE BITS
1364	001272	000000	\$USWR: .WORD	AUSWR	::APT SWITCH REGISTER
1365	001274	000000	\$CPUOP: .WORD	ACPUOP	::USER SWITCHES
1366	001276	000000			::CPU TYPE, OPTIONS
1367					BITS 15-11=CPU TYPE
1368					11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1369					11/70=06, PDQ=07, Q=10
1370					BIT 10=REAL TIME CLOCK
1371					BIT 9=FLOATING POINT PROCESSOR
1372					BIT 8=MEMORY MANAGEMENT
1373	001300	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS, M.S. BYTE
1374	001301	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE, BLK#1
1375					MEM. TYPE BYTE -- (HIGH BYTE)
1376					900 NSEC CORE=001
1377					300 NSEC BIPOLAR=002
1378					500 NSEC MOS=003
1379	001302	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS, BLK#1
1380					MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1381	001304	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS, M.S. BYTE
1382	001305	000	\$MTYP2: .BYTE	AMTYP2	::MEM. TYPE, BLK#2
1383	001306	000000	\$MADR2: .WORD	AMADR2	::MEM. LAST ADDRESS, BLK#2
1384	001310	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS, M.S. BYTE
1385	001311	000	\$MTYP3: .BYTE	AMTYP3	::MEM. TYPE, BLK#3
1386	001312	000000	\$MADR3: .WORD	AMADR3	::MEM. LAST ADDRESS, BLK#3
1387	001314	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS, M.S. BYTE
1388	001315	000	\$MTYP4: .BYTE	AMTYP4	::MEM. TYPE, BLK#4
1389	001316	000000	\$MADR4: .WORD	AMADR4	::MEM. LAST ADDRESS, BLK#4
1390	001320	120210	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1, BUS PRIORITY#1
1391	001322	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2, BUS PRIORITY#2
1392	001324	177440	\$BASE: .WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST

G03

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZ6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 32
DZ6EA.P11 28-SEP-76 15:31 APT MAILBOX-ETABLE

SEQ 0032

1393 001326 000000
1394 001330
1395

SDEV: .WORD ADEV: ;;DEVICE MAP
SETEND:
.MEXIT

1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410 001330
1411
1412 001330 044512
1413 001332 043071
1414 001334 041316
1415 001336 041634
1416
1417 001340 000000
1418 001342 000000
1419 001344 000000
1420 001346 000000
1421
1422 001350 000000
1423 001352 000000
1424 001354 000000
1425 001356 000000
1426
1427 001360 044557
1428 001362 044623
1429 001364 041322
1430 001366 041640
1431
1432
1433 001370 044557
1434 001372 044642
1435 001374 041340
1436 001376 041670
1437
1438
1439 001400 044557
1440 001402 044661
1441 001404 041350
1442 001406 041714
1443
1444
1445 001410 044557
1446 001412 044700
1447 001414 041360
1448 001416 041740
1449
1450
1451 001420 044771

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

; ERROR 1 ;UNEXPECTED PARITY ERROR
EM000
DH000C
DT000
DF000
; ERROR 2 ;UNUSED
0
0
0
0
; ERROR 3 ;UNUSED
0
0
0
0
; ERROR 4 ;ERROR ATTEMPTING IMPLIED SEEK
;CSI IN ERROR
EM5004
E5004A
DT0004
DF0004
; ERROR 5 ;MR1 IN ERROR ATTEMPTING IMPLIED SEEK
EM5004
E5004B
DT0005
DF0005
; ERROR 6 ;MR2 IN ERROR ATTEMPTING IMPLIED SEEK
EM5004
E5004C
DT0006
DF0006
; ERROR 7 ;MR3 IN ERROR ATTEMPTING IMPLIED SEEK
EM5004
E5004D
DT0007
DF0007
; ERROR 10 ;MR1 ERROR READING PREAMBLE
EM5010

1452	001422	044642	E50048	
1453	001424	041370	DT0010	
1454	001426	041764	DF0010	
1455				
1456				
1457	001430	044771	ERROR 11	;CSI ERROR READING PREAMBLE
1458	001432	044623	EM5010	
1459	001434	041406	E5004A	
1460	001436	042010	DT0011	
1461			DF0011	
1462				
1463	001440	045051	ERROR 12	;MR1 IN ERROR HEADER READ/COMPARE
1464	001442	044642	EM5011	
1465	001444	041370	E50048	
1466	001446	041764	DT0010	
1467			DF0010	
1468				
1469	001450	045051	ERROR 13	;CSI IN ERROR HEADER READ/COMPARE
1470	001452	044623	EM5011	
1471	001454	041406	E5004A	
1472	001456	042010	DT0011	
1473			DF0011	
1474				
1475	001460	045051	ERROR 14	;PACK ADDRESS ERROR HDR READ/COMPARE
1476	001462	045124	EM5011	
1477	001464	041432	E5011A	
1478	001466	042040	DT0014	
1479			DF0014	
1480				
1481	001470	045154	ERROR 15	;MR1 ERROR READING GAP
1482	001472	044642	EM5015	
1483	001474	041370	E50048	
1484	001476	041764	DT0010	
1485			DF0010	
1486				
1487	001500	044771	ERROR 16	;CSI ERROR AFTER READING GAP
1488	001502	044623	EM5010	
1489	001504	041406	E5004A	
1490	001506	042010	DT0011	
1491			DF0011	
1492				
1493	001510	000000	ERROR 17	;ERROR IN WRITING
1494	001512	000000	0	;REGISTER IN ERROR AND CLOCK
1495	001514	041446	0	;TRANSITION IS IDENTIFIED
1496	001516	042064	DT0017	
1497			DF0017	
1498				
1499	001520	045211	ERROR 20	;ERROR IN CSI AFTER WRITING SYNC
1500	001522	044623	EM5020	
1501	001524	041406	E5004A	
1502	001526	042010	DT0011	
1503			DF0011	
1504				
1505	001530	045270	ERROR 21	;ERROR IN ECC WORDS
1506	001532	045336	EM5021	
1507	001534	041470	E5021A	
			DT0021	

1508	001536	042110	DF0021	
1509				
1510			;	ERROR 22 ;ERR IN CS1 AFTER WRITE DATA
1511	001540	045270	EM5021	
1512	001542	044623	E5004A	
1513	001544	041406	DT0011	
1514	001546	042010	DF0011	
1515				
1516			;	ERROR 23 ;ERROR IN MR2
1517	001550	045366	EM5023	;DOING MULTI-SECTOR WRITE
1518	001552	044661	E5004C	
1519	001554	041350	DT0006	
1520	001556	041714	DF0006	
1521				
1522			;	ERROR 24 ;MR3 IN ERROR
1523	001560	045366	EM5023	;DOING MULTI-SECTOR WRITE
1524	001562	044700	E5004D	
1525	001564	041360	DT0007	
1526	001566	041740	DF0007	
1527				
1528			;	ERROR 25 ;CS1 IN ERROR
1529	001570	045366	EM5023	;AFTER MULTI-SECTOR WRITE
1530	001572	044623	E5004A	
1531	001574	041322	DT0004	
1532	001576	041640	DF0004	
1533				
1534			;	ERROR 26 ;NO COE ERROR
1535	001600	046006	EM5026	
1536	001602	000000	.WORD 0	
1537	001604	041502	DT0026	
1538	001606	042134	DF0026	
1539				
1540			;	ERROR 27 ;OTHER ERROR WHILE FORCING COE
1541	001610	046146	EM5027	
1542	001612	043040	DH000A	
1543	001614	041502	DT0026	
1544	001616	042134	DF0026	
1545				
1546			;	ERROR 30 ;UNEXPECTED CYL OVERFLOW
1547	001620	046320	EM5030	
1548	001622	000000	.WORD 0	
1549	001624	041502	DT0026	
1550	001626	042134	DF0026	
1551				
1552			;	ERROR 31 ;UNEXPECTED ERROR TESTING COE
1553	001630	046413	EM5031	
1554	001632	000000	.WORD 0	
1555	001634	041502	DT0026	
1556	001636	042134	DF0026	
1557				
1558			;	ERROR 32 ;MR1 ERROR READING DATA SYNC
1559	001640	046500	EM5032	
1560	001642	044642	E50048	
1561	001644	041370	DT0010	
1562	001646	041764	DF0010	
1563				

K03

1564			:	ERROR 33	;CS1 ERROR AFTER READING SYNC
1565	001650	046500		EMS032	
1566	001652	044623		ES004A	
1567	001654	041406		DT0011	
1568	001656	042010		DF0011	
1569					
1570			:	ERROR 34	;MRI IN ERROR READING DATA OR ECC
1571	001660	046556		EMS034	
1572	001662	044642		ES0048	
1573	001664	041370		DT0010	
1574	001666	041764		DF0010	
1575					
1576			:	ERROR 35	;ECC ERROR READING DATA
1577	001670	046623		EMS035	
1578	001672	045336		ES021A	
1579	001674	041470		DT0021	
1580	001676	042110		DF0021	
1581					
1582			:	ERROR 36	;CS1 ERROR AFTER READING DATA OR ECC
1583	001700	046556		EMS034	
1584	001702	044623		ES004A	
1585	001704	041406		DT0011	
1586	001706	042010		DF0011	
1587					
1588			:	ERROR 37	;DATA COMPARE ERROR (1ST MESSAGE)
1589	001710	046664		EMS037	
1590	001712	043040		DH000A	
1591	001714	041524		DT0037	
1592	001716	042170		DF0037	
1593					
1594			:	ERROR 40	;DATA COMPARE ERROR (2ND THRU 10TH ERR)
1595	001720	000000		0	
1596	001722	000000		0	
1597	001724	041530		DT0040	
1598	001726	042210		DF0040	
1599					
1600			:	ERROR 41	;ECC REGISTER INCORRECT AFTER ECC READ
1601	001730	046740		EMS041	
1602	001732	043040		DH000A	
1603	001734	041536		DT0041	
1604	001736	042214		DF0041	
1605					
1606			:	ERROR 42	;ERROR IN ECC PAT CALCULATION AFTER ECC ERROR
1607	001740	047025		EMS042	
1608	001742	043040		DH000A	
1609	001744	041470		DT0021	
1610	001746	042234		DF0042	
1611					
1612			:	ERROR 43	;ERROR IN ECC POSITION COUNTING AFTER ECC ERROR
1613	001750	047106		EMS043	
1614	001752	043040		DH000A	
1615	001754	041554		DT0043	
1616	001756	042254		DF0043	
1617					
1618			:	ERROR 44	;ERROR AFTER PROCESSING DATA CHECK ERR
1619	001760	047165		EMS044	

1620	001762	044623	E5004A	
1621	001764	041322	DT0004	
1622	001766	041640	DF0004	
1623				
1624			ERROR 45	
1625	001770	047165	EMS044	;ERROR AFTER PROCESSING DATA CHECK ERR
1626	001772	044717	E5004E	
1627	001774	041566	DT0045	
1628	001776	042274	DF0045	
1629				
1630			ERROR 46	
1631	002000	044557	EMS0046	;TRANSFER LENGTH ERROR PARTIAL SEC READ
1632	002002	043040	DH000A	
1633	002004	041576	DT0046	
1634	002006	042320	DF0046	
1635				
1636			ERROR 47	
1637	002010	047327	EMS047	;BSE DID NOT PREVENT DA OR DC INCREMENT
1638	002012	045121	E5011A	
1639	002014	041432	DT0014	
1640	002016	042040	DF0014	
1641				
1642			ERROR 50	
1643	002020	047422	EMS050	;BSE DID NOT CAUSE CERR
1644	002022	044623	E5004A	
1645	002024	041322	DT0004	
1646	002026	042340	DF0050	
1647				
1648			ERROR 51	
1649	002030	047475	EMS051	;BSE FAILED TO SET WHEN EXPECTED
1650	002032	044717	E5004E	
1651	002034	041566	DT0045	
1652	002036	042274	DF0045	
1653				
1654			ERROR 52	
1655	002040	047552	EMS052	;DATA XFER ABORT TO SOON
1656	002042	043040	DH000A	
1657	002044	041576	DT0046	
1658	002046	042320	DF0046	
1659				
1660			ERROR 53	
1661	002050	047672	EMS053	;MRI ERROR IN WRT CHK OR ECC READ AFTER WRT CHK
1662	002052	044642	E50048	
1663	002054	041370	DT0010	
1664	002056	041764	DF0010	
1665				
1666			ERROR 54	
1667	002060	047761	EMS054	;ECC ERROR IN WRITE CHECK OP
1668	002062	045336	E5021A	
1669	002064	041470	DT0021	
1670	002066	042110	DF0021	
1671				
1672			ERROR 55	
1673	002070	047672	EMS053	;CS1 ERROR AFTER WRT CHK OR ECC READ IN WRT CHK
1674	002072	044623	E5004A	
1675	002074	041406	DT0011	

1676	002076	042010	DF0011	
1677				
1678			;	ERROR 56
1679	002100	046556	EM5034	;CS1 ERROR AFTER READ DATA OR ECC
1680	002102	044623	E5004A	
1681	002104	041322	DT0004	
1682	002106	041640	DF0004	
1683				
1684			;	ERROR 57
1685	002110	047672	EM5053	;CS1 ERROR AFTER WRITE CHECK OR ECC READ
1686	002112	044623	E5004A	
1687	002114	041322	DT0004	
1688	002116	041640	DF0004	
1689				
1690			;	ERROR 60
1691	002120	050024	EM5060	;WCE FAILED TO SET IN 16 BIT MODE
1692	002122	043040	DH000A	
1693	002124	041524	DT0037	
1694	002126	042364	DF0060	
1695				
1696			;	ERROR 61
1697	002130	050103	EM5061	;TRANSFER LENGTH PROBLEM - RKBA
1698	002132	044735	E5004F	
1699	002134	041576	DT0046	
1700	002136	042404	DF0061	
1701				
1702			;	ERROR 62
1703	002140	050103	EM5061	;TRANSFER LENGTH PROBLEM - RKWC
1704	002142	044753	E5004G	
1705	002144	041606	DT0062	
1706	002146	042430	DF0062	
1707				
1708			;	ERROR 63
1709	002150	050160	EM5063	;WCE FAILED TO SET IN 18 BIT MODE
1710	002152	044753	E5004G	
1711	002154	041616	DT0063	
1712	002156	042454	DF0063	

1713				
1714			.SBTTL	TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
1715	002160	000000	T.CS1: .WORD	0 ;CONTROL AND STATUS REGISTER 1
1716	002162	000000	T.WC: .WORD	0 ;WORD COUNT REGISTER
1717	002164	000000	T.BA: .WORD	0 ;BUS ADDRESS REGISTER
1718	002166	000000	T.DA: .WORD	0 ;DESIRED TRACK SECTOR REGISTER
1719	002170	000000	T.CS2: .WORD	0 ;CONTROL AND STATUS REGISTER 2
1720	002172	000000	T.DS: .WORD	0 ;DRIVE STATUS REGISTER
1721	002174	000000	T.ER: .WORD	0 ;ERROR REGISTER
1722	002176	000000	T.ASOF: .WORD	0 ;ATTENTION SUMMARY AND OFFSET REGISTER
1723	002200	000000	T.DCYL: .WORD	0 ;DESIRED CYLINDER REGISTER
1724	002202	000000	T.DB: .WORD	0 ;DATA BUFFER
1725	002204	000000	T.MR1: .WORD	0 ;MAINTENANCE REGISTER 1
1726	002206	000000	T.MR2: .WORD	0 ;MAINTENANCE REGISTER 2
1727	002210	000000	T.MR3: .WORD	0 ;MAINTENANCE REGISTER 3
1728	002212	000000	T.ECPS: .WORD	0 ;ECC POSITION INFORMATION
1729	002214	000000	T.ECPT: .WORD	0 ;ECC PATTERN INFORMATION
1730	002216	000000	T.SPARE: .WORD	0 ;SPARE REGISTER
1731				

```

1732          .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS
1733
1734 002220 000000  E.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
1735 002222 000000  E.WC:  .WORD 0 ;WORD COUNT REGISTER
1736 002224 000000  E.BA:  .WORD 0 ;BUS ADDRESS REGISTER
1737 002226 000000  E.DA:  .WORD 0 ;DESIRED TRACK SECTOR REGISTER
1738 002230 000000  E.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
1739 002232 000000  E.DS:  .WORD 0 ;DRIVE STATUS REGISTER
1740 002234 000000  E.ER:  .WORD 0 ;ERROR REGISTER
1741 002236 000000  E.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
1742 002240 000000  E.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
1743 002242 000000  E.DB:  .WORD 0 ;DATA BUFFER
1744 002244 000000  E.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
1745 002246 000000  E.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
1746 002250 000000  E.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
1747 002252 000000  E.ECPS: .WORD 0 ;ECC POSITION INFORMATION
1748 002254 000000  E.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
1749 002256 000000  E.SPAR: .WORD 0 ;SPARE REGISTER
1750
1751          .SBTTL  "L" REGISTERS FOR COMMAND START
1752 002260 000000  L.CS1: .WORD 0 ;CONTROL AND STATUS 1
1753 002262 000000  L.WC:  .WORD 0 ;WORD COUNT
1754 002264 000000  L.BA:  .WORD 0 ;BUFER ADDRESS
1755 002266          L.DA:          ;DESIRED
1756 002266          L.DS:  .BYTE 0 ;SECTOR
1757 002267          L.DT:  .BYTE 0 ;TRACK
1758 002270 000000  L.CS2: .WORD 0 ;CONTROL AND STATUS
1759 002272 000000  L.ASOF: .WORD 0 ;ATTENTION AND OFFSET
1760 002274 000000  L.DCYL: .WORD 0 ;DESIRED CYLINDER
1761 002276 000000  L.MR1: .WORD 0 ;MAINT. REG 1
1762 002300 000000  L.MR2: .WORD 0 ;MAINT. REG 2
1763 002302 000000  L.MR3: .WORD 0 ;MAINT. REG 3
1764
1765
1766
1767          .SBTTL  PROGRAM DEFINED VARIABLES
1768
1769 002304 000210  RKVEC: .WORD 210 ;RK611 VECTOR
1770 002306 000240  RKPRI: .WORD PRS ;RK611 PRIORITY
1771 002310 000000  TRAPPC: .WORD 0 ;PC FOR MEMORY CHECK ENABLE TRAP
1772 002312 000000  SRTFLG: .WORD 0 ;START FLAG
1773          ; 0 = 200
1774          ; 1 = 214
1775          ; -1 = 204
1776 002314 000000  ERRCNT: .WORD 0 ;ERROR COUNT FOR SWITCH 12 ABORT
1777 002316 000000  P1.BIT: .WORD 0 ;NEXT BIT IN DATA SIMULATION
1778 002320 000000  PR.BIT: .WORD 0 ;PRESENT BIT IN DATA SIMULATION
1779 002322 000000  M1.BIT: .WORD 0 ;PREVIOUS BIT IN DATA SIMULATION
1780 002324 000000  M2.BIT: .WORD 0 ;BIT BEFORE PREVIOUS BIT
1781 002326 000000  BITCNT: .WORD 0 ;BIT POSITION
1782 002330 000000  WRDCNT: .WORD 0 ;WORD COUNT FOR NPR TRANSFER
1783 002332 000000  MEMPAR: .WORD 0 ;MEMORY EMABLE ON FIRST 24K
1784 002334 000000  BADPAR: .WORD 0 ;BAD PARITY FLAG
1785 002336 000000  ECCXOR: .WORD 0 ;ECC XOR SWITCH
1786 002340 000000  ECCHI:  .WORD 0 ;ECC HIGH STORAGE
1787 002342 000000  ECCLO:  .WORD 0 ;ECC LOW STORAGE
    
```

1788	002344	000000	ECCSRC: .WORD	0	:ECC SOURCE FLAG
1789	002346	000000	ECPATX: .WORD	0	:EXPECTED PATTERN ON ERROR
1790	002350	000000	ECPOSX: .WORD	0	:EXPECTED POSITION ON ERROR
1791	002352	000000	BSEDES: .WORD	0	:BAD SECTOR DESIRED
1792	002354	000000	HYBTS: .WORD	0	:HI ORDER BITS FOR 18 BIT WORD XFER
1793	002356	000000	SAVSWR: .WORD	0	:STORAGE FOR SWITCH REGISTER

```

1794 .SBTTL PROGRAM SETUP
1795
1796 002360 012737 000001 002312 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
1797 002366 000406 BR START1
1798
1799 002370 012737 177777 002312 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
1800 002376 000402 BR START1
1801
1802 002400 005037 002312 START: CLR SRTFLG ;CLEAR START FLAG
1803 002404 000005 START1: RESET ;RESET THE WHOLE SYSTEM
1804 002406 105037 001103 CLR#B $ERFLG ;CLEAR ERROR FLAG
1805 002412 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
1806 002416 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
1807 002422 012746 002430 MOV #1$,-(SP) ;LOAD START OF PROGRAM
1808 002426 000002 RTI ;LOAD PSW
1809
1810 002430 004737 037410 1$: JSR PC,#$STKINT ;INIT KEYBOARD
1811 .SBTTL INITIALIZE THE COMMON TAGS
1812 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1813 002434 012706 001100 MOV #SCMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
1814 002440 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
1815 002442 022706 001140 CMP #SWR,R6 ;;DONE?
1816 002446 001374 BNE -6 ;;LOOP BACK IF NO
1817 002450 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
1818 ;;INITIALIZE A FEW VECTORS
1819 002454 012737 034756 000020 MOV #SCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1820 002462 012737 000340 000022 MOV #340,#IOTVEC+2 ;;LEVEL 7
1821 002470 012737 035666 000030 MOV #ERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1822 002476 012737 000340 000032 MOV #340,#EMTVEC+2 ;;LEVEL 7
1823 002504 012737 041226 000034 MOV #TRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1824 002512 012737 000340 000036 MOV #340,#TRAPVEC+2;LEVEL 7
1825 002520 012737 041072 000024 MOV #SPWRDN,#PWRVEC ;;POWER FAILURE VECTOR
1826 002526 012737 000340 000026 MOV #340,#PWRVEC+2 ;;LEVEL 7
1827 002534 013737 024706 024700 MOV SENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
1828 002542 005037 001234 CL#TIMES ;;INITIALIZE NUMBER OF ITERATIONS
1829 002546 005037 001236 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1830 002552 012737 000001 001115 MOV#B #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
1831 002560 012737 002560 001106 MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1832 002566 012737 002566 001110 MOV #,$SLPERR ;;SETUP THE ERROR LOOP ADDRESS
1833 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1834 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1835 002574 013746 000004 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1836 002600 012737 002634 000004 MOV #64$,#ERRVEC ;;SET UP ERROR VECTOR
1837 002606 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
1838 002614 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1839 002622 022777 177777 176310 CMP #-1,$SWR ;;TRY TO REFERENCE HARDWARE SWR
1840 002630 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1841 ;;AND THE HARDWARE SWR IS NOT = -1
1842 002632 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
1843 002634 012716 002642 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
1844 002640 000002 RTI
1845 002642 012737 000176 001140 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
1846 002650 012737 000174 001142 MOV #DISPREG,DISPLAY
1847 002656 012637 000004 66$: MOV (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
1848
1849 002662 005037 001256 CLR $PASS ;;CLEAR PASS COUNT
    
```



```

1850 002666 132737 000200 001271 BITB #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
1851 002674 001403 BEQ 67$ ;;YES,USE NON-APT SWITCH
1852 002676 012737 001272 001140 MOV #$$SWREG,$SWR ;;NO,USE APT SWITCH REGISTER
1853 002704 67$:
1854 002704 005037 002314 CLR ERRCNT ;CLEAR ERROR COUNT FOR SWITCH 12 ABCRT
1855 .SBTTL TYPE PROGRAM NAME
1856 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1857 002710 005227 177777 INC #-1 ;;FIRST TIME?
1858 002714 001063 BNE 68$ ;;BRANCH IF NO
1859 002716 022737 025042 000042 CMP #SENDAD,2#42 ;;ACT-11?
1860 002724 001457 BEQ 68$ ;;BRANCH IF YES
1861 002726 104401 002774 TYPE ,69$ ;;TYPE ASCIZ STRING
1862 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1863 002732 005737 000042 TST 2#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
1864 002736 001012 BNE 70$ ;;BRANCH IF YES
1865 002740 123727 001270 000001 CMPB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
1866 002746 001406 BEQ 70$ ;;BRANCH IF YES
1867 002750 023727 001140 000176 CMP $SWR,$SWREG ;;SOFTWARE SWITCH REG SELECTED?
1868 002756 001005 BNE 71$ ;;BRANCH IF NO
1869 002760 104406 GTSWR ;;GET SOFT-SWR SETTINGS
1870 002762 000403 BR 71$
1871 002764 112737 000001 001134 70$: MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
1872 002772 71$:
1873 002772 000434 BR 68$ ;;GET OVER THE ASCIZ
1874 ;;69$: .ASCIZ <CRLF>RK611 DISKLESS DIAGNOSTIC: PART 5 MAINDEC-11-DZR6EA/<CRLF>
1875 003064 68$:
1876 003064 005227 177777 INC #-1 ;TEST IF PASS 0
1877 003070 001002 BNE 4$ ;NO - SKIP
1878 003072 104401 042612 TYPE ,OPR006 ;TYPE PASS TIME MESSAGE
1879 003076 022737 000001 002312 4$: CMP #1,$RTFLG ;CHECK IF PARAMETER START
1880 003104 001121 BNE 15$ ;NO, CONTINUE SETUP
1881 003106 104401 042500 5$: TYPE ,OPR001 ;TYPE "RK611 BUS ADDRESS ( ) ="
1882 003112 013746 001324 MOV $BASE,-(SP) ;;SAVE $BASE FOR TYPEOUT
1883 003116 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1884 003120 104401 042532 TYPE ,OPR002
1885 003124 104412 RDOCT ;GET VALUE
1886 003126 012637 001202 MOV (SP)+,$TMP0
1887 003132 001407 BEQ 7$ ;CHECK IF <CR>
1888 003134 022737 160000 001202 CMP #160000,$TMP0 ;CHECK IF IN I/O PAGE
1889 003142 101361 BHI 5$
1890 003144 013737 001202 001324 7$: MOV $TMP0,$BASE ;LOAD NEW BUS ADDRESS
1891 003152 104401 042540 TYPE ,OPR003 ;TYPE "RK611 VECTOR ADDRESS ( ) ="
1892 003156 013746 001320 MOV $VECT1,-(SP) ;TYPE OUT VECTOR ADDRESS
1893 003162 042716 160000 BIC #160000,(SP)
1894 003166 104401 042532 TYPE -,OPR002
1895 003172 104412 RDOCT ;GET VALUE
1896 003174 012637 001202 MOV (SP)+,$TMP0
1897 003200 001412 BEQ 10$ ;CHECK IF <CR>
1898 003202 022737 001000 001202 CMP #1000,$TMP0 ;CHECK IF LEGAL
1899 003210 101760 BLOS 7$
1900 003212 042737 017777 001320 BIC #17777,$VECT1 ;LOAD NEW VECTOR ADDRESS
1901 003220 053737 001202 001320 BIS $TMP0,$VECT1
1902 003226 104401 042570 10$: TYPE ,OPR004 ;TYPE "RK611 PRIORITY ( ) ="
1903 003232 005046 CLR -(SP)
1904 003234 113716 001321 MOVB $VECT1+1,(SP)
1905 003240 006216 ASR (SP) ;SHIFT 5 BITS RIGHT
    
```

E04

RK611 DISA LESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 43
 DZR6EA.P11 28-SEP-76 15:31 GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0042

1906	003242	006216			ASR	(SP)	
1907	003244	006216			ASR	(SP)	
1908	003246	006216			ASR	(SP)	
1909	003250	006216			ASR	(SP)	
1910	003252	104402			TYPOC		
1911	003254	104401	042532		TYPE	,OPR002	
1912	003260	104412			RDOCT		:GET VALUE
1913	003262	012637	001202		MOV	(SP)+,STMPD	
1914	003266	001430			BEQ	15\$:CHECK FOR DEFAULT
1915	003270	022737	000007	001202	CMP	#7,STMPD	:CHECK IF LEGAL
1916	003276	103753			BLO	10\$	
1917	003300	022737	000004	001202	CMP	#4,STMPD	
1918	003306	101347			BHI	10\$	
1919	003310	006337	001202		ASL	STMPD	:SHIFT 5 BITS LEFT
1920	003314	006337	001202		ASL	STMPD	
1921	003320	006337	001202		ASL	STMPD	
1922	003324	006337	001202		ASL	STMPD	
1923	003330	006337	001202		ASL	STMPD	
1924	003334	042737	160000	001320	BIC	#160000,\$VECT1	:STORE NEW PRIORITY
1925	003342	153737	001202	001321	BISB	STMPD,\$VECT1+1	
1926	003350	013737	001320	002304	MOV	\$VECT1,RKVEC	:STORE RK611 VECTOR
1927	003356	042737	160000	002304	BIC	#160000,RKVEC	
1928	003364	113737	001321	002306	MOVB	\$VECT1+1,RKPRI	:STORE PRIORITY
1929	003372	004737	034450		JSR	PC,\$SIZE	:SIZE MEMORY
1930							
1931	003376	013702	001324		NEWPAS: MOV	\$BASE,R2	:SET RK POINTER
1932	003402	005037	001236		CLR	\$ESCAPE	:CLEAR ESCAPE
1933	003406	004737	034314		JSR	PC,PARCHK	:CHECK OF MEMORY CHECK ENABLE
1934	003412	012746	000000		MOV	#PR0,-(SP)	:LOCK OUT INTERRUPTS
1935	003416	012746	003424		MOV	#TST1,-(SP)	
1936	003422	000002			RTI		

F04

.SBTTL **MULTI-SECTOR WRITE OPERATIONS

 TEST 1 MULTI-SECTOR WRITE (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
 CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
 AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
 AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE
 TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION
 CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR.
 PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION
 OF HEADER.

```

TEST1:  SCOPE
        MOV     #10.,$TIMES      ;;DO 10. ITERATIONS
        MOV     #0CLR,RKCS1(R2) ;CLR CONTROLLER
        CLR     BSEDES          ;CLEAR BAD SEC DESIRED
        JSR     R4,LOADRK       ;LOAD "L" REGISTERS
        0              ;CYLINDER 0
        .BYTE    0              ;SECTOR 0
        .BYTE    0              ;TRACK 0
        OBUFF     051444        ;BUFFER ADDRESS OBUFF
        -401      ;WORD COUNT -401
        WRDATA    ;COMMAND WRDATA
        JSR     R4,BLDDAT       ;GO BUILD DATA
        1              ;PATTERN 1
        JSR     R4,OPSTR       ;START THE OPERATION
        JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
        ERROR    4              ;CSI MISCOMPARE
        ERROR    5              ;MR1
        ERROR    6              ;MR2
        ERROR    7              ;MR3
        JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
        JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
        ERROR    10             ;MR1 CONTENTS IN ERROR
        ERROR    11             ;CSI CONTENTS IN ERROR
        JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
        ERROR    12             ;MR1 CONTENTS IN ERROR
        ERROR    13             ;CSI IN ERROR AFTER SEARCH
        ERROR    14             ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
        JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
    
```

1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952
 1953
 1954 003424 000004
 1955 003426 012737 000012 001234
 1956
 1957 003434 012762 100000 000000
 1958
 1959 003442 005037 002352
 1960
 1961 003446 004437 034026
 1962 003452 000000
 1963 003454 000
 1964 003455 000
 1965 003456 051444
 1966 003460 177377
 1967 003462 000023
 1968
 1969 003464 004437 033612
 1970 003470 000001
 1971
 1972 003472 004437 034072
 1973
 1974 003476 004437 025100
 1975 003502 104004
 1976 003504 104005
 1977 003506 104006
 1978 003510 104007
 1979
 1980 003512 004437 025062
 1981
 1982 003516 004437 025504
 1983 003522 104010
 1984 003524 104011
 1985
 1986 003526 004437 026042
 1987 003532 104012
 1988 003534 104013
 1989 003536 104014
 1990
 1991 003540 012737 045211 001510
 1992 003546 004437 027002

G04

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 45
 DZR6EA.P11 28-SEP-76 15:31 T1 MULTI-SECTOR WRITE (PART 1)

SEG 0045

1993	003552	104015			ERROR	15		;MR1 IN ERROR IN READ GAP
1994	003554	104016			ERROR	16		;CS1 IN ERROR AFTER READ GAP
1995	003556	104017			ERROR	17		;MR1 IN ERROR IN WRITING SYNC
1996	003560	104020			ERROR	20		;CS1 IN ERROR AFTER WRITING SYNC
1997								
1998	003562	012737	045270	001510	MOV	#EM5021,EMW		;SET MESSAGE FOR WRITE FAILURE
1999	003570	004437	027402		JSR	R4,DWRITE		;SIMULATE WRITE DATA AND ECC
2000	003574	104017			ERROR	17		;MR1 IN ERROR IN WRITING DATA OR ECC
2001	003576	104021			ERROR	21		;ECC IN ERROR WHILE WRITING DATA
2002	003600	104022			ERROR	22		;CS1 IN ERROR AFTER WRITING DATA AND ECC
2003								
2004	003602	012700	000034		MOV	#7*4,R0		;ISSUE CLOCKS TO NEXT HDR COMPARE
2005	003606	012762	000440	000026	15:	MOV	#DMD!MCLK,RKMR1(R2)	;CLOCK LOOP
2006	003614	012762	000040	000026	MOV	#DMD,RKMR1(R2)		
2007	003622	005300			DEC	R0		
2008	003624	001370			BNE	15		
2009								
2010	003626	013703	002274		MOV	L.DCYL,R3		;GET DESIRED CYLINDER
2011	003632	004437	034242		JSR	R4,FSBLVV		;INVERT WORD
2012	003636	010337	002246		MOV	R3,E.MR2		;STORE EXPECTED MR2
2013	003642	113703	002267		MOV	L.DT,R3		;GET DESIRED TRACK/SECTOR
2014	003646	042703	177400		BIC	#177400,R3		;CLEAR UNUSED BITS
2015	003652	012700	000005		MOV	#5,R0		;SET COUNT FOR SHIFTING
2016	003656	006303			55:	ASL	R3	;SHIFT TRACK FOR HEADER ALIGNMENT
2017	003660	005300			DEC	R0		;DEC COUNT
2018	003662	001375			BNE	55		;LOOP UNTIL ZERO
2019	003664	153703	002266		BISB	L.DS,R3		;INSERT SECTOR NUMBER
2020								
2021	003670	032737	010000	002260	BIT	#CFMT,L.CS1		;TEST IF 18 BIT MODE
2022	003676	001402			BEQ	65		;NO - SKIP
2023	003700	052703	001000		BIS	#BIT9,R3		;SET FORMAT BIT FOR WORD TWO
2024	003704				65:			
2025	003704	004437	034242		JSR	R4,FSBLVV		;INVERT WORD
2026	003710	010337	002250		MOV	R3,E.MR3		;STORE EXPECTED MR3
2027	003714	016237	000034	002206	MOV	RKMR2(R2),T.MR2		;GET MR2
2028	003722	016237	000036	002210	MOV	RKMR3(R2),T.MR3		;GET MR3
2029	003730	016237	000000	002160	MOV	RKCS1(R2),T.CS1		;GET CS1
2030	003736	013737	002260	002220	MOV	L.CS1,E.CS1		;GET EXPECTED CS1
2031	003744	023737	002160	002220	CMP	T.CS1,E.CS1		;CHECK IF OK
2032	003752	001402			BEQ	35		;YES - SKIP
2033	003754	104025			ERROR	25		;CS1 IN ERROR AT MULTI-SECTOR TIME
2034	003756	000427			BR	TST2		;GO TO NEXT TEST
2035								
2036	003760	023737	002206	002246	35:	CMP	T.MR2,E.MR2	;CHECK IF MR2 OK
2037	003766	001402			BEQ	25		;YES - SKIP
2038	003770	104023			ERROR	23		;MR2 IN ERROR AT MULTI-SECTOR TIME
2039	003772	000421			BR	TST2		;GO TO NEXT TEST
2040								
2041	003774	023737	002210	002250	25:	CMP	T.MR3,E.MR3	;CHECK IF MR3 OK
2042	004002	001402			BEQ	45		;YES - SKIP
2043	004004	104024			ERROR	24		;MR3 IN ERROR AT MULTI-SECTOR TIME
2044	004006	000413			BR	TST2		;GO TO NEXT TEST
2045								
2046	004010	004437	025062		45:	JSR	R4,DSECTR	;SIMULATE 2ND SECTOR PULSE
2047								
2048	004014	004437	025504		JSR	R4,DRSYNC		;SIMULATE PREAMBLE

H04

2049	004020	104010		ERROR	10		;MRI ERROR
2050	004022	104011		ERROR	11		;CSI ERROR
2051							
2052	004024	004437	026042	JSR	R4,DHDCMP		;SIMULATE HEADER SEARCH
2053	004030	104012		ERROR	12		;MRI ERROR
2054	004032	104013		ERROR	13		;CSI ERROR
2055	004034	104014		ERROR	14		;RKDCYL OR RKDA IN ERROR
2056							
2057							

*TEST 2 MULTI-SECTOR WRITE (PART 2)

* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 23. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
* TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION
* CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR.
* PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION
* OF HEADER.

2072	004036	000004		TEST2: SCOPE			
2073	004040	012737	000012 001234	MOV	#10.,\$TIMES		::DO 10. ITERATIONS
2074							
2075	004046	012762	100000 000000	MOV	#CCLR,RKCS1(R2)		;CLEAR CONTROLLER
2076							
2077	004054	005037	002352	CLR	BSEDES		;CLEAR BAD SEC DESIRED
2078							
2079	004060	004437	034026	JSR	R4,LOADRK		;LOAD "L" REGISTERS
2080	004064	000000		0			;CYLINDER 0
2081	004066	023		.BYTE	23		;SECTOR 23
2082	004067	000		.BYTE	0		;TRACK 0
2083	004070	051444		OBUFF			;BUFFER ADDRESS OBUFF
2084	004072	177377		-401			;WORD COUNT -401
2085	004074	000023		WRDATA			;COMMAND WRDATA
2086							
2087	004076	004437	033612	JSR	R4,BLDDAT		;GO BUILD DATA
2088	004102	000002		2			;PATTERN 2
2089							
2090	004104	004437	034072	JSR	R4,OPSTRT		;START THE OPERATION
2091							
2092	004110	004437	025100	JSR	R4,DISEEK		;SIMULATE IMPLIED SEEK
2093	004114	104004		ERROR	4		;CSI MISCOMPARE
2094	004116	104005		ERROR	5		;MRI "
2095	004120	104006		ERROR	6		;MR2 "
2096	004122	104007		ERROR	7		;MR3 "
2097							
2098	004124	004437	025062	JSR	R4,DSECTR		;SIMULATE SECTOR PULSE
2099							
2100	004130	004437	025504	JSR	R4,DRSYNC		;SIMULATE HEADER PREAMBLE
2101	004134	104010		ERROR	10		;MRI CONTENTS IN ERROR
2102	004136	104011		ERROR	11		;CSI CONTENTS IN ERROR
2103							
2104	004140	004437	026042	JSR	R4,DHDCMP		;SIMULATE HEADER SEARCH

2105	004144	104012				EPROR	12		;MRI CONTENTS IN ERROR
2106	004146	104013				ERROR	13		;CSI IN ERROR AFTER SEARCH
2107	004150	104014				ERROR	14		;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2108									
2109	004152	012737	045211	001510		MOV	#EM5020,EMW		;SET MESSAGE FOR WRITE FAILURE
2110	004160	004437	027002			JSR	R4,DWGPSN		;SIMULATE GAP AND SYNC FOR WRITE
2111	004164	104015				ERROR	15		;MRI IN ERROR IN READ GAP
2112	004166	104016				ERROR	16		;CSI IN ERROR AFTER READ GAP
2113	004170	104017				ERROR	17		;MRI IN ERROR IN WRITING SYNC
2114	004172	104020				ERROR	20		;CSI IN ERROR AFTER WRITING SYNC
2115									
2116	004174	012737	045270	001510		MOV	#EM5021,EMW		;SET MESSAGE FOR WRITE FAILURE
2117	004202	004437	027402			JSR	R4,DWRITE		;SIMULATE WRITE DATA AND ECC
2118	004206	104017				ERROR	17		;MRI IN ERROR IN WRITING DATA OR ECC
2119	004210	104021				ERROR	21		;ECC IN ERROR WHILE WRITING DATA
2120	004212	104022				ERROR	22		;CSI IN ERROR AFTER WRITING DATA AND ECC
2121									
2122	004214	012700	000034			MOV	#7*4,R0		;ISSUE CLOCKS TO NEXT HDR COMPARE
2123	004220	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)		;CLOCK LOOP
2124	004226	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
2125	004234	005300				DEC	R0		
2126	004236	001370				BNE	1\$		
2127									
2128	004240	013703	002274			MOV	L.DCYL,R3		;GET DESIRED CYLINDER
2129	004244	004437	C34242			JSR	R4,FSBLVV		;INVERT WORD
2130	004250	010337	002246			MOV	R3,E.MR2		;STORE EXPECTED MR2
2131	004254	113703	002267			MOVB	L.DT,R3		;GET DESIRED TRACK/SECTOR
2132	004260	042703	177400			BIC	#177400,R3		;CLEAR UNUSED BITS
2133	004264	012700	000005			MOV	#5,R0		;SET COUNT FOR SHIFTING
2134	004270	006303			5\$:	ASL	R3		;SHIFT TRACK FOR HEADER ALIGNMENT
2135	004272	005300				DEC	R0		;DEC COUNT
2136	004274	001375				BNE	5\$;LOOP UNTIL ZERO
2137	004276	153703	002266			BISB	L.DS,R3		;INSERT SECTOR NUMBER
2138									
2139	004302	032737	010000	002260		BIT	#CFMT,L.CS1		;TEST IF 18 BIT MODE
2140	004310	001402				BEQ	6\$;NO - SKIP
2141	004312	052703	001000			BIS	#BIT9,R3		;SET FORMAT BIT FOR WORD TWO
2142	004316				6\$:				
2143	004316	004437	034242			JSR	R4,FSBLVV		;INVERT WORD
2144	004322	010337	002250			MOV	R3,E.MR3		;STORE EXPECTED MR3
2145	004326	016237	000034	002206		MOV	RKMR2(R2),T.MR2		;GET MR2
2146	004334	016237	000036	002210		MOV	RKMR3(R2),T.MR3		;GET MR3
2147	004342	016237	000000	002160		MOV	RKCS1(R2),T.CS1		;GET CS1
2148	004350	013737	002260	002220		MOV	L.CS1,E.CS1		;GET EXPECTED CS1
2149	004356	023737	002160	002220		CMP	T.CS1,E.CS1		;CHECK IF OK
2150	004364	001402				BEQ	3\$;YES - SKIP
2151	004366	104025				ERROR	25		;CSI IN ERROR AT MULTI-SECTOR TIME
2152	004370	000427				BR	TST3		;GO TO NEXT TEST
2153									
2154	004372	023737	002206	002246	3\$:	CMP	T.MR2,E.MR2		;CHECK IF MR2 OK
2155	004400	001402				BEQ	2\$;YES - SKIP
2156	004402	104023				ERROR	23		;MR2 IN ERROR AT MULTI-SECTOR TIME
2157	004404	000421				BR	TST3		;GO TO NEXT TEST
2158									
2159	004406	023737	002210	002250	2\$:	CMP	T.MR3,E.MR3		;CHECK IF MR3 OK
2160	004414	001402				BEQ	4\$;YES - SKIP

J04

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 48
 DZR6EA.P11 28-SEP-76 15:31 T2 MULTI-SECTOR WRITE (PART 2)

SEG 0048

```

2161 004416 104024      ERROR 24      ;MR3 IN ERROR AT MULTI-SECTOR TIME
2162 004420 000413      BR      TST3      ;;GO TO NEXT TEST
2163
2164 004422 004437 025062 4S:  JSR      R4,DSECTR      ;SIMULATE 2ND SECTOR PULSE
2165
2166 004426 004437 025504      JSR      R4,DRSYNC      ;SIMULATE PREAMBLE
2167 004432 104010      ERROR 10      ;MR1 ERROR
2168 004434 104011      ERROR 11      ;CS1 ERROR
2169
2170 004436 004437 026042      JSR      R4,DHDCMP      ;SIMULATE HEADER SEARCH
2171 004442 104012      ERROR 12      ;MR1 ERROR
2172 004444 104013      ERROR 13      ;CS1 ERROR
2173 004446 104014      ERROR 14      ;RKDCYL OR RKDA IN ERROR
2174
2175 *****
2176 *TEST 3          MID-TRANSFER SEEK (PART 1)
2177 *
2178 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2179 * PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2180 * DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT.
2181 * CYLINDER 0, HEAD 0, SECTOR 25.  CLOCK THROUGH SEEK
2182 * AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2183 * AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
2184 * TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
2185 * AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2186 * MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2187 *
2188 *****
2189 *
2189 004450 000004      TST3: SCOPE
2190 004452 012737 000012 001234      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
2191
2192 004460 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2193
2194 004466 005037 002352      CLR      BSEDES      ;CLEAR BAD SEC DESIRED
2195
2196 004472 004437 034026      JSR      R4,LOADRK      ;LOAD "L" REGISTERS
2197 004476 000000      0          ;CYLINDER 0
2198 004500      025          .BYTE 25      ;SECTOR 25
2199 004501      000          .BYTE 0       ;TRACK 0
2200 004502 051444      OBUFF      ;BUFFER ADDRESS OBUFF
2201 004504 177377      -401      ;WORD COUNT -401
2202 004506 000023      WRDATA      ;COMMAND WRDATA
2203
2204 004510 004437 033612      JSR      R4,BLDDAT      ;GO BUILD DATA
2205 004514 000003      3          ;PATTERN 3
2206
2207 004516 004437 034072      JSR      R4,OPSTRT      ;START THE OPERATION
2208
2209 004522 004437 025100      JSR      R4,DISEEK      ;SIMULATE IMPLIED SEEK
2210 004526 104004      ERROR 4      ;CS1 MISCOMPARE
2211 004530 104005      ERROR 5      ;MR1 ""
2212 004532 104006      ERROR 6      ;MR2 ""
2213 004534 104007      ERROR 7      ;MR3 ""
2214
2215 004536 004437 025062      JSR      R4,DSECTR      ;SIMULATE SECTOR PULSE
2216
  
```


K04

2217	004542	004437	025504			JSR	R4,DRSYNC	;SIMULATE HEADER PREAMBLE
2218	004546	104010				ERROR	10	;MR1 CONTENTS IN ERROR
2219	004550	104011				ERROR	11	;CS1 CONTENTS IN ERROR
2220								
2221	004552	004437	026042			JSR	R4,DHDCMP	;SIMULATE HEADER SEARCH
2222	004556	104012				ERROR	12	;MR1 CONTENTS IN ERROR
2223	004560	104013				ERROR	13	;CS1 IN ERROR AFTER SEARCH
2224	004562	104014				ERROR	14	;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2225								
2226	004564	012737	045211	001510		MOV	#EM5020,EMW	;SET MESSAGE FOR WRITE FAILURE
2227	004572	004437	027002			JSR	R4,DWGPN	;SIMULATE GAP AND SYNC FOR WRITE
2228	004576	104015				ERROR	15	;MR1 IN ERROR IN READ GAP
2229	004600	104016				ERROR	16	;CS1 IN ERROR AFTER READ GAP
2230	004602	104017				ERROR	17	;MR1 IN ERROR IN WRITING SYNC
2231	004604	104020				ERROR	20	;CS1 IN ERROR AFTER WRITING SYNC
2232								
2233	004606	012737	045270	001510		MOV	#EM5021,EMW	;SET MESSAGE FOR WRITE FAILURE
2234	004614	004437	027402			JSR	R4,DWRITE	;SIMULATE WRITE DATA AND ECC
2235	004620	104017				ERROR	17	;MR1 IN ERROR IN WRITING DATA OR ECC
2236	004622	104021				ERROR	21	;ECC IN ERROR WHILE WRITING DATA
2237	004624	104022				ERROR	22	;CS1 IN ERROR AFTER WRITING DATA AND ECC
2238								
2239	004626	012700	000324			MOV	#53.*4,R0	;ISSUE CLOCKS TO NEXT HDR COMPARE
2240	004632	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	;CLOCK LOOP
2241	004640	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2242	004646	005300				DEC	R0	
2243	004650	001370				BNE	1\$	
2244								
2245	004652	013703	002274			MOV	L.DCYL,R3	;GET DESIRED CYLINDER
2246	004656	004437	034242			JSR	R4,FSBLVV	;INVERT WORD
2247	004662	010337	002246			MOV	R3,E.MR2	;STORE EXPECTED MR2
2248	004666	113703	002267			MOV	L.DT,R3	;GET DESIRED TRACK/SECTOR
2249	004672	042703	177400			BIC	#177400,R3	;CLEAR UNUSED BITS
2250	004676	012700	000005			MOV	#5,R0	;SET COUNT FOR SHIFTING
2251	004702	006303			5\$:	ASL	R3	;SHIFT TRACK FOR HEADER ALIGNMENT
2252	004704	005300				DEC	R0	;DEC COUNT
2253	004706	001375				BNE	5\$;LOOP UNTIL ZERO
2254	004710	153703	002266			BISB	L.DS,R3	;INSERT SECTOR NUMBER
2255								
2256	004714	032737	010000	002260		BIT	#CFMT,L.CS1	;TEST IF 18 BIT MODE
2257	004722	001402				BEQ	6\$;NO - SKIP
2258	004724	052703	001000			BIS	#BIT9,R3	;SET FORMAT BIT FOR WORD TWO
2259	004730				6\$:			
2260	004730	004437	034242			JSR	R4,FSBLVV	;INVERT WORD
2261	004734	010337	002250			MOV	R3,E.MR3	;STORE EXPECTED MR3
2262	004740	016237	000034	002206		MOV	RKMR2(R2),T.MR2	;GET MR2
2263	004746	016237	000036	002210		MOV	RKMR3(R2),T.MR3	;GET MR3
2264	004754	016237	000000	002160		MOV	RKCS1(R2),T.CS1	;GET CS1
2265	004762	013737	002260	002220		MOV	L.CS1,E.CS1	;GET EXPECTED CS1
2266	004770	023737	002160	002220		CMP	T.CS1,E.CS1	;CHECK IF OK
2267	004776	001402				BEQ	3\$;YES - SKIP
2268	005000	104025				ERROR	25	;CS1 IN ERROR AT MULTI-SECTOR TIME
2269	005002	000427				BR	TST4	;GO TO NEXT TEST
2270								
2271	005004	023737	002206	002246	3\$:	CMP	T.MR2,E.MR2	;CHECK IF MR2 OK
2272	005012	001402				BEQ	2\$;YES - SKIP

L04

```

2273 005014 104023          ERROR 23          ;MR2 IN ERROR AT MULTI-SECTOR TIME
2274 005016 000421          BR      TST4          ;;GO TO NEXT TEST
2275
2276 005020 023737 002210 002250 2$:  CMP      T.MR3,E.MR3  ;CHECK IF MR3 OK
2277 005026 001402          BEQ      4$          ;YES - SKIP
2278 005030 104024          ERROR 24          ;MR3 IN ERROR AT MULTI-SECTOR TIME
2279 005032 000413          BR      TST4          ;;GO TO NEXT TEST
2280
2281 005034 004437 025062          4$:  JSR      R4,DSECTR  ;SIMULATE 2ND SECTOR PULSE
2282
2283 005040 004437 025504          JSR      R4,DRSYNC  ;SIMULATE PREAMBLE
2284 005044 104010          ERROR 10          ;MR1 ERROR
2285 005046 104011          ERROR 11          ;CS1 ERROR
2286
2287 005050 004437 026042          JSR      R4,DHDCMP  ;SIMULATE HEADER SEARCH
2288 005054 104012          ERROR 12          ;MR1 ERROR
2289 005056 104013          ERROR 13          ;CS1 ERROR
2290 005060 104014          ERROR 14          ;RKDCYL OR RKDA IN ERROR
2291
2292          ;*****
2293          ;*TEST 4          MID-TRANSFER SEEK (PART 2)
2294          ;*
2295          ;*          CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2296          ;*          PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2297          ;*          DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2298          ;*          CYLINDER 0, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
2299          ;*          AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2300          ;*          AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
2301          ;*          TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
2302          ;*          AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2303          ;*          MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2304          ;*
2305          ;*****
2306 005062 000004          TST4: SCOPE
2307 005064 012737 000012 001234          MO'      #10.,$TIMES  ;,DO 10. ITERATIONS
2308
2309 005072 012762 100000 000000          MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2310
2311 005100 005037 002352          CLR      BSEDES          ;CLEAR BAD SEC DESIRED
2312
2313 005104 004437 034026          JSR      R4,LOADRK  ;LOAD "L" REGISTERS
2314 005110 000000          0          ;CYLINDER 0
2315 005112          025          .BYTE 25          ;SECTOR 25
2316 005113          002          .BYTE 2          ;TRACK 2
2317 005114 051444          OBUFF          ;BUFFER ADDRESS OBUFF
2318 005116 177377          -401          ;WORD COUNT -401
2319 005120 000023          WRDATA          ;COMMAND WRDATA
2320
2321 005122 004437 033612          JSR      R4,BLDDAT  ;GO BUILD DATA
2322 005126 000004          4          ;PATTERN 4
2323
2324 005130 004437 034072          JSR      R4,OPSTRT  ;START THE OPERATION
2325
2326 005134 004437 025100          JSR      R4,DISEEK  ;SIMULATE IMPLIED SEEK
2327 005140 104004          ERROR 4          ;CS1 MISCOMPARE
2328 005142 104005          ERROR 5          ;MR1

```

M04

2329	005144	104006				ERROR	6		;MR2	"
2330	005146	104007				ERROR	7		;MR3	"
2331										
2332	005150	004437	025062			JSR	R4,DSECTR		;SIMULATE SECTOR PULSE	
2333										
2334	005154	004437	025504			JSR	R4,DRSYNC		;SIMULATE HEADER PREAMBLE	
2335	005160	104010				ERROR	10		;MR1 CONTENTS IN ERROR	
2336	005162	104011				ERROR	11		;CS1 CONTENTS IN ERROR	
2337										
2338	005164	004437	026042			JSR	R4,DHDCMP		;SIMULATE HEADER SEARCH	
2339	005170	104012				ERROR	12		;MR1 CONTENTS IN ERROR	
2340	005172	104013				ERROR	13		;CS1 IN ERROR AFTER SEARCH	
2341	005174	104014				ERROR	14		;RKDCYL OR RKDA IN ERROR AFTER SEARCH	
2342										
2343	005176	012737	045211	001510		MOV	#EM5020,EMW		;SET MESSAGE FOR WRITE FAILURE	
2344	005204	004437	027002			JSR	R4,DWGPSN		;SIMULATE GAP AND SYNC FOR WRITE	
2345	005210	104015				ERROR	15		;MR1 IN ERROR IN READ GAP	
2346	005212	104016				ERROR	16		;CS1 IN ERROR AFTER READ GAP	
2347	005214	104017				ERROR	17		;MR1 IN ERROR IN WRITING SYNC	
2348	005216	104020				ERROR	20		;CS1 IN ERROR AFTER WRITING SYNC	
2349										
2350	005220	012737	045270	001510		MOV	#EM5021,EMW		;SET MESSAGE FOR WRITE FAILURE	
2351	005226	004437	027402			JSR	R4,DWRITE		;SIMULATE WRITE DATA AND ECC	
2352	005232	104017				ERROR	17		;MR1 IN ERROR IN WRITING DATA OR ECC	
2353	005234	104021				ERROR	21		;ECC IN ERROR WHILE WRITING DATA	
2354	005236	104022				ERROR	22		;CS1 IN ERROR AFTER WRITING DATA AND ECC	
2355										
2356	005240	012700	000324			MOV	#53,#4,R0		;ISSUE CLOCKS TO NEXT HDR COMPARE	
2357	005244	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)		;CLOCK LOOP	
2358	005252	012762	000040	000026		MOV	#DMD,RKMR1(R2)			
2359	005260	005300				DEC	R0			
2360	005262	001370				BNE	1\$			
2361										
2362	005264	013703	002274			MOV	L.DCYL,R3		;GET DESIRED CYLINDER	
2363	005270	004437	034242			JSR	R4,FSBLVV		;INVERT WORD	
2364	005274	010337	002246			MOV	R3,E.MR2		;STORE EXPECTED MR2	
2365	005300	113703	002267			MOV	L.DT,R3		;GET DESIRED TRACK/SECTOR	
2366	005304	042703	177400			BIC	#177400,R3		;CLEAR UNUSED BITS	
2367	005310	012700	000005			MOV	#5,R0		;SET COUNT FOR SHIFTING	
2368	005314	006303			5\$:	ASL	R3		;SHIFT TRACK FOR HEADER ALIGNMENT	
2369	005316	005300				DEC	R0		;DEC COUNT	
2370	005320	001375				BNE	5\$;LOOP UNTIL ZERO	
2371	005322	153703	002266			BISB	L.DS,R3		;INSERT SECTOR NUMBER	
2372										
2373	005326	032737	010000	002260		BIT	#CFMT,L.CS1		;TEST IF 18 BIT MODE	
2374	005334	001402				BEQ	6\$;NO - SKIP	
2375	005336	052703	001000			BIS	#BIT9,R3		;SET FORMAT BIT FOR WORD TWO	
2376	005342				6\$:					
2377	005342	004437	034242			JSR	R4,FSBLVV		;INVERT WORD	
2378	005346	010337	002250			MOV	R3,E.MR3		;STORE EXPECTED MR3	
2379	005352	016237	000034	002206		MOV	RKMR2(R2),T.MR2		;GET MR2	
2380	005360	016237	000036	002210		MOV	RKMR3(R2),T.MR3		;GET MR3	
2381	005366	016237	000000	002160		MOV	RKCS1(R2),T.CS1		;GET CS1	
2382	005374	013737	002260	002220		MOV	L.CS1,E.CS1		;GET EXPECTED CS1	
2383	005402	023737	002160	002220		CMP	T.CS1,E.CS1		;CHECK IF OK	
2384	005410	001402				BEQ	3\$;YES - SKIP	

N04

```

2385 005412 104025      ERROR 25      ;CS1 IN ERROR AT MULTI-SECTOR TIME
2386 005414 000427      BR      TSTS      ;;GO TO NEXT TEST
2387
2388 005416 023737 002206 002246 3$:  CMP      T.MR2,E.MR2  ;CHECK IF MR2 OK
2389 005424 001402      BEQ      2$      ;YES - SKIP
2390 005426 104023      ERROR 23      ;MR2 IN ERROR AT MULTI-SECTOR TIME
2391 005430 000421      BR      TSTS      ;;GO TO NEXT TEST
2392
2393 005432 023737 002210 002250 2$:  CMP      T.MR3,E.MR3  ;CHECK IF MR3 OK
2394 005440 001402      BEQ      4$      ;YES - SKIP
2395 005442 104024      ERROR 24      ;MR3 IN ERROR AT MULTI-SECTOR TIME
2396 005444 000413      BR      TSTS      ;;GO TO NEXT TEST
2397
2398 005446 004437 025062      4$:  JSR      R4,DSECTR  ;SIMULATE 2ND SECTOR PULSE
2399
2400 005452 004437 025504      JSR      R4,DRSYNC  ;SIMULATE PREAMBLE
2401 005456 104010      ERROR 10      ;MR1 ERROR
2402 005460 104011      ERROR 11      ;CS1 ERROR
2403
2404 005462 004437 026042      JSR      R4,DHDCMP  ;SIMULATE HEADER SEARCH
2405 005466 104012      ERROR 12      ;MR1 ERROR
2406 005470 104013      ERROR 13      ;CS1 ERROR
2407 005472 104014      ERROR 14      ;RKDCYL OR RKDA IN ERROR
2408

```

```

*****
;TEST 5      MID-TRANSFER SEEK (PART 3)
*****
;
; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
; PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
; DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
; CYLINDER 0, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
; AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
; AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
; TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
; AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
; MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

```

```

*****
2423 005474 000004      TSTS:  SCOPE
2424 005476 012737 000012 001234      MOV      #10,$TIMES  ;;DO 10. ITERATIONS
2425
2426 005504 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2427
2428 005512 005037 002352      CLR      BSEDES      ;CLEAR BAD SEC DESIRED
2429
2430 005516 004437 034026      JSR      R4,LOADRK  ;LOAD "L" REGISTERS
2431 005522 000000      0      ;CYLINDER 0
2432 005524      025      .BYTE 25      ;SECTOR 25
2433 005525      002      .BYTE 2      ;TRACK 2
2434 005526 051444      OBUFF      ;BUFFER ADDRESS OBUFF
2435 005530 177377      -401      ;WORD COUNT -401
2436 005532 002023      WRDATA!CDT      ;COMMAND WRDATA!CDT
2437
2438 005534 004437 033612      JSR      R4,BLDDAT  ;GO BUILD DATA
2439 005540 000005      5      ;PATTERN 5
2440

```

B05

2441	005542	004437	034072		JSR	R4,OPSTR		:START THE OPERATION
2442	005546	004437	025100		JSR	R4,DISEEK		:SIMULATE IMPLIED SEEK
2443	005552	104004			ERROR	4		:CSI MISCOMPARE
2444	005554	104005			ERROR	5		:MR1
2445	005556	104006			ERROR	6		:MR2
2446	005560	104007			ERROR	7		:MR3
2447								
2448	005562	004437	025062		JSR	R4,DSECTR		:SIMULATE SECTOR PULSE
2449								
2450	005566	004437	025504		JSR	R4,DRSYNC		:SIMULATE HEADER PREAMBLE
2451	005572	104010			ERROR	10		:MR1 CONTENTS IN ERROR
2452	005574	104011			ERROR	11		:CSI CONTENTS IN ERROR
2453								
2454	005576	004437	026042		JSR	R4,DHDCMP		:SIMULATE HEADER SEARCH
2455	005602	104012			ERROR	12		:MR1 CONTENTS IN ERROR
2456	005604	104013			ERROR	13		:CSI IN ERROR AFTER SEARCH
2457	005606	104014			ERROR	14		:RKDCYL OR RKDA IN ERROR AFTER SEARCH
2458								
2459	005610	012737	045211	001510	MOV	#EMSO20,EMW		:SET MESSAGE FOR WRITE FAILURE
2460	005616	004437	027002		JSR	R4,DWGPSN		:SIMULATE GAP AND SYNC FOR WRITE
2461	005622	104015			ERROR	15		:MR1 IN ERROR IN READ GAP
2462	005624	104016			ERROR	16		:CSI IN ERROR AFTER READ GAP
2463	005626	104017			ERROR	17		:MR1 IN ERROR IN WRITING SYNC
2464	005630	104020			ERROR	20		:CSI IN ERROR AFTER WRITING SYNC
2465								
2466	005632	012737	045270	001510	MOV	#EMSO21,EMW		:SET MESSAGE FOR WRITE FAILURE
2467	005640	004437	027402		JSR	R4,DWRITE		:SIMULATE WRITE DATA AND ECC
2468	005644	104017			ERROR	17		:MR1 IN ERROR IN WRITING DATA OR ECC
2469	005646	104021			ERROR	21		:ECC IN ERROR WHILE WRITING DATA
2470	005650	104022			ERROR	22		:CSI IN ERROR AFTER WRITING DATA AND ECC
2471								
2472								
2473	005652	012700	000324		MOV	#53,*4,R0		:ISSUE CLOCKS TO NEXT HDR COMPARE
2474	005656	012762	000440	000026	MOV	#DMD!MCLK,RKMR1(R2)		:CLOCK LOOP
2475	005664	012762	000040	000026	MOV	#DMD,RKMR1(R2)		
2476	005672	005300			DEC	R0		
2477	005674	001370			BNE	18		
2478								
2479	005676	013703	002274		MOV	L,DCYL,R3		:GET DESIRED CYLINDER
2480	005702	004437	034242		JSR	R4,FSBLVV		:INVERT WORD
2481	005706	010337	002246		MOV	R3,E,MR2		:STORE EXPECTED MR2
2482	005712	113703	002267		MOV	L,DT,R3		:GET DESIRED TRACK/SECTOR
2483	005716	042703	177400		BIC	#177400,R3		:CLEAR UNUSED BITS
2484	005722	012700	000005		MOV	#5,R0		:SET COUNT FOR SHIFTING
2485	005726	006303		55:	ASL	R3		:SHIFT TRACK FOR HEADER ALIGNMENT
2486	005730	005300			DEC	R0		:DEC COUNT
2487	005732	001375			BNE	55		:LOOP UNTIL ZERO
2488	005734	153703	002266		BISB	L,DS,R3		:INSERT SECTOR NUMBER
2489								
2490	005740	032737	010000	002260	BIT	#CFMT,L,CSI		:TEST IF 18 BIT MODE
2491	005746	001402			BEQ	65		:NO - SKIP
2492	005750	052703	001000		BIS	#BIT9,R3		:SET FORMAT BIT FOR WORD TWO
2493	005754			65:				
2494	005754	004437	034242		JSR	R4,FSBLVV		:INVERT WORD
2495	005760	010337	002250		MOV	R3,E,MR3		:STORE EXPECTED MR3
2496	005764	016237	000034	002206	MOV	RKMR2(R2),T,MR2		:GET MR2

C05

```

2500 005772 016237 000036 002210      MOV      RKMR3(R2),T.MR3 ;GET MR3
2501 006000 016237 000000 002160      MOV      RKCSI(R2),T.CSI ;GET CSI
2502 006006 013737 002260 002220      MOV      L.CSI,E.CSI ;GET EXPECTED CSI
2503 006014 023737 002160 002220      CMP      T.CSI,E.CSI ;CHECK IF OK
2504 006022 001402          BEQ      3$ ;YES - SKIP
2505 006024 104025          ERROR   25 ;CSI IN ERROR AT MULTI-SECTOR TIME
2506 006026 000427          BR      TST6 ;GO TO NEXT TEST

2507 006030 023737 002206 002246 3$:      CMP      T.MR2,E.MR2 ;CHECK IF MR2 OK
2508 006036 001402          BEQ      2$ ;YES - SKIP
2509 006040 104023          ERROR   23 ;MR2 IN ERROR AT MULTI-SECTOR TIME
2510 006042 000421          BR      TST6 ;GO TO NEXT TEST

2511 006044 023737 002210 002250 2$:      CMP      T.MR3,E.MR3 ;CHECK IF MR3 OK
2512 006052 001402          BEQ      4$ ;YES - SKIP
2513 006054 104024          ERROR   24 ;MR3 IN ERROR AT MULTI-SECTOR TIME
2514 006056 000413          BR      TST6 ;GO TO NEXT TEST

2515 006060 004437 025062          4$:      JSR      R4,DSECTR ;SIMULATE 2ND SECTOR PULSE

2516 006064 004437 025504          JSR      R4,DRSYNC ;SIMULATE PREAMBLE
2517 006070 104010          ERROR   10 ;MR1 ERROR
2518 006072 104011          ERROR   11 ;CSI ERROR

2519 006074 004437 026042          JSR      R4,DHDCMP ;SIMULATE HEADER SEARCH
2520 006100 104012          ERROR   12 ;MR1 ERROR
2521 006102 104013          ERROR   13 ;CSI ERROR
2522 006104 104014          ERROR   14 ;RKDCYL OR RKDA IN ERROR
  
```

 *TEST 6 MID-TRANSFER SEEK (PART 4)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
 CYLINDER 0, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK AND
 DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
 AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
 TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES
 IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
 MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

```

2540 006106 000004          TST6:   SCOPE
2541 006110 012737 000012 001234      MOV      #10.,$TIMES ;DO 10. ITERATIONS
2542 006116 012762 100000 000000      MOV      #CLR,RKCSI(R2) ;CLEAR CONTROLLER
2543 006124 005037 002352          CLR      BSEDES ;CLEAR BAD SEC DESIRED
2544 006130 004437 034026          JSR      R4,LOADRK ;LOAD "L" REGISTERS
2545 006134 000000          0 ;CYLINDER 0
2546 006136 025          .BYTE 25 ;SECTOR 25
2547 006137 004          .BYTE 4 ;TRACK 4
2548 006140 051444          OBUFF ;BUFFER ADDRESS OBUFF
2549 006142 177377          -401 ;WORD COUNT -401
  
```

```

2553 006144 002023          WRDATA!COT          ;COMMAND WRDATA!COT
2554
2555 006146 004437 033612  JSR      R4,BLDDAT    ;GO BUILD DATA
2556 006152 000006          6                ;PATTERN 6
2557
2558 006154 004437 034072  JSR      R4,OPSTRT    ;START THE OPERATION
2559
2560 006160 004437 025100  JSR      R4,DISEEK    ;SIMULATE IMPLIED SEEK
2561 006164 104004          4                ;CSI MISCOMPARE
2562 006166 104005          5                ;MR1
2563 006170 104006          6                ;MR2
2564 006172 104007          7                ;MR3
2565
2566 006174 004437 025062  JSR      R4,DSECTR    ;SIMULATE SECTOR PULSE
2567
2568 006200 004437 025504  JSR      R4,DRSYNC    ;SIMULATE HEADER PREAMBLE
2569 006204 104010          10               ;MR1 CONTENTS IN ERROR
2570 006206 104011          11               ;CSI CONTENTS IN ERROR
2571
2572 006210 004437 026042  JSR      R4,DHDCMP    ;SIMULATE HEADER SEARCH
2573 006214 104012          12               ;MR1 CONTENTS IN ERROR
2574 006216 104013          13               ;CSI IN ERROR AFTER SEARCH
2575 006220 104014          14               ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2576
2577 006222 012737 045211 001510  MOV      #EM5020,EMW   ;SET MESSAGE FOR WRITE FAILURE
2578 006230 004437 027002          JSR      R4,DWGPSN    ;SIMULATE GAP AND SYNC FOR WRITE
2579 006234 104015          15               ;MR1 IN ERROR IN READ GAP
2580 006236 104016          16               ;CSI IN ERROR AFTER READ GAP
2581 006240 104017          17               ;MR1 IN ERROR IN WRITING SYNC
2582 006242 104020          20               ;CSI IN ERROR AFTER WRITING SYNC
2583
2584 006244 012737 045270 001510  MOV      #EM5021,EMW   ;SET MESSAGE FOR WRITE FAILURE
2585 006252 004437 027402          JSR      R4,DWRITE    ;SIMULATE WRITE DATA AND ECC
2586 006256 104017          17               ;MR1 IN ERROR IN WRITING DATA OR ECC
2587 006260 104021          21               ;ECC IN ERROR WHILE WRITING DATA
2588 006262 104022          22               ;CSI IN ERROR AFTER WRITING DATA AND ECC
2589
2590 006264 012700 000324          MOV      #53,*4,R0     ;ISSUE CLOCKS TO NEXT HDR COMPARE
2591 006270 012762 000440 000026 15:  MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK LOOP
2592 006276 012762 000040 000026  MOV      #DMD,RKMR1(R2)
2593 006304 005300          DEC      R0
2594 006306 001370          BNE     15
2595
2596 006310 013703 002274          MOV      L,DCYL,R3    ;GET DESIRED CYLINDER
2597 006314 004437 034242          JSR      R4,FSBLVV    ;INVERT WORD
2598 006320 010337 002246          MOV      R3,E.MR2    ;STORE EXPECTED MR2
2599 006324 113703 002267          MOV      L,DT,R3     ;GET DESIRED TRACK/SECTOR
2600 006330 042703 177400          BIC     #177400,R3   ;CLEAR UNUSED BITS
2601 006334 012700 000005          MOV      #5,R0       ;SET COUNT FOR SHIFTING
2602 006340 006303          ASL     R3           ;SHIFT TRACK FOR HEADER ALIGNMENT
2603 006342 005300          DEC     R0           ;DEC COUNT
2604 006344 001375          BNE     55           ;LOOP UNTIL ZERO
2605 006346 153703 002266          BISB   L,DS,R3       ;INSERT SECTOR NUMBER
2606
2607 006352 032737 010000 002260  BIT     #CFMT,L.CS1   ;TEST IF 18 BIT MODE
2608 006360 001402          BEQ     65           ;NO - SKIP
    
```

```

2609 006362 052703 001000          BIS      #BIT9,R3          ;SET FORMAT BIT FOR WORD TWO
2610 006366                          65:
2611 006366 004437 034242          JSR      R4,FSBLVV        ;INVERT WORD
2612 006372 010337 002250          MOV      R3,E.MR3        ;STORE EXPECTED MR3
2613 006376 016237 000034 002206  MOV      RKMR2(R2),T.MR2  ;GET MR2
2614 006404 016237 000036 002210  MOV      RKMR3(R2),T.MR3  ;GET MR3
2615 006412 016237 000000 002160  MOV      RKCS1(R2),T.CS1  ;GET CS1
2616 006420 013737 002260 002220  MOV      L.CS1,E.CS1     ;GET EXPECTED CS1
2617 006426 023737 002160 002220  CMP      T.CS1,E.CS1     ;CHECK IF OK
2618 006434 001402                          BEQ      35              ;YES - SKIP
2619 006436 104025                          ERROR    25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2620 006440 000427                          BR       TST7            ;GO TO NEXT TEST
2621 006442 023737 002206 002246  35:    CMP      T.MR2,E.MR2     ;CHECK IF MR2 OK
2622 006450 001402                          BEQ      25              ;YES - SKIP
2623 006452 104023                          ERROR    23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2624 006454 000421                          BR       TST7            ;GO TO NEXT TEST
2625
2626
2627 006456 023737 002210 002250  25:    CMP      T.MR3,E.MR3     ;CHECK IF MR3 OK
2628 006464 001402                          BEQ      45              ;YES - SKIP
2629 006466 104024                          ERROR    24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2630 006470 000413                          BR       TST7            ;GO TO NEXT TEST
2631
2632 006472 004437 025062          45:    JSR      R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2633
2634 006476 004437 025504          JSR      R4,DRSYNC        ;SIMULATE PREAMBLE
2635 006502 104010                          ERROR    10              ;MR1 ERROR
2636 006504 104011                          ERROR    11              ;CS1 ERROR
2637
2638 006506 004437 026042          JSR      R4,DHDCMP        ;SIMULATE HEADER SEARCH
2639 006512 104012                          ERROR    12              ;MR1 ERROR
2640 006514 104013                          ERROR    13              ;CS1 ERROR
2641 006516 104014                          ERROR    14              ;RKDCYL OR RKDA IN ERROR
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657 006520 000004          TST7:  SCOPE
2658 006522 012737 000012 001234  MOV      #10.,$TIMES     ;;DO 10. ITERATIONS
2659
2660 006530 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2661
2662 006536 005037 002352          CLR      BSEDES          ;CLEAR BAD SEC DESIRED
2663 006542 012700 002000          MOV      #2000,R0        ;SET A COUNT TO DELAY FOR CLEAR
2664 006546 005300          205:   DEC      R0

```

```

*****
;TEST 7      MID-TRANSFER SEEK AND 24 SECTOR FORMAT
;
; CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
; PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
; DATA OF 401 WORDS TO AN RK06 IN 24 SECTOR FORMAT.
; CYLINDER 0, HEAD 0, SECTOR 23.  CLOCK THROUGH SEEK
; AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
; AND A GOOD HEADER.  CLOCK THROUGH 400 18 BIT WORDS
; AND THE 32 BIT ECC.  CHECK THAT IMPLIED SEEK IS
; ISSUE TO NEXT TRACK.  CHECK FOR PROPER HEADER RECOGNITION
; OCCURS.
*****

```


F05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZ6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 57
 DZ6EA.P11 28-SEP-76 15:31 T7 MID-TRANSFER SEEK AND 24 SECTOR FORMAT

SEG 0057

2665	006550	001376		BNE	20\$:LOOP UNTIL ZERO
2666							
2667	006552	004437	034026	JSR	R4,LOADRK		:LOAD "L" REGISTERS
2668	006556	000000		0			:CYLINDER 0
2669	006560	023		.BYTE	23		:SECTOR 23
2670	006561	000		.BYTE	0		:TRACK 0
2671	006562	051444		0BUFF			:BUFFER ADDRESS 0BUFF
2672	006564	177377		-401			:WORD COUNT -401
2673	006566	010023		WRDATA!CFMT			:COMMAND WRDATA!CFMT
2674							
2675	006570	004437	033612	JSR	R4,BLDDAT		:GO BUILD DATA
2676	006574	000007		7			:PATTERN 7
2677							
2678	006576	004437	034072	JSR	R4,OPSTR		:START THE OPERATION
2679							
2680	006602	004437	025100	JSR	R4,DISEEK		:SIMULATE IMPLIED SEEK
2681	006606	104004		ERROR	4		:CSI MISCOMPARE
2682	006610	104005		ERROR	5		:MR1 "
2683	006612	104006		ERROR	6		:MR2 "
2684	006614	104007		ERROR	7		:MR3 "
2685							
2686	006616	004437	025062	JSR	R4,0SECTR		:SIMULATE SECTOR PULSE
2687							
2688	006622	004437	025504	JSR	R4,DRSYNC		:SIMULATE HEADER PREAMBLE
2689	006626	104010		ERROR	10		:MR1 CONTENTS IN ERROR
2690	006630	104011		ERROR	11		:CSI CONTENTS IN ERROR
2691							
2692	006632	004437	026042	JSR	R4,DHDCMP		:SIMULATE HEADER SEARCH
2693	006636	104012		ERROR	12		:MR1 CONTENTS IN ERROR
2694	006640	104013		ERROR	13		:CSI IN ERROR AFTER SEARCH
2695	006642	104014		ERROR	14		:RKDCYL OR RKDA IN ERROR AFTER SEARCH
2696							
2697	006644	012737	045211	MOV	#EM5020,EMW		:SET MESSAGE FOR WRITE FAILURE
2698	006652	004437	027002	JSR	R4,DWGPSN		:SIMULATE GAP AND SYNC FOR WRITE
2699	006656	104015		ERROR	15		:MR1 IN ERROR IN READ GAP
2700	006660	104016		ERROR	16		:CSI IN ERROR AFTER READ GAP
2701	006662	104017		ERROR	17		:MR1 IN ERROR IN WRITING SYNC
2702	006664	104020		ERROR	20		:CSI IN ERROR AFTER WRITING SYNC
2703							
2704	006666	012737	045270	MOV	#EM5021,EMW		:SET MESSAGE FOR WRITE FAILURE
2705	006674	004437	027402	JSR	R4,DWRITE		:SIMULATE WRITE DATA AND ECC
2706	006700	104017		ERROR	17		:MR1 IN ERROR IN WRITING DATA OR ECC
2707	006702	104021		ERROR	21		:ECC IN ERROR WHILE WRITING DATA
2708	006704	104022		ERROR	22		:CSI IN ERROR AFTER WRITING DATA AND ECC
2709							
2710	006706	012700	000324	MOV	#53,*4,R0		:ISSUE CLOCKS TO NEXT HDR COMPARE
2711	006712	012762	000440	MOV	#DMD!MCLK,RKMR1(R2)		:CLOCK LOOP
2712	006720	012762	000040	MOV	#DMD,RKMR1(R2)		
2713	006726	005300		DEC	R0		
2714	006730	001370		BNE	1\$		
2715							
2716	006732	013703	002274	MOV	L.DCYL,R3		:GET DESIRED CYLINDER
2717	006736	004437	034242	JSR	R4,FSBLVV		:INVERT WORD
2718	006742	010337	002246	MOV	R3,E.MR2		:STORE EXPECTED MR2
2719	006746	113703	002267	MOV	L.DT,R3		:GET DESIRED TRACK/SECTOR
2720	006752	042703	177400	BIC	#177400,R3		:CLEAR UNUSED BITS

G05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 58
 DZR6EA.P11 28-SEP-76 15:31 T7 MID-TRANSFER SEEK AND 24 SECTOR FORMAT

SEQ 0058

```

2721 006756 012700 000005      MOV      #5,R0      ;SET COUNT FOR SHIFTING
2722 006762 006303      5$: ASL      R3      ;SHIFT TRACK FOR HEADER ALIGNMENT
2723 006764 005300      DEC      R0      ;DEC COUNT
2724 006766 001375      BNE     5$      ;LOOP UNTIL ZERO
2725 006770 153703 002266      BISB    L.DS,R3    ;INSERT SECTOR NUMBER
2726
2727 006774 032737 010000 002260      BIT     #CFMT,L.CS1 ;TEST IF 18 BIT MODE
2728 007002 001402      BEQ     6$      ;NO - SKIP
2729 007004 052703 001000      BIS     #BIT9,R3   ;SET FORMAT BIT FOR WORD TWO
2730
2731 007010 004437 034242      6$: JSR     R4,FSBLVV ;INVERT WORD
2732 007014 010337 002250      MOV     R3,E.MR3  ;STORE EXPECTED MR3
2733 007020 016237 000034 002206      MOV     RKMR2(R2),T.MR2 ;GET MR2
2734 007026 016237 000036 002210      MOV     RKMR3(R2),T.MR3 ;GET MR3
2735 007034 016237 000000 002160      MOV     RKCS1(R2),T.CS1 ;GET CS1
2736 007042 013737 002260 002220      MOV     L.CS1,E.CS1 ;GET EXPECTED CS1
2737 007050 023737 002160 002220      CMP     T.CS1,E.CS1 ;CHECK IF OK
2738 007056 001402      BEQ     3$      ;YES - SKIP
2739 007060 104025      ERROR   25      ;CS1 IN ERROR AT MULTI-SECTOR TIME
2740 007062 000427      BR     TST10    ;GO TO NEXT TEST
2741
2742 007064 023737 002206 002246 3$:  CMP     T.MR2,E.MR2 ;CHECK IF MR2 OK
2743 007072 001402      BEQ     2$      ;YES - SKIP
2744 007074 104023      ERROR   23      ;MR2 IN ERROR AT MULTI-SECTOR TIME
2745 007076 000421      BR     TST10    ;GO TO NEXT TEST
2746
2747 007100 023737 002210 002250 2$:  CMP     T.MR3,E.MR3 ;CHECK IF MR3 OK
2748 007106 001402      BEQ     4$      ;YES - SKIP
2749 007110 104024      ERROR   24      ;MR3 IN ERROR AT MULTI-SECTOR TIME
2750 007112 000413      BR     TST10    ;GO TO NEXT TEST
2751
2752 007114 004437 025062      4$:  JSR     R4,DSECTR ;SIMULATE 2ND SECTOR PULSE
2753
2754 007120 004437 025504      JSR     R4,DRSYNC ;SIMULATE PREAMBLE
2755 007124 104010      ERROR   10      ;MR1 ERROR
2756 007126 104011      ERROR   11      ;CS1 ERROR
2757
2758 007130 004437 026042      JSR     R4,DHDCMP ;SIMULATE HEADER SEARCH
2759 007134 104012      ERROR   12      ;MR1 ERROR
2760 007136 104013      ERROR   13      ;CS1 ERROR
2761 007140 104014      ERROR   14      ;RKDCYL OR RKDA IN ERROR
2762

```

```

*****
*TEST 10      CYLINDER OVERFLOW (PART 1)
*
```

```

*
*   CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
*   PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
*   DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
*   CYLINDER 632, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
*   AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
*   AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
*   TWO WORD ECC.  MAKE SURE CYLINDER OVERFLOW ERROR DOES
*   NOT SET.
*
```

```

*****
*ST10:  SCOPE

```

2775 007142 000004

H05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 59
 DZR6EA.P11 28-SEP-76 15:31 T10 CYLINDER OVERFLOW (PART 1)

SEG 0059

```

2777 007144 012737 000012 001234      MOV      #10.,$TIMES      ;;DC 10. ITERATIONS
2778
2779 007152 005037 002352              CLR      BSEDES          ;CLEAR BAD SEC DESIRED
2780 007156 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2781
2782 007164 004437 034026      JSR      R4,LOADRK      ;LOAD "L" REGISTERS
2783 007170 000632 632              ;CYLINDER 632
2784 007172      025          .BYTE     25            ;SECTOR 25
2785 007173      002          .BYTE     2            ;TRACK 2
2786 007174 051444 0BUFF          ;BUFFER ADDRESS OBUFF
2787 007176 177400 -400          ;WORD COUNT -400
2788 007200 000023  WRDATA        ;COMMAND WRDATA
2789
2790 007202 004437 033612      JSR      R4,BLDDAT      ;GO BUILD DATA
2791 007206 000001 1              ;PATTERN 1
2792
2793 007210 004437 034072      JSR      R4,OPSTAT      ;START THE OPERATION
2794
2795 007214 004437 025100      JSR      R4,DISEEK      ;SIMULATE IMPLIED SEEK
2796 007220 104004 4              ERROR     4            ;CSI MISCOMPARE
2797 007222 104005 5              ERROR     5            ;MR1
2798 007224 104006 6              ERROR     6            ;MR2
2799 007226 104007 7              ERROR     7            ;MR3
2800
2801 007230 004437 025062      JSR      R4,DSECTR      ;SIMULATE SECTOR PULSE
2802
2803 007234 004437 025504      JSR      R4,DRSYNC      ;SIMULATE HEADER PREAMBLE
2804 007240 104010 10             ERROR     10           ;MR1 CONTENTS IN ERROR
2805 007242 104011 11             ERROR     11           ;CSI CONTENTS IN ERROR
2806
2807 007244 004437 026042      JSR      R4,DHDCMP      ;SIMULATE HEADER SEARCH
2808 007250 104012 12             ERROR     12           ;MR1 CONTENTS IN ERROR
2809 007252 104013 13             ERROR     13           ;CSI IN ERROR AFTER SEARCH
2810 007254 104014 14             ERROR     14           ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2811
2812 007256 012737 045211 001510      MOV      #EM5020,EMW    ;SET MESSAGE FOR WRITE FAILURE
2813 007264 004437 027002              JSR      R4,DWGPSN      ;SIMULATE GAP AND SYNC FOR WRITE
2814 007270 104015 15             ERROR     15           ;MR1 IN ERROR IN READ GAP
2815 007272 104016 16             ERROR     16           ;CSI IN ERROR AFTER READ GAP
2816 007274 104017 17             ERROR     17           ;MR1 IN ERROR IN WRITING SYNC
2817 007276 104020 20             ERROR     20           ;CSI IN ERROR AFTER WRITING SYNC
2818
2819 007300 012737 045270 001510      MOV      #EM5021,EMW    ;SET MESSAGE FOR WRITE FAILURE
2820 007306 004437 027402              JSR      R4,DWRITE      ;SIMULATE WRITE DATA AND ECC
2821 007312 104017 17             ERROR     17           ;MR1 IN ERROR IN WRITING DATA OR ECC
2822 007314 104021 21             ERROR     21           ;ECC IN ERROR WHILE WRITING DATA
2823 007316 104022 22             ERROR     22           ;CSI IN ERROR AFTER WRITING DATA AND ECC
2824
2825
2826 007320 012700 000016              MOV      #3*4+2,R0      ;SET COUNT TO END OPERATION
2827 007324 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2) ;SET CLOCK
2828 007332 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;CLEAR CLOCK
2829 007340 005300 0              DEC      R0             ;DEC COUNT
2830 007342 001370 1$              BNE     1$             ;LOOP UNTIL 0
2831
2832 007344 004437 034154      JSR      R4,GETREG      ;GET RK611 REGISTERS
  
```

```

2833 007350 032737 100000 002160 BIT #CERR,T.CS1 ;TEST IF ANY ERROR SET
2834 007356 001415 BEQ 3$ ;NO - SKIP TO EXIT
2835
2836 007360 032737 001000 002174 BIT #COE,T.ER ;TEST IF IT IS COE
2837 007366 001405 BEQ 2$ ;NO - SKIP
2838 007370 012737 046061 001622 MOV #E5026A,EH030 ;SET MESSAGE HEADER
2839 007376 104030 ERROR 30 ;"UNEXPECTED COE ERROR"
2840 007400 000404 BR 3$ ;EXIT
2841
2842 007402 012737 046061 001632 2$: MOV #E5026A,EH031 ;SET HEADER
2843 007410 104031 ERROR 31 ;"UNEXPECTED ERROR TESTING COE"
2844
2845 007412 3$:
2846
2847
2848 ;*****
2849 ;TEST 11 CYLINDER OVERFLOW (PART 2)
2850 ;
2851 ; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2852 ; PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
2853 ; DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT.
2854 ; CYLINDER 632, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK
2855 ; AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
2856 ; AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
2857 ; TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.
2858 ;*****
2859 007412 000004 ST11: SCOPE
2860 007414 012737 000012 001234 MOV #10,$TIMES ;;DO 10. ITERATIONS
2861
2862 007422 005037 002352 CLR BSEDES ;CLEAR BAD SEC DESIRED
2863 007426 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2864
2865 007434 004437 034026 JSR R4,LOADRK ;LOAD "L" REGISTERS
2866 007440 000632 632 ;CYLINDER 632
2867 007442 025 .BYTE 25 ;SECTOR 25
2868 007443 002 .BYTE 2 ;TRACK 2
2869 007444 051444 OBUFF ;BUFFER ADDRESS OBUFF
2870 007446 177377 -401 ;WORD COUNT -401
2871 007450 000023 WRDATA ;COMMAND WRDATA
2872
2873 007452 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
2874 007456 000002 2 ;PATTERN 2
2875
2876 007460 004437 034072 JSR R4,OPSTRT ;START THE OPERATION
2877
2878 007464 004437 025100 JSR R4,DISEEK ;SIMULATE IMPLIED SEEK
2879 007470 104004 ERROR 4 ;CS1 MISCOMPARE
2880 007472 104005 ERROR 5 ;MR1
2881 007474 104006 ERROR 6 ;MR2
2882 007476 104007 ERROR 7 ;MR3
2883
2884 007500 004437 025062 JSR R4,DSECTR ;SIMULATE SECTOR PULSE
2885
2886 007504 004437 025504 JSR R4,DRSYNC ;SIMULATE HEADER PREAMBLE
2887 007510 104010 ERROR 10 ;MR1 CONTENTS IN ERROR
2888 007512 104011 ERROR 11 ;CS1 CONTENTS IN ERROR

```

```

2889
2890 007514 004437 026042 JSR R4,DHDCMP ;SIMULATE HEADER SEARCH
2891 007520 104012 ERROR 12 ;MRI CONTENTS IN ERROR
2892 007522 104013 ERROR 13 ;CSI IN ERROR AFTER SEARCH
2893 007524 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2894
2895 007526 012737 045211 001510 MOV #EM5020,EMW ;SET MESSAGE FOR WRITE FAILURE
2896 007534 004437 027002 JSR R4,DWGPSN ;SIMULATE GAP AND SYNC FOR WRITE
2897 007540 104015 ERROR 15 ;MRI IN ERROR IN READ GAP
2898 007542 104016 ERROR 16 ;CSI IN ERROR AFTER READ GAP
2899 007544 104017 ERROR 17 ;MRI IN ERROR IN WRITING SYNC
2900 007546 104020 ERROR 20 ;CSI IN ERROR AFTER WRITING SYNC
2901
2902 007550 012737 045270 001510 MOV #EM5021,EMW ;SET MESSAGE FOR WRITE FAILURE
2903 007556 004437 027402 JSR R4,DWRITE ;SIMULATE WRITE DATA AND ECC
2904 007562 104017 ERROR 17 ;MRI IN ERROR IN WRITING DATA OR ECC
2905 007564 104021 ERROR 21 ;ECC IN ERROR WHILE WRITING DATA
2906 007566 104022 ERROR 22 ;CSI IN ERROR AFTER WRITING DATA AND ECC
2907
2908
2909 007570 012700 000016 MOV #3*4+2,R0 ;SET COUNT TO COMPLETE OPERATION
2910 007574 012762 000440 000026 15: MOV #DMD!MCLK,RKMRI(R2) ;RUN THE CLOCK
2911 007602 012762 000040 000026 MOV #DMD,RKMRI(R2)
2912 007610 005300 DEC R0 ;DEC COUNT
2913 007612 001370 BNE 15 ;LOOP UNTIL 0
2914
2915 007614 004437 034154 JSR R4,GETREG ;GET RK611 REGS
2916 007620 032737 001000 002174 BIT #COE,T.ER ;TEST IF COE SET
2917 007626 001012 BNE 35 ;YES - EXIT TEST
2918
2919 007630 032737 100000 002160 BIT #CERR,T.CSI ;TEST IF ANY OTHER ERROR SET
2920 007636 001402 BEQ 25 ;NO - SKIP
2921 007640 104027 ERROR 27 ;"NO COE WHEN EXPECTED"
2922 007642 000404 BR 35 ;GO TO EXIT
2923
2924 007644 012737 046061 001602 25: MOV #E5026A,EH026 ;SET HEADER
2925 007652 104026 ERROR 26 ;"UNEXPECTED ERROR WHEN FORCING COE"
2926
2927 007654 35:

```

```

2928
2929 *****
2930 *TEST 12 CYLINDER OVERFLOW (PART 3)
2931 *
2932 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2933 * PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
2934 * DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET
2935 * CYLINDER 632, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK
2936 * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
2937 * AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
2938 * TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW.
2939 * ERROR DOES NOT SET.
2940 *
2941 *****
2942 007654 000004 TST12: SCOPE
2943 007656 012737 000012 001234 MOV #10.,$TIMES ;;DO 10. ITERATIONS
2944

```

K05

```

2945 007664 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2946
2947 007672 005037 002352      CLR      BSEDES          ;CLEAR BAD SEC DESIRED
2948
2949 007676 004437 034026      JSR      R4,LOADRK      ;LOAD "L" REGISTERS
2950 007702 000632      632          ;CYLINDER 632
2951 007704      025          .BYTE 25          ;SECTOR 25
2952 007705      004          .BYTE 4           ;TRACK 4
2953 007706 051444      OBUFF        ;BUFFER ADDRESS OBUFF
2954 007710 177377      -401        ;WORD COUNT -401
2955 007712 002023      WRDATA!CDT  ;COMMAND WRDATA!CDT
2956
2957 007714 004437 033612      JSR      R4,BLDDAT      ;GO BUILD DATA
2958 007720 000003      3           ;PATTERN 3
2959
2960 007722 004437 034072      JSR      R4,OPSTRT      ;START THE OPERATION
2961
2962 007726 004437 025100      JSR      R4,DISEEK      ;SIMULATE IMPLIED SEEK
2963 007732 104004      ERROR 4       ;CS1 MISCOMPARE
2964 007734 104005      ERROR 5       ;MR1 "
2965 007736 104006      ERROR 6       ;MR2 "
2966 007740 104007      ERROR 7       ;MR3 "
2967
2968 007742 004437 025062      JSR      R4,DSECTR      ;SIMULATE SECTOR PULSE
2969
2970 007746 004437 025504      JSR      R4,DRSYNC      ;SIMULATE HEADER PREAMBLE
2971 007752 104010      ERROR 10      ;MR1 CONTENTS IN ERROR
2972 007754 104011      ERROR 11      ;CS1 CONTENTS IN ERROR
2973
2974 007756 004437 026042      JSR      R4,DHDCMP      ;SIMULATE HEADER SEARCH
2975 007762 104012      ERROR 12      ;MR1 CONTENTS IN ERROR
2976 007764 104013      ERROR 13      ;CS1 IN ERROR AFTER SEARCH
2977 007766 104014      ERROR 14      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2978
2979 007770 012737 045211 001510      MOV      #EM5020,EMW    ;SET MESSAGE FOR WRITE FAILURE
2980 007776 004437 027002      JSR      R4,DWGPSN      ;SIMULATE GAP AND SYNC FOR WRITE
2981 010002 104015      ERROR 15      ;MR1 IN ERROR IN READ GAP
2982 010004 104016      ERROR 16      ;CS1 IN ERROR AFTER READ GAP
2983 010006 104017      ERROR 17      ;MR1 IN ERROR IN WRITING SYNC
2984 010010 104020      ERROR 20      ;CS1 IN ERROR AFTER WRITING SYNC
2985
2986 010012 012737 045270 001510      MOV      #EM5021,EMW    ;SET MESSAGE FOR WRITE FAILURE
2987 010020 004437 027402      JSR      R4,DWRITE      ;SIMULATE WRITE DATA AND ECC
2988 010024 104017      ERROR 17      ;MR1 IN ERROR IN WRITING DATA OR ECC
2989 010026 104021      ERROR 21      ;ECC IN ERROR WHILE WRITING DATA
2990 010030 104022      ERROR 22      ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2991
2992
2993 010032 012700 000016      MOV      #3*4+2,R0      ;SET COUNT TO END OPERATION
2994 010036 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2) ;SET CLOCK
2995 010044 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;CLEAR CLOCK
2996 010052 005300      DEC      R0           ;DEC COUNT
2997 010054 001370      BNE     1$          ;LOOP UNTIL 0
2998
2999 010056 004437 034154      JSR      R4,GETREG      ;GET RK611 REGISTERS
3000 010062 032737 100000 002160      BIT      #CERR,T.CS1    ;TEST IF ANY ERROR SET
    
```

L05

```

3001 010070 001415      BEQ      3$      ;NO - SKIP TO EXIT
3002 010072 032737 001000 002174  BIT      #COE,T.ER ;CHECK IF COE
3003 010100 001405      BEQ      2$      ;NO - SKIP
3004
3005 010102 012737 046232 001622  MOV      #E50268,EH030 ;SET HEADER
3006 010110 104030      ERROR    30      ;"UNEXPECTED COE ERROR"
3007 010112 000404      BR       3$      ;EXIT
3008
3009 010114 012737 046232 001632 2$:  MOV      #E50268,EH031 ;SET HEADER
3010 010122 104031      ERROR    31      ;"UNEXPECTED ERROR WHILE TESTING COE"
3011
3012 010124      3$:
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026 010124 000004      *TEST 13      CYLINDER OVERFLOW (PART 4)
3027 010126 012737 000012 001234  *
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056

```

 *TEST 13 CYLINDER OVERFLOW (PART 4)
 *
 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
 * PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
 * DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET
 * CYLINDER 1456, HEAD 4, SECTOR 25. CLOCK THROUGH SEEK
 * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
 * AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND
 * TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.

 TST13: SCOPE
 MOV #10.,\$TIMES ;;DO 10. ITERATIONS
 CLR BSEDES ;CLEAR BAD SEC DESIRED
 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
 JSR R4,LOADRK ;LOAD "L" REGISTERS
 1456 ;CYLINDER 1456
 .BYTE 25 ;SECTOR 25
 .BYTE 4 ;TRACK 4
 OBUFF ;BUFFER ADDRESS OBUFF
 -401 ;WORD COUNT -401
 WRDATA!CDT ;COMMAND WRDATA!CDT
 JSR R4,BLDDAT ;GO BUILD DATA
 4 ;PATTERN 4
 JSR R4,OPSTRT ;START THE OPERATION
 JSR R4,DISEEK ;SIMULATE IMPLIED SEEK
 ERROR 4 ;CS1 MISCOMPARE
 ERROR 5 ;MR1
 ERROR 6 ;MR2
 ERROR 7 ;MR3
 JSR R4,DSECTR ;SIMULATE SECTOR PULSE
 JSR R4,DRSYNC ;SIMULATE HEADER PREAMBLE
 ERROR 10 ;MR1 CONTENTS IN ERROR
 ERROR 11 ;CS1 CONTENTS IN ERROR

M05

```

3057 010226 004437 026042 JSR R4,DHDCMP ;SIMULATE HEADER SEARCH
3058 010232 104012 ERROR 12 ;MRI CONTENTS IN ERROR
3059 010234 104013 ERROR 13 ;CSI IN ERROR AFTER SEARCH
3060 010236 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3061
3062 010240 012737 045211 001510 MOV #EM5020,EMW ;SET MESSAGE FOR WRITE FAILURE
3063 010246 004437 027002 JSR R4,DWGPSN ;SIMULATE GAP AND SYNC FOR WRITE
3064 010252 104015 ERROR 15 ;MRI IN ERROR IN READ GAP
3065 010254 104016 ERROR 16 ;CSI IN ERROR AFTER READ GAP
3066 010256 104017 ERROR 17 ;MRI IN ERROR IN WRITING SYNC
3067 010260 104020 ERROR 20 ;CSI IN ERROR AFTER WRITING SYNC
3068
3069 010262 012737 045270 001510 MOV #EM5021,EMW ;SET MESSAGE FOR WRITE FAILURE
3070 010270 004437 027402 JSR R4,DWRITE ;SIMULATE WRITE DATA AND ECC
3071 010274 104017 ERROR 17 ;MRI IN ERROR IN WRITING DATA OR ECC
3072 010276 104021 ERROR 21 ;ECC IN ERROR WHILE WRITING DATA
3073 010300 104022 ERROR 22 ;CSI IN ERROR AFTER WRITING DATA AND ECC
3074
3075
3076 010302 012700 000016 MOV #3*4+2,R0 ;SET CCUNT TO END OPERATION
3077 010306 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ;SET CLOCK
3078 010314 012762 000040 000026 MOV #DMD,RKMR1(R2) ;CLEAR CLOCK
3079 010322 005300 DEC R0 ;DEC COUNT
3080 010324 001370 BNE 1$ ;LOOP UNTIL 0
3081
3082 010326 004437 034154 JSR R4,GETREG ;GET RK611 REGISTERS
3083 010332 032737 001000 002174 BIT #COE,T.ER ;TEST IF COE SET
3084 010340 001012 BNE 3$ ;YES - EXIT
3085
3086 010342 032737 100000 002160 BIT #CERR,T.CS1 ;TEST IF ANY OTHER ERROR SET
3087 010350 001402 BEQ 2$ ;NO SKIP
3088 010352 104027 ERROR 27 ;"UNEXPECTED ERROR WHILE TESTING COE"
3089 010354 000404 BR 3$ ;EXIT
3090
3091 010356 012737 046232 001602 2$: MOV #E5026B,EH026 ;SET HEADER
3092 010364 104026 ERROR 26 ;"COE DID NOT SET WHEN EXPECTED"
3093 010366 3$:
3094
3095 .SBTTL **NPR WRITING OF MEMORY
3096
3097 ;*****
3098 ;*TEST 14 NPR WRITING OF MEMORY
3099 ;*
3100 ;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3101 ;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
3102 ;* OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3103 ;* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
3104 ;* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
3105 ;* A GOOD HEADER, 40 DATA WORDS. VERIFY THAT THE WORDS
3106 ;* ARE WRITTEN PROPERLY IN MEMORY.
3107 ;*
3108 ;*****
3109 010366 000004 ST14: SCOPE
3110 010370 012737 000012 001234 MOV #10.,$TIMES ;;DO 10. ITERATIONS
3111
3112 010376 005037 002352 CLR BSEDES ;CLEAR BAD SEC DESIRED

```


N05

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 65
 DZR6EA.P11 28-SEP-76 15:31 T14 NPR WRITING OF MEMORY

SEQ 0065

```

3113 010402 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3114
3115 010410 005037 002344      CLR      ECCSRC          ;CLEAR ECC SOURCE
3116 010414 005037 002354      CLR      HIBITS         ;CLEAR HI ORDER BITS
3117 010420 004437 034026      JSR      R4,LOADRK      ;LOAD "L" REGISTERS
3118 010424 000000 0          0          ;CYLINDER 0
3119 010426 000      .BYTE 0          ;SECTOR 0
3120 010427 000      .BYTE 0          ;TRACK 0
3121 010430 050440  IBUFF          ;BUFFER ADDRESS IBUFF
3122 010432 177400  -400         ;WORD COUNT -400
3123 010434 000021  RDDATA       ;COMMAND RDDATA
3124
3125 010436 004437 033566      JSR      R4,CLRIBF      ;GO CLEAR INPUT BUFFER
3126
3127 010442 004437 033612      JSR      R4,BLDDAT     ;GO BUILD DATA
3128 010446 000003  3          ;PATTERN 3
3129
3130
3131 010450 004437 034072      JSR      R4,OPSTRT     ;START THE OPERATION
3132 010454 004437 025100      JSR      R4,DISEEK     ;SIMULATE IMPLIED SEEK
3133 010460 104004  4          ERROR        ;CSI MISCOMPARE
3134 010462 104005  5          ERROR        ;MR1
3135 010464 104006  6          ERROR        ;MR2
3136 010466 104007  7          ERROR        ;MR3
3137
3138 010470 004437 025062      JSR      R4,DSECTR     ;SIMULATE SECTOR PULSE
3139
3140 010474 004437 025504      JSR      R4,DRSYNC     ;SIMULATE HEADER PREAMBLE
3141 010500 104010  10         ERROR        ;MR1 CONTENTS IN ERROR
3142 010502 104011  11         ERROR        ;CSI CONTENTS IN ERROR
3143
3144 010504 004437 026042      JSR      R4,DHDCMP     ;SIMULATE HEADER SEARCH
3145 010510 104012  12         ERROR        ;MR1 CONTENTS IN ERROR
3146 010512 104013  13         ERROR        ;CSI IN ERROR AFTER SEARCH
3147 010514 104014  14         ERROR        ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3148
3149 010516 004437 030276      JSR      R4,DRGPSN     ;GO READ GAP AND SYNC
3150 010522 104015  15         ERROR        ;MR1 IN ERROR READING GAP
3151 010524 104016  16         ERROR        ;CSI IN ERROR AFTER READING GAP
3152 010526 104032  32         ERROR        ;MR1 IN ERROR READING SYNC
3153 010530 104033  33         ERROR        ;CSI IN ERROR AFTER READING SYNC
3154
3155 010532 012737 000000 002346      MOV      #0,ECPATX     ;LOAD EXPECTED PATTERN
3156 010540 012737 004066 002350      MOV      #4066,ECPOSX  ;LOAD EXPECTED POSITION
3157 010546 004437 030774      JSR      R4,DREAD      ;GO SIMULATE DATA READ
3158 010552 000060  60         ;NUMBER OF WORDS TO SIMULATE
3159 010554 104034  34         ERROR        ;MR1 IN ERROR READING DATA OR ECC
3160 010556 104035  35         ERROR        ;ECC ERROR READING DATA
3161 010560 104036  36         ERROR        ;CSI ERROR AFTER READING DATA OR ECC
3162 010562 104041  41         ERROR        ;ECC REG INCORRECT AFTER ECC READ
3163 010564 104042  42         ERROR        ;ERR IN ECC PAT CALC.
3164 010566 104043  43         ERROR        ;ERR IN ECC POS COUNT
3165
3166
3167 010570 012700 051444      MOV      #OBUFF,R0     ;SET POINTER TO GOOD DATA
3168 010574 012701 050440      MOV      #IBUFF,R1     ;SET POINTER TO INPUT DATA
  
```

```

3169
3170 010600 012704 000012      MOV      #10,R4      ;SET ERROR LIMIT COUNT
3171 010604 005037 001236      CLR      $ESCAPE    ;CLEAR ESCAPE
3172 010610 005037 001220      CLR      $TMP7      ;CLEAR ERROR REPORT SWITCH
3173 010614 012703 00004C      MOV      #40,R3     ;SET COMPARE LIMIT
3174 010620 005005                CLR      R5         ;CLEAR WORD COUNTER
3175
3176 010622 021011      15:     CMP      (R0),(R1)  ;COMPARE DATA
3177 010624 001005                BNE     25         ;SKIP IF NOT EQUAL
3178 010626 005303      45:     DEC      R3      ;ELSE DEC WORD COUNT LIMIT
3179 010630 001424                BEQ     55         ;IF 0 - SKIP TO EXIT
3180 010632 005205                INC     R5         ;BUMP WORD COUNT
3181 010634 022021      CMP     (R0)+,(R1)+ ;BUMP DATA POINTERS
3182 010636 000771                BR     15         ;LOOP
3183
3184 010640 005304      25:     DEC     R4      ;DEC ERROR LIMIT
3185 010642 001417                BEQ     55         ;IF 0 - SKIP
3186
3187 010644 011037 001162      MOV     (R0),$REG0  ;GET GOOD WORD FOR REPORT
3188 010650 011137 001164      MOV     (R1),$REG1  ;GET BAD WORD
3189 010654 010537 001166      MOV     R5,$REG2   ;GET WORD NUMBER
3190
3191 010660 005737 001220      TST     $TMP7      ;DECIDE WHICH ERROR REPORT TO USE
3192 010664 001004                BNE     35         ;SKIP IF NOT 0
3193 010666 005237 001220      INC     $TMP7      ;ELSE BUMP $TMP7
3194 010672 104037      ERROR  37         ;REPORT FIRST ERROR LINE
3195 010674 000754                BR     45         ;SKIP
3196
3197 010676 104040      35:     ERROR  40     ;REPORT ALL OTHER LINES
3198 010700 000752                BR     45         ;SKIP
3199
3200 010702      55:
3201
3202 .SBTTL **ECC ERROR DETECTION/CORRECTION TESTS
3203
3204 :*****
3205 :*TEST 15      READ DATA WITH NO ERROR
3206 :*
3207 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3208 :* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
3209 :* OF 400 WORDS TO AN RALB IN 26 SECTOR FORMAT.
3210 :* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
3211 :* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE.
3212 :* A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
3213 :* VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.
3214 :*****
3215
3216 010702 000004      15:     SCOPE
3217 010704 012737 000001 001234      MOV     #1,$TIMES  ;:DO 1 ITERATION
3218
3219 010712 005037 002352      CLR     BSEDES     ;CLEAR BAD SEC DESIRED
3220 010716 012762 100000 000000      MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3221 010724 005037 002344      CLR     ECCSRC     ;CLEAR ECC SOURCE FLAG
3222 010730 005037 002344      CLR     ECCSRC     ;CLEAR ECC SOURCE
3223 010734 005037 002354      CLR     HIBITS     ;CLEAR HI ORDER BITS
3224 010740 004437 034026      JSR     R4,LOADRK  ;LOAD "L" REGISTERS

```

```

3225 010744 000000 0 ;CYLINDER 0
3226 010746 000 .BYTE 0 ;SECTOR 0
3227 010747 000 .BYTE 0 ;TRACK 0
3228 010750 050440 Ibuff ;BUFFER ADDRESS Ibuff
3229 010752 177400 -400 ;WORD COUNT -400
3230 010754 000021 RDATA ;COMMAND RDATA
3231
3232 010756 004437 033566 JSR R4,CLRIBF ;GO CLEAR INPUT BUFFER
3233
3234 010762 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
3235 010766 003004 4 ;PATTERN 4
3236
3237
3238 010770 004437 034072 JSR R4,OPSTRT ;START THE OPERATION
3239 010774 004437 025100 JSR R4,DISEEK ;SIMULATE IMPLIED SEEK
3240 011000 104004 ERROR 4 ;CSI MISCOMPARE
3241 011002 104005 ERROR 5 ;MR1
3242 011004 104006 ERROR 6 ;MR2
3243 011006 104007 ERROR 7 ;MR3
3244
3245 011010 004437 025062 JSR R4,DSECTR ;SIMULATE SECTOR PULSE
3246
3247 011014 004437 025504 JSR R4,DRSYNC ;SIMULATE HEADER PREAMBLE
3248 011020 104010 ERROR 10 ;MR1 CONTENTS IN ERROR
3249 011022 104011 ERROR 11 ;CSI CONTENTS IN ERROR
3250
3251 011024 004437 026042 JSR R4,DHDCMP ;SIMULATE HEADER SEARCH
3252 011030 104012 ERROR 12 ;MR1 CONTENTS IN ERROR
3253 011032 104013 ERROR 13 ;CSI IN ERROR AFTER SEARCH
3254 011034 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3255
3256 011036 004437 030276 JSR R4,DRGPSN ;GO READ GAP AND SYNC
3257 011042 104015 ERROR 15 ;MR1 IN ERROR READING GAP
3258 011044 104016 ERROR 16 ;CSI IN ERROR AFTER READING GAP
3259 011046 104032 ERROR 32 ;MR1 IN ERROR READING SYNC
3260 011050 104033 ERROR 33 ;CSI IN ERROR AFTER READING SYNC
3261
3262 011052 012737 000000 002346 MOV #0,ECPATX ;LOAD EXPECTED PATTERN
3263 011060 012737 004066 002350 MOV #4066,ECPOSX ;LOAD EXPECTED POSITION
3264 011066 004437 030774 JSR R4,DRREAD ;GO SIMULATE DATA READ
3265 011072 000400 400 ;NUMBER OF WORDS TO SIMULATE
3266 011074 104034 ERROR 34 ;MR1 IN ERROR READING DATA OR ECC
3267 011076 104035 ERROR 35 ;ECC ERROR READING DATA
3268 011100 104036 ERROR 36 ;CSI ERROR AFTER READING DATA OR ECC
3269 011102 104041 ERROR 41 ;ECC REG INCORRECT AFTER ECC READ
3270 011104 104042 ERROR 42 ;ERR IN ECC PAT CALC.
3271 011106 104043 ERROR 43 ;ERR IN ECC POS COUNT
3272
3273
3274 011110 012700 002710 MOV #40.*37.,R0 ;SET COUNT TO EMPTY SILO
3275
3276 011114 012762 000440 000026 15: MOV #DMD!MCLK,RKMR1,R2 ;CLOCK CONTROLLER
3277 011122 012762 000040 000026 MOV #DMD,RKMR1(R2)
3278 011130 005300 DEC R0 ;DEC COUNT
3279 011132 001370 BNE 15 ;LOOP UNTIL 0
3280

```

3281	011134	016237	000000	002160	MOV	RKCS1,R2),T.CS1	:GET CS1
3282	011142	004437	034154		JSR	R4,GETREG	:GET 611 REGS
3283	011146	042737	000001	002220	BIC	#00,E.CS1	:CLEAR GO BIT
3284	011154	052737	000200	002220	BIS	#00,E.CS1	:SET READY BIT
3285	011162	023737	002160	002220	CMF	*.CS1,E.CS1	:CHECK IF CS1 CORRECT
3286	011170	001401			BEG	25	:YES - SKIP
3287	011172	104056			ERROR	56	:*CS1 IN ERROR AFTER READ DATA*
3288	011174						

```

25:
*****
*TEST 16      READ DATA WITH ONE BIT BURST ERROR (PART 1)
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
* OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
* A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
* A ONE BIT BURST ERROR OF A 0 THAT SHOULD BE A 1.
* VERIFY THE COUNTING OF THE POSITION
* REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
* MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
*****

```

3304	011174	000004			JS*16:	SCOPE	
3305	011176	012737	000001	001234	MOV	#1,5TIMES	::DC 1 ITERATION
3306							
3307	011204	005037	002352		CLR	BSEDES	:CLEAR BAD SEC DESIRED
3308	011210	012762	100000	000000	MOV	#0CLR,RKCS1,R2)	:CLEAR CONTROLLER
3309							
3310	011216	005037	002354		CLR	HIBITS	:CLEAR HI ORDER BITS
3311	011222	004437	034026		JSR	R4,LOADRK	:LOAD "L" REGISTERS
3312	011226	000000			D		:CYLINDER 0
3313	011230	000			.BYTE	0	:SECTOR 0
3314	011231	000			.BYTE	0	:TRACK 0
3315	011232	050440			IBLFF		:BUFFER ADDRESS IBLFF
3316	011234	177400			-400		:WORD COUNT -400
3317	011236	000021			RCDATA		:COMMAND RCDATA
3318							
3319	011240	004437	033566		JSR	R4,CLRIBF	:GO CLEAR INPUT BUFFER
3320							
3321	011244	004437	033612		JSR	R4,BLDDAT	:GO BUILD DATA
3322	011250	000007			?		:PATTERN ?
3323							
3324	011252	012700	000530		MOV	#530,R0	:SET INDEX INTO BUFFER
3325							
3326	011256	012737	177446	052444	MOV	#177446,0BLFF+1000	:STORE ECC WORDS
3327	011264	012737	015457	052446	MOV	#15457,0BLFF+1002	
3328	011272	012737	000001	002344	MOV	#1,ECCSRC	:SET SOURCE TO INDICATE ECC IN BUFFER
3329	011300	032760	000100	051444	BIT	#BIT6,0BUFF,R0)	:TEST IF BIT SET
3330	011306	001004			BNE	1005	:YES - SKIP
3331	011310	052760	000100	051444	BIS	#BIT6,0BUFF,R0)	:ELSE SET BIT
3332	011316	000403			BR	1015	
3333	011320	042760	000100	051444	BIC	#BIT6,0BUFF,R0)	:CLEAR BIT
3334	011326						
3335							
3336	011326	004437	034072		JSR	R4,OPSTRT	:START THE OPERATION

E06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MC-11-D2R6E-A MACY11 27 1006 05-OCT-76 09:12 PAGE 69
 D2R6EA.P11 28-SEP-76 15:31 T16 REAC DATA WITH ONE BIT BURST ERROR (PART 1)

SEQ 0069

3337	011332	004437	025100		JSR	R4,DISEEK	:SIMULATE IMPLIED SEEK	
3338	011336	104004			ERROR	4	:CSI MISCOMPARE	
3339	011340	104005			ERROR	5	:MRI	
3340	011342	104006			ERROR	6	:MRI	
3341	011344	104007			ERROR	7	:MRI	
3342	011346	004437	025062		JSR	R4,DSECTA	:SIMULATE SECTOR PULSE	
3343	011352	004437	025504		JSR	R4,DRSYNC	:SIMULATE HEADER PREAMBLE	
3344	011356	104010			ERROR	10	:MRI CONTENTS IN ERROR	
3345	011360	104011			ERROR	11	:CSI CONTENTS IN ERROR	
3346	011362	004437	026042		JSR	R4,DHDCMP	:SIMULATE HEADER SEARCH	
3347	011366	104012			ERROR	12	:MRI CONTENTS IN ERROR	
3348	011370	104013			ERROR	13	:CSI IN ERROR AFTER SEARCH	
3349	011372	104014			ERROR	14	:RKDCYL OR RKDA IN ERROR AFTER SEARCH	
3350	011374	004437	030276		JSR	R4,DRGPSN	:GO READ GAP AND SYNC	
3351	011400	104015			ERROR	15	:MRI IN ERROR READING GAP	
3352	011402	104016			ERROR	16	:CSI IN ERROR AFTER READING GAP	
3353	011404	104032			ERROR	32	:MRI IN ERROR READING SYNC	
3354	011406	104033			ERROR	33	:CSI IN ERROR AFTER READING SYNC	
3355	011410	012737	000001	002346	MOV	#1,ECPATX	:LOAD EXPECTED PATTERN	
3356	011416	012737	004066	002350	MOV	#4066,ECP0SX	:LOAD EXPECTED POSITION	
3357	011424	004437	030774		JSR	R4,DRDREAD	:GO SIMULATE DATA READ	
3358	011430	000400			400		:NUMBER OF WORDS TO SIMULATE	
3359	011432	104034			ERROR	34	:MRI IN ERROR READING DATA OR ECC	
3360	011434	104035			ERROR	35	:ECC ERROR READING DATA	
3361	011436	104036			ERROR	36	:CSI ERROR AFTER READING DATA OR ECC	
3362	011440	104041			ERROR	41	:ECC REG INCORRECT AFTER ECC READ	
3363	011442	104042			ERROR	42	:ERR IN ECC PAT CALC.	
3364	011444	104043			ERROR	43	:ERR IN ECC POS COUNT	
3365	011446	012737	000020	002326	MOV	#20,BITCNT	:SET BIT COUNT FOR CORRECTION COUNT	
3366	011454	012700	000005		MOV	#5,RD	:SET COUNT FOR ECCPOS PASS THRU 0	
3367	011460	012701	013672		MOV	#13672,R1	:SET COUNT FOR ECCPOS TO 0 FIRST PASS	
3368	011464	005037	001214		CLR	\$TMPS	:CLEAR SWITCH	
3369	011470	004737	033414	3\$:	JSR	PC,RDBIT	:GO READ BIT	
3370	011474	004737	032260		JSR	PC,ECCGEN	:GENERATE ECC	
3371	011500	016237	000032	002214	MOV	RKECPT(R2),T.ECPT	:GET PATTERN	
3372	011506	016237	000030	002212	MOV	RKECPS(R2),T.ECPS	:GET POSITION	
3373	011514	005237	002350		INC	ECP0SX	:BUMP POSITION EXPECTED	
3374	011520	013737	002350	002252	MOV	ECP0SX,E.ECPS	:SET FOR COMPARE AND REPORT	
3375	011526	023737	002254	002214	CMP	E.ECPT,T.ECPT	:CHECK IF PATTERN CORRECT	
3376	011534	001401			BEQ	1\$:YES - SKIP	
3377	011536	104042			ERROR	42		
3378	011540	023737	002252	002212	1\$:	CMP	E.ECPS,T.ECPS	:CHECK IF POSITION CORRECT
3379	011546	001401			BEQ	2\$:YES - SKIP	
3380	011550	104043			ERROR	43		
3381	011552	005237	002326	2\$:	INC	BITCNT	:BUMP BIT COUNT	
3382	011556	005301			DEC	R1	:DEC COUNT TO ZERO IN ECCPOS	
3383	011560	001343			BNE	3\$:NOT YET ZERO - LOOP	
3384	011562	005300			DEC	RD	:DEC PASS THRU 0 COUNT	

F06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27,1006) 05-OCT-76 09:12 PAGE 70
 DZR6EA.P11 28-SEP-76 15:31 T16 READ DATA WITH ONE BIT BURST ERROR (PART 1)

SEG 0070

```

3393 011564 001406          BEQ      4$          ;IF 0 - SKIP
3394 011566 012737 177777 002350  MOV     #-1,ECPOSX   ;PRESET EXPECTED POS TO GO TO ZERO
3395 011574 012701 020000          MOV     #20000,R1    ;SET COUNT TO NEXT 0 IN ECC POS
3396 011600 000733          BR      3$          ;GO DO LOOP AGAIN
3397
3398 011602 005737 001214          4$:   TST     $TMP5      ;TEST SWITCH
3399 011606 001012          BNE     5$          ;A_READY SET - SKIP
3400 011610 012701 005276          MOV     #5276,R1    ;SET COUNT TO FINAL POSITION
3401 011614 012700 000001          MOV     #1,R0       ;SET R0 TO RETURN AFTER ONE PASS
3402 011620 012737 177777 002350  MOV     #-1,ECPOSX   ;SET EXP POS TO GO TO 0 ON NEXT BIT
3403 011626 005237 001214          INC     $TMP5      ;BUMP SWITCH
3404 011632 000716          BR      3$          ;GO DO LOOP AGAIN
3405
3406 011634 004737 033414          5$:   JSR     PC,RDBIT    ;ONE MORE BIT TO SET READY
3407 011640 004437 034154          JSR     R4,GETREG   ;GET '611 REGISTERS
3408 011644 013737 002260 002220  MOV     L.CS1,E.CS1 ;GET EXPECTED CS1
3409 011652 052737 100200 002220  BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
3410 011660 042737 000001 002220  BIC     #GO,E.CS1   ;CLEAR GO
3411 011666 023737 002160 002220  CMP     T.CS1,E.CS1 ;CHECK IF CS1 CORRECT
3412 011674 001402          BEQ     6$          ;YES - SKIP
3413 011676 104044          ERROR   44
3414 011700 003410          BR      7$
3415 011702 032737 000100 002174  6$:   BIT     #ECH,T.ER   ;TEST IF ECC HARD ERROR SET
3416 011710 001404          BEQ     7$          ;NO - SKIP
3417 011712 012737 100000 002234  MOV     #DCK,E.ER   ;SET EXPECTED ERROR REG
3418 011720 104045          ERROR   45
3419 011722          7$:

```

```

*****
*TEST 17      READ DATA WITH 11 BIT BURST ERROR
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
* OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
* AN 11 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION
* REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
* MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
*****

```

```

3435 011722 000004          †ST17: SCOPE
3436 011724 012737 000001 001234  MOV     #1,$TIMES   ;DO 1 ITERATION
3437 011732 005037 002352          CLR     BSEDES     ;CLEAR BAD SEC DESIRED
3438 011736 012762 100000 000000  MOV     #CLR,RKCS1(R2) ;CLEAR CONTROLLER
3439
3440 011744 005037 002354          CLR     HIBITS     ;CLEAR HI ORDER BITS
3441 011750 004437 034026          JSR     R4,LOADRK  ;LOAD "L" REGISTERS
3442 011754 000000          0              ;CYLINDER 0
3443 011756          0              ;SECTOR 0
3444 011757          0              ;TRACK 0
3445 011760 050440          Ibuff         ;BUFFER ADDRESS Ibuff
3446 011762 177400          -400          ;WORD COUNT -400
3447 011764 000021          RDATA        ;COMMAND RDATA
3448

```

G06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 71
 DZR6EA.P11 28-SEP-76 15:31 T17 READ DATA WITH 11 BIT BURST ERROR

SEQ 0071

3449	011766	004437	033566		JSR	R4,CLRIBF	;GO CLEAR INPUT BUFFER
3450							
3451	011772	004437	033612		JSR	R4,BLDDAT	;GO BUILD DATA
3452	011776	000007			7		;PATTERN 7
3453							
3454	012000	012700	000530		MOV	#530,R0	;SET INDEX INTO BUFFER
3455							
3456	012004	012737	177446	052444	MOV	#177446,0BUFF+1000	;STORE ECC WORDS
3457	012012	012737	015457	052446	MOV	#15457,0BUFF+1002	
3458	012020	012737	000001	002344	MOV	#1,ECCSRC	;SET SOURCE TO INDICATE ECC IN BUFFER
3459	012026	032760	000100	051444	BIT	#BIT6,0BUFF(R0)	;TEST IF BIT SET
3460	012034	001004			BNE	100\$;YES - SKIP
3461	012036	052760	000100	051444	BIS	#BIT6,0BUFF(R0)	;ELSE SET BIT
3462	012044	000403			BR	101\$	
3463	012046	042760	000100	051444	BIC	#BIT6,0BUFF(R0)	;CLEAR BIT
3464	012054						100\$: 101\$:
3465							
3466	012054	004437	034072		JSR	R4,OPSTRT	;START THE OPERATION
3467	012060	004437	025100		JSR	R4,DISEEK	;SIMULATE IMPLIED SEEK
3468	012064	104004			ERROR	4	;CS1 MISCOMPARE
3469	012066	104005			ERROR	5	;MR1 "
3470	012070	104006			ERROR	6	;MR2 "
3471	012072	104007			ERROR	7	;MR3 "
3472							
3473	012074	004437	025062		JSR	R4,DSECTR	;SIMULATE SECTOR PULSE
3474							
3475	012100	004437	025504		JSR	R4,DRSYNC	;SIMULATE HEADER PREAMBLE
3476	012104	104010			ERROR	10	;MR1 CONTENTS IN ERROR
3477	012106	104011			ERROR	11	;CS1 CONTENTS IN ERROR
3478							
3479	012110	004437	026042		JSR	R4,DHDCMP	;SIMULATE HEADER SEARCH
3480	012114	104012			ERROR	12	;MR1 CONTENTS IN ERROR
3481	012116	104013			ERROR	13	;CS1 IN ERROR AFTER SEARCH
3482	012120	104014			ERROR	14	;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3483							
3484	012122	004437	030276		JSR	R4,DRGPSN	;GO READ GAP AND SYNC
3485	012126	104015			ERROR	15	;MR1 IN ERROR READING GAP
3486	012130	104016			ERROR	16	;CS1 IN ERROR AFTER READING GAP
3487	012132	104032			ERROR	32	;MR1 IN ERROR READING SYNC
3488	012134	104033			ERROR	33	;CS1 IN ERROR AFTER READING SYNC
3489							
3490	012136	012737	000001	002346	MOV	#1,ECPATX	;LOAD EXPECTED PATTERN
3491	012144	012737	004066	002350	MOV	#4066,ECP0SX	;LOAD EXPECTED POSITION
3492	012152	004437	030774		JSR	R4,DREAD	;GO SIMULATE DATA READ
3493	012156	000400			400		;NUMBER OF WORDS TO SIMULATE
3494	012160	104034			ERROR	34	;MR1 IN ERROR READING DATA OR ECC
3495	012162	104035			ERROR	35	;ECC ERROR READING DATA
3496	012164	104036			ERROR	36	;CS1 ERROR AFTER READING DATA OR ECC
3497	012166	104041			ERROR	41	;ECC REG INCORRECT AFTER ECC READ
3498	012170	104042			ERROR	42	;ERR IN ECC PAT CALC.
3499	012172	104043			ERROR	43	;ERR IN ECC POS COUNT
3500							
3501							
3502	012174	012737	000020	002326	MOV	#20,BITCNT	;SET BIT COUNT FOR CORRECTION COUNT
3503	012202	012700	000005		MOV	#5,R0	;SET COUNT FOR ECCPOS PASS THRU 0
3504	012206	012701	013672		MOV	#13672,R1	;SET COUNT FOR ECCPOS TO 0 FIRST PASS

H06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 72
 CZR6EA.P11 28-SEP-76 15:31 T17 READ DATA WITH 11 BIT BURST ERROR

SEG 0072

```

3505 012212 005037 001214          CLR      $TMP5          ;CLEAR SWITCH
3506
3507 012216 004737 033414          3$:     JSR      PC,RDBIT          ;GO READ BIT
3508 012222 004737 032260          JSR      PC,ECCGEN          ;GENERATE ECC
3509 012226 016237 000032 002214  MOV      RKECPT(R2),T.ECPT    ;GET PATTERN
3510 012234 016237 000030 002212  MOV      RKECPS(R2),T.ECPS    ;GET POSITION
3511 012242 005237 002350          INC      ECPOSX              ;BUMP POSITION EXPECTED
3512 012246 013737 002350 002252  MOV      ECPOSX,E.ECPS        ;SET FOR COMPARE AND REPORT
3513 012254 023737 002254 002214  CMP      E.ECPT,T.ECPT        ;CHECK IF PATTERN CORRECT
3514 012262 001401          BEQ      1$                  ;YES - SKIP
3515 012264 104042          ERROR    42
3516 012266 023737 002252 002212  1$:     CMP      E.ECPS,T.ECPS        ;CHECK IF POSITION CORRECT
3517 012274 001401          BEQ      2$                  ;YES - SKIP
3518 012276 104043          ERROR    43
3519 012300 005237 002326          2$:     INC      BITCNT              ;BUMP BIT COUNT
3520 012304 005301          DEC      R1                  ;DEC COUNT TO ZERO IN ECCPOS
3521 012306 001343          BNE      3$                  ;NOT YET ZERO - LOOP
3522 012310 005300          DEC      R0                  ;DEC PASS THRU 0 COUNT
3523 012312 001406          BEQ      4$                  ;IF 0 - SKIP
3524 012314 012737 177777 002350  MOV      #-1,ECPOSX          ;PRESET EXPECTED POS TO GO TO ZERO
3525 012322 012701 020000          MOV      #20000,R1           ;SET COUNT TO NEXT 0 IN ECC POS
3526 012326 000733          BR       3$                  ;GO DO LOOP AGAIN
3527
3528 012330 005737 001214          4$:     TST      $TMP5              ;TEST SWITCH
3529 012334 001012          BNE      5$                  ;ALREADY SET - SKIP
3530 012336 012701 005276          MOV      #5276,R1           ;SET COUNT TO FINAL POSITION
3531 012342 012700 000001          MOV      #1,R0              ;SET R0 TO RETURN AFTER ONE PASS
3532 012346 012737 177777 002350  MOV      #-1,ECPOSX          ;SET EXP POS TO GO TO 0 ON NEXT BIT
3533 012354 005237 001214          INC      $TMP5              ;BUMP SWITCH
3534 012360 000716          BR       3$                  ;GO DO LOOP AGAIN
3535
3536 012362 004737 033414          5$:     JSR      PC,RDBIT              ;ONE MORE BIT TO SET READY
3537 012366 004437 034154          JSR      R4,GETREG           ;GET '611 REGISTERS
3538 012372 013737 002260 002220  MOV      L.CS1,E.CS1         ;GET EXPECTED CS1
3539 012400 052737 100200 002220  BIS      #RDY!CERR,E.CS1     ;SET READY AND ERROR
3540 012406 042737 000001 002220  BIC      #GO,E.CS1           ;CLEAR GO
3541 012414 023737 002160 002220  CMP      T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
3542 012422 001402          BEQ      6$                  ;YES - SKIP
3543 012424 104044          ERROR    44
3544 012426 000410          BR       7$                  ;
3545 012430 032737 000100 002174  6$:     BIT      #ECH,T.ER          ;TEST IF ECC HARD ERROR SET
3546 012436 001404          BEQ      7$                  ;NO SKIP
3547 012440 012737 100000 002234  MOV      #DCK,E.ER          ;SET EXPECTED ERROR REG
3548 012446 104045          ERROR    45
3549
3550 012450          7$:
3551
3552  ;*****
3553  ;*TEST 20      READ DATA WITH ONE BIT BURST ERROR (PART 2)
3554  ;*
3555  ;*
3556  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3557  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3558  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3559  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3560  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3561  ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING

```


I06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 73
 DZR6EA.F11 28-SEP-76 15:31 T20 READ DATA WITH ONE BIT BURST ERROR (PART 2)

SEG 0073

```

3561      :*      A ONE BIT BURST ERROR OF A 1 THAT SHOULD BE A 0.
3562      :*      VERIFY THE COUNTING OF THE POSITION
3563      :*      REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
3564      :*      MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
3565      :*
3566      :*
3567      *****
3567 012450 000004      †ST20: SCOPE
3568 012452 012737 000001 001234      MOV      #1,$TIMES      ;;DO 1 ITERATION
3569
3570 012460 005037      CLR      BSEDES      ;CLEAR BAD SEC DESIRED
3571 012464 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3572
3573 012472 005037 002352      CLR      HIBITS      ;CLEAR HI ORDER BITS
3574 012476 004437 034026      JSR      R4,LOADRK    ;LOAD "L" REGISTERS
3575 012502 000000      0          ;CYLINDER 0
3576 012504      000      .BYTE 0      ;SECTOR 0
3577 012505      000      .BYTE 0      ;TRACK 0
3578 012506 050440      Ibuff     ;BUFFER ADDRESS Ibuff
3579 012510 177400      -400     ;WORD COUNT -400
3580 012512 000021      RDDATA   ;COMMAND RDDATA
3581
3582 012514 004437 033566      JSR      R4,CLRIBF    ;GO CLEAR INPUT BUFFER
3583
3584 012520 004437 033612      JSR      R4,BLDDAT    ;GO BUILD DATA
3585 012524 000005      5        ;PATTERN 5
3586
3587 012526 012700 000530      MOV      #530,R0      ;SET INDEX INTO BUFFER
3588
3589 012532 012737 126073 052444      MOV      #126073,Obuff+1000 ;STORE ECC WORDS
3590 012540 012737 151052 052446      MOV      #151052,Obuff+1002
3591 012546 012737 000001 002344      MOV      #1,ECCSRC    ;SET SOURCE TO INDICATE ECC IN BUFFER
3592 012554 032760 000100 051444      BIT      #BIT6,Obuff(R0) ;TEST IF BIT SET
3593 012562 001004      BNE      100$         ;YES - SKIP
3594 012564 052760 000100 051444      BIS      #BIT6,Obuff(R0) ;ELSE SET BIT
3595 012572 000403      BR       101$
3596 012574 042760 000100 051444 100$: BIC      #BIT6,Obuff(R0) ;CLEAR BIT
3597 012602      101$:
3598
3599 012602 004437 034072      JSR      R4,OPSTAT    ;START THE OPERATION
3600 012606 004437 025100      JSR      R4,DISEEK    ;SIMULATE IMPLIED SEEK
3601 012612 104004      ERROR    4          ;CS1 MISCOMPARE
3602 012614 104005      ERROR    5          ;MR1  "
3603 012616 104006      ERROR    6          ;MR2  "
3604 012620 104007      ERROR    7          ;MR3  "
3605
3606 012622 004437 025062      JSR      R4,DSECTR    ;SIMULATE SECTOR PULSE
3607
3608 012626 004437 025504      JSR      R4,DRSYNC    ;SIMULATE HEADER PREAMBLE
3609 012632 104010      ERROR    10         ;MR1 CONTENTS IN ERROR
3610 012634 104011      ERROR    11         ;CS1 CONTENTS IN ERROR
3611
3612 012636 004437 026042      JSR      R4,DHDCMP    ;SIMULATE HEADER SEARCH
3613 012642 104012      ERROR    12         ;MR1 CONTENTS IN ERROR
3614 012644 104013      ERROR    13         ;CS1 IN ERROR AFTER SEARCH
3615 012646 104014      ERROR    14         ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3616

```

JOB

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 74
 DZR6EA.P11 28-SEP-76 15:31 T20 READ DATA WITH ONE BIT BURST ERROR (PART 2)

SEG 0074

3617	012650	004437	030276		JSR	R4,DRGPSN		;GO READ GAP AND SYNC
3618	012654	104015			ERROR	15		;MR1 IN ERROR READING GAP
3619	012656	104016			ERROR	16		;CSI IN ERROR AFTER READING GAP
3620	012660	104032			ERROR	32		;MR1 IN ERROR READING SYNC
3621	012662	104033			ERROR	33		;CSI IN ERROR AFTER READING SYNC
3622								
3623	012664	012737	000001	002346	MOV	#1,ECPATX		;LOAD EXPECTED PATTERN
3624	012672	012737	004066	002350	MOV	#4066,ECPOX		;LOAD EXPECTED POSITION
3625	012700	004437	030774		JSR	R4,DREAD		;GO SIMULATE DATA READ
3626	012704	000400			400			;NUMBER OF WORDS TO SIMULATE
3627	012706	104034			ERROR	34		;MR1 IN ERROR READING DATA OR ECC
3628	012710	104035			ERROR	35		;ECC ERROR READING DATA
3629	012712	104036			ERROR	36		;CSI ERROR AFTER READING DATA OR ECC
3630	012714	104041			ERROR	41		;ECC REG INCORRECT AFTER ECC READ
3631	012716	104042			ERROR	42		;ERR IN ECC PAT CALC.
3632	012720	104043			ERROR	43		;ERR IN ECC POS COUNT
3633								
3634								
3635	012722	012737	000020	002326	MOV	#20,BITCNT		;SET BIT COUNT FOR CORRECTION COUNT
3636	012730	012700	000005		MOV	#5,R0		;SET COUNT FOR ECCPOS PASS THRU 0
3637	012734	012701	013672		MOV	#13672,R1		;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3638	012740	005037	001214		CLR	\$TMP5		;CLEAR SWITCH
3639								
3640	012744	004737	033414		3\$: JSR	PC,RDBIT		;GO READ BIT
3641	012750	004737	032260		JSR	PC,ECCGEN		;GENERATE ECC
3642	012754	016237	000032	002214	MOV	RKECPT(R2),T.ECPT		;GET PATTERN
3643	012762	016237	000030	002212	MOV	RKECPS(R2),T.ECPS		;GET POSITION
3644	012770	005237	002350		INC	ECPOX		;BUMP POSITION EXPECTED
3645	012774	013737	002350	002252	MOV	ECPOX,E.ECPS		;SET FOR COMPARE AND REPORT
3646	013002	023737	002254	002214	CMP	E.ECPT,T.ECPT		;CHECK IF PATTERN CORRECT
3647	013010	001401			BEQ	1\$;YES - SKIP
3648	013012	104042			ERROR	42		
3649	013014	023737	002252	002212	1\$: CMP	E.ECPS,T.ECPS		;CHECK IF POSITION CORRECT
3650	013022	001401			BEQ	2\$;YES - SKIP
3651	013024	104043			ERROR	43		
3652	013026	005237	002326		2\$: INC	BITCNT		;BUMP BIT COUNT
3653	013032	005301			DEC	R1		;DEC COUNT TO ZERO IN ECCPOS
3654	013034	001343			BNE	3\$;NOT YET ZERO - LOOP
3655	013036	005300			DEC	R0		;DEC PASS THRU 0 COUNT
3656	013040	001406			BEQ	4\$;IF 0 - SKIP
3657	013042	012737	177777	002350	MOV	#-1,ECPOX		;PRESET EXPECTED POS TO GO TO ZERO
3658	013050	012701	020000		MOV	#20000,R1		;SET COUNT TO NEXT 0 IN ECC POS
3659	013054	000733			BR	3\$;GO DO LOOP AGAIN
3660								
3661	013056	005737	001214		4\$: TST	\$TMP5		;TEST SWITCH
3662	013062	001012			BNE	5\$;ALREADY SET - SKIP
3663	013064	012701	005276		MOV	#5276,R1		;SET COUNT TO FINAL POSITION
3664	013070	012700	000001		MOV	#1,R0		;SET R0 TO RETURN AFTER ONE PASS
3665	013074	012737	177777	002350	MOV	#-1,ECPOX		;SET EXP POS TO GO TO 0 ON NEXT BIT
3666	013102	005237	001214		INC	\$TMP5		;BUMP SWITCH
3667	013106	000716			BR	3\$;GO DO LOOP AGAIN
3668								
3669	013110	004737	033414		5\$: JSR	PC,RDBIT		;ONE MORE BIT TO SET READY
3670	013114	004437	034154		JSR	R4,GETREG		;GET '611 REGISTERS
3671	013120	013737	002260	002220	MOV	L.CS1,E.CS1		;GET EXPECTED CS1
3672	013126	052737	100200	002220	BIS	#RDY!CERR,E.CS1		;SET READY AND ERROR

K06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 75
 DZR6EA.P11 28-SEP-76 15:31 T20 READ DATA WITH ONE BIT BURST ERROR (PART 2)

SEG 0075

3673	013134	042737	000001	002220	BIC	#GO,E.CS1	:CLEAR GO
3674	013142	023737	002160	002220	CMP	T.CS1,E.CS1	:CHECK IF CS1 CORRECT
3675	013150	001402			BEQ	6\$:YES - SKIP
3676	013152	104044			ERROR	'4	
3677	013154	000410			BR	7\$	
3678	013156	032737	000100	002174	6\$: BIT	#ECH,T.ER	:TEST IF ECC HARD ERROR SET
3679	013164	001404			BEQ	7\$:NO SKIP
3680	013166	012737	100000	002234	MOV	#DCK,E.ER	:SET EXPECTED ERROR REG
3681	013174	104045			ERROR	45	
3682							
3683	013176				7\$:		
3684							
3685							
3686					:*****		
3687					:TEST 21 READ DATA WITH 11 BIT BURST ERROR		
3688					:*		
3689					:CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.		
3690					:PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA		
3691					:OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,		
3692					:CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK		
3693					:AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,		
3694					:A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING		
3695					:AN 11 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION		
3696					:REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.		
3697					:MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.		
3698					:*****		
3699	013176	000004			TST21:	SCOPE	
3700	013200	012737	000001	001234	MOV	#1,\$TIMES	::DO 1 ITERATION
3701							
3702	013206	005037	002352		CLR	BSEDES	:CLEAR BAD SEC DESIRED
3703							
3704	013212	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR CONTROLLER
3705							
3706	013220	005037	002354		CLR	HIBITS	:CLEAR HI ORDER BITS
3707	013224	004437	034026		JSR	R4,LOADRK	:LOAD "L" REGISTERS
3708	013230	000000			0		:CYLINDER 0
3709	013232	000			.BYTE	0	:SECTOR 0
3710	013233	000			.BYTE	0	:TRACK 0
3711	013234	050440			IBUFF		:BUFFER ADDRESS IBUFF
3712	013236	177400			-400		:WORD COUNT -400
3713	013240	000021			RDDATA		:COMMAND RDDATA
3714							
3715	013242	004437	033566		JSR	R4,CLRIBF	:GO CLEAR INPUT BUFFER
3716							
3717	013246	004437	033612		JSR	R4,BLDDAT	:GO BUILD DATA
3718	013252	000005			5		:PATTERN 5
3719							
3720	013254	012700	000530		MOV	#530,R0	:SET INDEX INTO BUFFER
3721							
3722	013260	012737	126073	052444	MOV	#126073,OBUFF+1000	:STORE ECC WORDS
3723	013266	012737	151052	052446	MOV	#151052,OBUFF+1002	
3724	013274	012737	000001	002344	MOV	#1,ECCSAC	:SET SOURCE TO INDICATE ECC IN BUFFER
3725	013302	042760	005000	051444	BIC	#BIT9!BIT11,OBUFF(R0)	:CREATE 11 BIT BURST ERROR
3726	013310	052760	050100	051444	BIS	#BIT6!BIT12!BIT14,OBUFF(R0)	
3727	013316	062700	000002		ADD	#2,R0	:BUMP TO NEXT WORD
3728	013322	052760	000001	051444	BIS	#BIT0,OBUFF(R0)	

L06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 76
 DZR6EA.P11 28-SEP-76 15:31 T21 READ DATA WITH 11 BIT BURST ERROR

SEQ 0076

3729										
3730	013330	004437	034072		JSR	R4,OPSTRT			;START THE OPERATION	
3731	013334	004437	025100		JSR	R4,DISEEK			;SIMULATE IMPLIED SEEK	
3732	013340	104004			ERROR	4			;CSI MISCOMPARE	
3733	013342	104005			ERROR	5			;MR1 "	
3734	013344	104006			ERROR	6			;MR2 "	
3735	013346	104007			ERROR	7			;MR3 "	
3736										
3737	013350	004437	025062		JSR	R4,DSECTR			;SIMULATE SECTOR PULSE	
3738										
3739	013354	004437	025504		JSR	R4,DRSYNC			;SIMULATE HEADER PREAMBLE	
3740	013360	104010			ERROR	10			;MR1 CONTENTS IN ERROR	
3741	013362	104011			ERROR	11			;CSI CONTENTS IN ERROR	
3742										
3743	013364	004437	026042		JSR	R4,DHDCMP			;SIMULATE HEADER SEARCH	
3744	013370	104012			ERROR	12			;MR1 CONTENTS IN ERROR	
3745	013372	104013			ERROR	13			;CSI IN ERROR AFTER SEARCH	
3746	013374	104014			ERROR	14			;RKDCYL OR RKDA IN ERROR AFTER SEARCH	
3747										
3748	013376	004437	030276		JSR	R4,DRGPSN			;GO READ GAP AND SYNC	
3749	013402	104015			ERROR	15			;MR1 IN ERROR READING GAP	
3750	013404	104016			ERROR	16			;CSI IN ERROR AFTER READING GAP	
3751	013406	104032			ERROR	32			;MR1 IN ERROR READING SYNC	
3752	013410	104033			ERROR	33			;CSI IN ERROR AFTER READING SYNC	
3753										
3754	013412	012737	000001	002346	MOV	#1,ECPATX			;LOAD EXPECTED PATTERN	
3755	013420	012737	004066	002350	MOV	#4066,ECPOSX			;LOAD EXPECTED POSITION	
3756	013426	004437	030774		JSR	R4,DRREAD			;GO SIMULATE DATA READ	
3757	013432	000400			400				;NUMBER OF WORDS TO SIMULATE	
3758	013434	104034			ERROR	34			;MR1 IN ERROR READING DATA OR ECC	
3759	013436	104035			ERROR	35			;ECC ERROR READING DATA	
3760	013440	104036			ERROR	36			;CSI ERROR AFTER READING DATA OR ECC	
3761	013442	104041			ERROR	41			;ECC REG INCORRECT AFTER ECC READ	
3762	013444	104042			ERROR	42			;ERR IN ECC PAT CALC.	
3763	013446	104043			ERROR	43			;ERR IN ECC POS COUNT	
3764										
3765										
3766	013450	012737	000020	002326	MOV	#20,BITCNT			;SET BIT COUNT FOR CORRECTION COUNT	
3767	013456	012700	000005		MOV	#5,R0			;SET COUNT FOR ECCPOS PASS THRU 0	
3768	013462	012701	013672		MOV	#13672,R1			;SET COUNT FOR ECCPOS TO 0 FIRST PASS	
3769	013466	005037	001214		CLR	\$TMP5			;CLEAR SWITCH	
3770										
3771	013472	004737	033414		3\$: JSR	PC,RDBIT			;GO READ BIT	
3772	013476	004737	032260		JSR	PC,ECCGEN			;GENERATE ECC	
3773	013502	016237	000032	002214	MOV	RKECPT(R2),T.ECPT			;GET PATTERN	
3774	013510	016237	000030	002212	MOV	RKECPS(R2),T.ECPS			;GET POSITION	
3775	013516	005237	002350		INC	ECPOSX			;BUMP POSITION EXPECTED	
3776	013522	013737	002350	002252	MOV	ECPOSX,E.ECPS			;SET FOR COMPARE AND REPORT	
3777	013530	023737	002254	002214	CMP	E.ECPT,T.ECPT			;CHECK IF PATTERN CORRECT	
3778	013536	001401			BEG	1\$;YES - SKIP	
3779	013540	104042			ERROR	42				
3780	013542	023737	002252	002212	1\$: CMP	E.ECPS,T.ECPS			;CHECK IF POSITION CORRECT	
3781	013550	001401			BEG	2\$;YES - SKIP	
3782	013552	104043			ERROR	43				
3783	013554	005237	002326		2\$: INC	BITCNT			;BUMP BIT COUNT	
3784	013560	005301			DEC	R1			;DEC COUNT TO ZERO IN ECCPOS	

M06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 77
 DZR6EA.P11 28-SEP-76 15:31 T21 READ DATA WITH 11 BIT BURST ERROR

SEQ 0077

```

3785 013562 001343      BNE      3$          ;NOT YET ZERO - LOOP
3786 013564 005300      DEC      R0          ;DEC PASS THRU 0 COUNT
3787 013566 001406      BEQ      4$          ;IF 0 - SKIP
3788 013570 012737 177777 002350      MOV      #-1,ECPOSX ;PRESET EXPECTED POS TO GO TO ZERO
3789 013576 012701 020000      MOV      #20000,R1  ;SET COUNT TO NEXT 0 IN ECC POS
3790 013602 000733      BR       3$          ;GO DO LOOP AGAIN
3791
3792 013604 005737 001214      4$:      TST      $TMP5    ;TEST SWITCH
3793 013610 001012      BNE      5$          ;ALREADY SET - SKIP
3794 013612 012701 005310      MOV      #5310,R1   ;SET COUNT TO FINAL POSITION
3795 013616 012700 000001      MOV      #1,R0      ;SET R0 TO RETURN AFTER ONE PASS
3796 013622 012737 177777 002350      MOV      #-1,ECPOSX ;SET EXP POS TO GO TO 0 ON NEXT BIT
3797 013630 005237 001214      INC      $TMP5    ;BUMP SWITCH
3798 013634 000716      BR       3$          ;GO DO LOOP AGAIN
3799
3800 013636 004737 033414      5$:      JSR      PC,RDBIT   ;ONE MORE BIT TO SET READY
3801 013642 004437 034154      JSR      R4,GETREG  ;GET '611 REGISTERS
3802 013646 013737 002260 002220      MOV      L.CS1,E.CS1 ;GET EXPECTED CS1
3803 013654 052737 100200 002220      BIS      #RDY!CERR,E.CS1 ;SET READY AND ERROR
3804 013662 042737 000001 002220      BIC      #GO,E.CS1  ;CLEAR GO
3805 013670 023737 002160 002220      CMP      T.CS1,E.CS1 ;CHECK IF CS1 CORRECT
3806 013676 001402      BEQ      6$          ;YES - SKIP
3807 013700 104044      ERROR   44
3808 013702 000410      BR       7$
3809 013704 032737 000100 002174      6$:      BIT      #ECH,T.ER  ;TEST IF ECC HARD ERROR SET
3810 013712 001404      BEQ      7$          ;NO SKIP
3811 013714 012737 100000 002234      MOV      #DCK,E.ER  ;SET EXPECTED ERROR REG
3812 013722 104045      ERROR   45
3813
3814 013724      7$:
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830 013724 000004      ;*****
3831 013726 012737 000001 001234      ;TEST 22      READ DATA WITH 12 BIT BURST ERROR
3832
3833
3834
3835
3836
3837
3838
3839
3840
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3830 013724 000004      ;*****
3831 013726 012737 000001 001234      ;TEST 22:  SCOPE
3832
3833 013734 005037 002352      MOV      #1,$TIMES  ;;DO 1 ITERATION
3834 013740 012762 100000 000000      CLR      BSEDES    ;CLEAR BAD SEC DESIRED
3835
3836 013746 005037 002354      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3837 013752 004437 034026      CLR      HIBITS    ;CLEAR HI ORDER BITS
3838 013756 000000      JSR      R4,LOADRK ;LOAD "L" REGISTERS
3839 013760 000      0          ;CYLINDER 0
3840 013761 000      .BYTE 0          ;SECTOR 0
          .BYTE 0          ;TRACK 0
  
```

N06

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 78
 DZR6EA.P11 28-SEP-76 15:31 T22 READ DATA WITH 12 BIT BURST ERROR

SEQ 0078

3841	013762	050440			IBUFF		;BUFFER ADDRESS IBUFF
3842	013764	177400			-400		;WORD COUNT -400
3843	013766	000021			RDDATA		;COMMAND RDDATA
3844							
3845	013770	004437	033566		JSR	R4,CLRIBF	;GO CLEAR INPUT BUFFER
3846							
3847	013774	004437	033612		JSR	R4,BLDDAT	;GO BUILD DATA
3848	014000	000005			5		;PATTERN 5
3849							
3850	014002	012700	000530		MOV	#530,R0	;SET INDEX INTO BUFFER
3851							
3852	014006	012737	126073	052444	MOV	#126073,OBUFF+1000	;STORE ECC WORDS
3853	014014	012737	151052	052446	MOV	#151052,OBUFF+1002	
3854	014022	012737	000001	002344	MOV	#1,ECCSRC	;SET SOURCE TO INDICATE ECC IN BUFFER
3855	014030	042760	005000	051444	BIC	#BIT9:BIT11,OBUFF(R0)	;CREATE 11 BIT BURST ERROR
3856	014036	052760	050100	051444	BIS	#BIT6:BIT12:BIT14,OBUFF(R0)	
3857	014044	062700	000002		ADD	#2,R0	;BUMP TO NEXT WORD
3858	014050	052760	000001	051444	BIS	#BIT0,OBUFF(R0)	
3859	014056	042760	000002	051444	BIC	#BIT1,OBUFF(R0)	;MAKE IT A 12 BIT BURST ERROR
3860							
3861	014064	004437	034072		JSR	R4,OPSTRT	;START THE OPERATION
3862	014070	004437	025100		JSR	R4,DISEEK	;SIMULATE IMPLIED SEEK
3863	014074	104004			ERROR	4	;CS1 MISCOMPARE
3864	014076	104005			ERROR	5	;MR1
3865	014100	104006			ERROR	6	;MR2
3866	014102	104007			ERROR	7	;MR3
3867							
3868	014104	004437	025062		JSR	R4,DSECTR	;SIMULATE SECTOR PULSE
3869							
3870	014110	004437	025504		JSR	R4,DRSYNC	;SIMULATE HEADER PREAMBLE
3871	014114	104010			ERROR	10	;MR1 CONTENTS IN ERROR
3872	014116	104011			ERROR	11	;CS1 CONTENTS IN ERROR
3873							
3874	014120	004437	026042		JSR	R4,DHDCMP	;SIMULATE HEADER SEARCH
3875	014124	104012			ERROR	12	;MR1 CONTENTS IN ERROR
3876	014126	104013			ERROR	13	;CS1 IN ERROR AFTER SEARCH
3877	014130	104014			ERROR	14	;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3878							
3879	014132	004437	030276		JSR	R4,DRGPSN	;GO READ GAP AND SYNC
3880	014136	104015			ERROR	15	;MR1 IN ERROR READING GAP
3881	014140	104016			ERROR	16	;CS1 IN ERROR AFTER READING GAP
3882	014142	104032			ERROR	32	;MR1 IN ERROR READING SYNC
3883	014144	104033			ERROR	33	;CS1 IN ERROR AFTER READING SYNC
3884							
3885	014146	012737	000001	002346	MOV	#1,ECPATX	;LOAD EXPECTED PATTERN
3886	014154	012737	004066	002350	MOV	#4066,ECPOSX	;LOAD EXPECTED POSITION
3887	014162	004437	030774		JSR	R4,DREAD	;GO SIMULATE DATA READ
3888	014166	000400			400		;NUMBER OF WORDS TO SIMULATE
3889	014170	104034			ERROR	34	;MR1 IN ERROR READING DATA OR ECC
3890	014172	104035			ERROR	35	;ECC ERROR READING DATA
3891	014174	104036			ERROR	36	;CS1 ERROR AFTER READING DATA OR ECC
3892	014176	104041			ERROR	41	;ECC REG INCORRECT AFTER ECC READ
3893	014200	104042			ERROR	42	;ERR IN ECC PAT CALC.
3894	014202	104043			ERROR	43	;ERR IN ECC POS COUNT
3895							
3896							

B07

```

3897 014204 012737 000020 002326      MOV      #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3898 014212 012700 000005          MOV      #5,R0          ;SET COUNT FOR ECCPOS PASS THRU 0
3899 014216 012701 013672          MOV      #13&72,R1     ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3900 014222 005037 001214          CLR      $TMP5         ;CLEAR SWITCH
3901
3902 014226 004737 033414      3$:      JSR      PC,R0BIT      ;GO READ BIT
3903 014232 004737 032260          JSR      PC,ECCGEN     ;GENERATE ECC
3904 014235 016237 000032 002214      MOV      RKECPT(R2),T.ECPT ;GET PATTERN
3905 014244 016237 000030 002212      MOV      RKECPS(R2),T.ECPS ;GET POSITION
3906 014252 005237 002350          INC      ECPOSX        ;BUMP POSITION EXPECTED
3907 014256 013737 002350 002252      MOV      ECPOSX,E.ECPS ;SET FOR COMPARE AND REPORT
3908 014264 023737 002254 002214      CMP      E.ECPT,T.ECPT ;CHECK IF PATTERN CORRECT
3909 014272 001401          BEQ     1$            ;YES - SKIP
3910 014274 104042          ERROR   42
3911 014276 023737 002252 002212 1$:      CMP      E.ECPS,T.ECPS ;CHECK IF POSITION CORRECT
3912 014304 001401          BEQ     2$            ;YES - SKIP
3913 014306 104043          ERROR   43
3914 014310 005237 002326      3$:      INC      BITCNT        ;BUMP BIT COUNT
3915 014314 005301          DEC      R1            ;DEC COUNT TO ZERO IN ECCPOS
3916 014316 001343          BNE     3$            ;NOT YET ZERO - LOOP
3917 014320 005300          DEC      R0            ;DEC PASS THRU 0 COUNT
3918 014322 001406          BEQ     4$            ;IF 0 - SKIP
3919 014324 012737 177777 002350      MOV      #-1,ECPOSX    ;PRESET EXPECTED POS TO GO TO ZERO
3920 014332 012701 020000      MOV      #20000,R1     ;SET COUNT TO NEXT 0 IN ECC POS
3921 014336 000733          BR      3$            ;GO DO LOOP AGAIN
3922
3923 014340 005737 001214      4$:      TST     $TMP5         ;TEST SWITCH
3924 014344 001012          BNE     5$            ;ALREADY SET - SKIP
3925 014346 012701 010041      MOV      #10041,R1     ;SET COUNT TO FINAL POSITION
3926 014352 012700 000001      MOV      #1,R0         ;SET R0 TO RETURN AFTER ONE PASS
3927 014356 012737 177777 002350      MOV      #-1,ECPOSX    ;SET EXP POS TO GO TO 0 ON NEXT BIT
3928 014364 005237 001214          INC      $TMP5         ;BUMP SWITCH
3929 014370 000716          BR      3$            ;GO DO LOOP AGAIN
3930
3931 014372 004737 033414      5$:      JSR      PC,R0BIT      ;ONE MORE BIT TO SET READY
3932 014376 004437 034154          JSR      R4,GETREG     ;GET 1611 REGISTERS
3933 014402 013737 002260 002220      MOV      L.CS1,E.CS1  ;GET EXPECTED CS1
3934 014410 052737 100200 002220      BIS      #RDY!ERR,E.CS1 ;SET READY AND ERROR
3935 014416 042737 000001 002220      BIC      #GO,E.CS1     ;CLEAR GO
3936 014424 023737 002160 002220      CMP      T.CS1,E.CS1  ;CHECK IF CS1 CORRECT
3937 014432 001402          BEQ     6$            ;YES - SKIP
3938 014434 104044          ERROR   44
3939 014436 000410          BR      7$            ;
3940 014440 032737 000100 002174 6$:      BIT     #ECH,T.ER      ;TEST IF ECC HARD ERROR SET
3941 014446 001004          BNE     7$            ;YES - SKIP
3942 014450 012737 100100 002234      MOV      #DCK!ECH,E.ER ;SET EXPECTED ERROR REG
3943 014456 104045          ERROR   45
3944
3945 014460      7$:
3946
3947
3948 *****
3949 *TEST 23      READ DATA WITH 23 BIT BURST ERROR
3950 *
3951 *      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3952 *      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
          *      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,

```


C07

```

* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE. A
* GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
* A 23 BIT BURST ERROR. VERIFY THE COUNTING OF THE
* POSITION REGISTER. MAKE SURE DATA CHECK, ECC HARD,
* AND CONTROLLER ERROR SETS.

```

3953										
3954										
3955										
3956										
3957										
3958										
3959										
3960										
3961	014460	000004								
3962	014462	012737	000001	001234						
3963										
3964	014470	005037	002352							
3965	014474	012762	100000	000000						
3966										
3967	014502	005037	002354							
3968	014506	004437	034026							
3969	014512	000000								
3970	014514	000								
3971	014515	000								
3972	014516	050440								
3973	014520	177400								
3974	014522	000021								
3975										
3976	014524	004437	033566							
3977										
3978	014530	004437	033612							
3979	014534	000005								
3980										
3981	014536	012700	000530							
3982										
3983	014542	012737	126073	052444						
3984	014550	012737	151052	052446						
3985	014556	012737	000001	002344						
3986	014564	042760	005000	051444						
3987	014572	052760	050100	051444						
3988	014600	062700	000002							
3989	014604	052760	000001	051444						
3990	014612	052760	010000	051444						
3991										
3992	014620	004437	034072							
3993	014624	004437	025100							
3994	014630	104004								
3995	014632	104005								
3996	014634	104006								
3997	014636	104007								
3998										
3999	014640	004437	025062							
4000										
4001	014644	004437	025504							
4002	014650	104010								
4003	014652	104011								
4004										
4005	014654	004437	026042							
4006	014660	104012								
4007	014662	104013								
4008	014664	104014								

4009	014437	004437	030276		JSR	R4,CRGFSN	:GO READ GAP AND SYNC
4010	014437	104015			ERROR	15	:MRI IN ERROR READING GAP
4011	014437	104016			ERROR	16	:CSI IN ERROR AFTER READING GAP
4012	014437	104032			ERROR	32	:MRI IN ERROR READING SYNC
4013	014437	104033			ERROR	33	:CSI IN ERROR AFTER READING SYNC
4014	014733	012737	000001	002346	MOV	#1,ECPTX	:LOAD EXPECTED PATTERN
4015	014710	012737	004066	002350	MOV	#4066,ECPOSX	:LOAD EXPECTED POSITION
4016	014710	004437	030276		JSR	R4,CRGFSN	:GO SIMULATE DATA READ
4017	014710	000400			LD	#0,0	:NUMBER OF WORDS TO SIMULATE
4018	014710	104034			ERROR	34	:MRI IN ERROR READING DATA OR ECC
4019	014710	104035			ERROR	35	:ECC ERROR READING DATA
4020	014710	104036			ERROR	36	:CSI ERROR AFTER READING DATA OR ECC
4021	014710	104041			ERROR	41	:ECC REG INCORRECT AFTER ECC REAC
4022	014710	104042			ERROR	42	:ERR IN ECC PAT CALC.
4023	014710	104043			ERROR	43	:ERR IN ECC POS COUNT
4024	014710	012737	000020	002326	MOV	#20,BITCNT	:SET BIT COUNT FOR CORRECTION COUNT
4025	014710	012737	000005		MOV	#5,0	:SET COUNT FOR ECCPOS PASS THRU 0
4026	014710	012737	013672		MOV	#13672,R1	:SET COUNT FOR ECCPOS TO 0 FIRST PASS
4027	014710	005037	001214		CLR	#495	:CLEAR SWITCH
4028	014710	004737	033414	38:	JSR	PC,ROBIT	:GO READ BIT
4029	014710	004737	032260		JSR	PC,ECCGEN	:GENERATE ECC
4030	016237	000032	002214		MOV	RKECPT,R2,T,ECPT	:GET PATTERN
4031	016237	000030	002212		MOV	RKECPS,R2,T,ECPS	:GET POSITION
4032	005237	002350			INC	ECPOSX	:BUMP POSITION EXPECTED
4033	013737	002350	002252		MOV	ECPOSX,E,ECPS	:SET FOR COMPARE AND REPORT
4034	013737	002254	002214		COMP	F,ECPT,T,ECPT	:CHECK IF PATTERN CORRECT
4035	001401				BEG	15	:YES - SKIP
4036	104043				ERROR	43	
4037	013737	002252	002212	39:	COMP	F,ECPS,T,ECPS	:CHECK IF POSITION CORRECT
4038	001401				BEG	25	:YES - SKIP
4039	104043				ERROR	43	
4040	005237	002326		28:	INC	BITCNT	:BUMP BIT COUNT
4041	005301				DEC	R1	:DEC COUNT TO ZERO IN ECCPOS
4042	005343				BNE	35	:NOT YET ZERO - LOOP
4043	005300				DEC	R0	:DEC PASS THRU 0 COUNT
4044	001401				BEG	45	:IF 0 - SKIP
4045	012737	177777	002350		MOV	#-1,ECPOSX	:PRESET EXPECTED POS TO GO TO ZERO
4046	012701	020000			MOV	#20000,R1	:SET COUNT TO NEXT 0 IN ECC POS
4047	00733				BR	38	:GO DO LOOP AGAIN
4048	015074	005737	001214	48:	TST	STMP5	:TEST SWITCH
4049	001012				BNE	55	:ALREADY SET - SKIP
4050	012701	010041			MOV	#10041,R1	:SET COUNT TO FINAL POSITION
4051	012700	000001			MOV	#1,R0	:SET R0 TO RETURN AFTER ONE PASS
4052	012737	177777	002350		MOV	#-1,ECPOSX	:SET EXP POS TO GO TO 0 ON NEXT BIT
4053	005237	001214			INC	STMP5	:BUMP SWITCH
4054	000716				BR	38	:GO DO LOOP AGAIN
4055	004737	033414		58:	JSR	PC,ROBIT	:ONE MORE BIT TO SET READY
4056	004437	034154			JSR	R4,GETREG	:GET '611 REGISTERS
4057	013737	002260	002220		MOV	L,CSI,E,CSI	:GET EXPECTED CSI

F07

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 83
 DZR6EA.P11 28-SEP-76 15:31 T24 READ DATA WITH 1 BIT ECC WORD ERROR

SEG 0083

4121	015340	042760	000100	051444	1005:	BIC	#BIT6,0BUFF(RO)	:CLEAR BIT
4122	015346				1015:			
4123								
4124	015346	004437	034072			JSR	R4,OPSTRT	:START THE OPERATION
4125	015352	004437	025100			JSR	R4,DISEEK	:SIMULATE IMPLIED SEEK
4126	015356	104004				ERROR	4	:CSI MISCOMPARE
4127	015360	104005				ERROR	5	:MR1
4128	015362	104006				ERROR	6	:MR2
4129	015364	104007				ERROR	7	:MR3
4130								
4131	015366	004437	025062			JSR	R4,DSECTR	:SIMULATE SECTOR PULSE
4132								
4133	015372	004437	025504			JSR	R4,DRSYNC	:SIMULATE HEADER PREAMBLE
4134	015376	104010				ERROR	10	:MR1 CONTENTS IN ERROR
4135	015400	104011				ERROR	11	:CSI CONTENTS IN ERROR
4136								
4137	015402	004437	026042			JSR	R4,DHDCMP	:SIMULATE HEADER SEARCH
4138	015406	104012				ERROR	12	:MR1 CONTENTS IN ERROR
4139	015410	104013				ERROR	13	:CSI IN ERROR AFTER SEARCH
4140	015412	104014				ERROR	14	:RKDCYL OR RKDA IN ERROR AFTER SEARCH
4141								
4142	015414	004437	030276			JSR	R4,DRGPSN	:GO READ GAP AND SYNC
4143	015420	104015				ERROR	15	:MR1 IN ERROR READING GAP
4144	015422	104016				ERROR	16	:CSI IN ERROR AFTER READING GAP
4145	015424	104032				ERROR	32	:MR1 IN ERROR READING SYNC
4146	015426	104033				ERROR	33	:CSI IN ERROR AFTER READING SYNC
4147								
4148	015430	012737	000001	002346		MOV	#1,ECPATX	:LOAD EXPECTED PATTERN
4149	015436	012737	004066	002350		MOV	#4066,ECPOSX	:LOAD EXPECTED POSITION
4150	015444	004437	030774			JSR	R4,DREAD	:GO SIMULATE DATA READ
4151	015450	000400				400		:NUMBER OF WORDS TO SIMULATE
4152	015452	104034				ERROR	34	:MR1 IN ERROR READING DATA OR ECC
4153	015454	104035				ERROR	35	:ECC ERROR READING DATA
4154	015456	104036				ERROR	36	:CSI ERROR AFTER READING DATA OR ECC
4155	015460	104041				ERROR	41	:ECC REG INCORRECT AFTER ECC READ
4156	015462	104042				ERROR	42	:ERR IN ECC PAT CALC.
4157	015464	104043				ERROR	43	:ERR IN ECC POS COUNT
4158								
4159								
4160	015466	012737	000020	002326		MOV	#20,BITCNT	:SET BIT COUNT FOR CORRECTION COUNT
4161	015474	012700	000005			MOV	#5,RO	:SET COUNT FOR ECCPOS PASS THRU 0
4162	015500	012701	013672			MOV	#13672,R1	:SET COUNT FOR ECCPOS TO 0 FIRST PASS
4163	015504	005037	001214			CLR	\$TMP5	:CLEAR SWITCH
4164								
4165	015510	004737	033414		3\$:	JSR	PC,RDBIT	:GO READ BIT
4166	015514	004737	032260			JSR	PC,ECCGEN	:GENERATE ECC
4167	015520	016237	000032	002214		MOV	RKECPT(R2),T.ECPT	:GET PATTERN
4168	015526	016237	000030	002212		MOV	RKECPS(R2),T.ECPS	:GET POSITION
4169	015534	005237	002350			INC	ECPOSX	:BUMP POSITION EXPECTED
4170	015540	013737	002350	002252		MOV	ECPOSX,E.ECPS	:SET FOR COMPARE AND REPORT
4171	015546	023737	002254	002214		CMP	E.ECPT,T.ECPT	:CHECK IF PATTERN CORRECT
4172	015554	001401				BEQ	1\$:YES - SKIP
4173	015556	104042				ERROR	42	
4174	015560	023737	002252	002212	1\$:	CMP	E.ECPS,T.ECPS	:CHECK IF POSITION CORRECT
4175	015566	001401				BEQ	2\$:YES - SKIP
4176	015570	104043				ERROR	43	

G07

```

4177 015572 005237 002326      2$:  INC      BITCNT      ;BUMP BIT COUNT
4178 015576 005301              DEC      R1          ;DEC COUNT TO ZERO IN ECCPOS
4179 015600 001343              BNE      .          ;NOT YET ZERO - LOOP
4180 015602 005300              DEC      R0          ;DEC PASS THRU 0 COUNT
4181 015604 001406              BEQ      4$         ;IF 0 - SKIP
4182 015606 012737 177777 002350  MOV      #-1,ECP0SX ;PRESET EXPECTED POS TO GO TO ZERO
4183 015614 012701 020000      MOV      #20000,R1  ;SET COUNT TO NEXT 0 IN ECC POS
4184 015620 000733              BR       3$         ;GO DO LOOP AGAIN
4185
4186 015622 005737 001214      4$:  TST      $TMP5      ;TEST SWITCH
4187 015626 001012              BNE      5$         ;ALREADY SET - SKIP
4188 015630 012701 010016      MOV      #10016,R1 ;SET COUNT TO FINAL POSITION
4189 015634 012700 000001      MOV      #1,R0      ;SET R0 TO RETURN AFTER ONE PASS
4190 015640 012737 177777 002350  MOV      #-1,ECP0SX ;SET EXP POS TO GO TO 0 ON NEXT BIT
4191 015646 005237 001214      INC      $TMP5      ;BUMP SWITCH
4192 015652 000716              BR       3$         ;GO DO LOOP AGAIN
4193
4194 015654 004737 033414      5$:  JSR      PC,R0BIT    ;ONE MORE BIT TO SET READY
4195 015660 004437 034154      JSR      R4,GETREG  ;GET '611 REGISTERS
4196 015664 013737 002260 002220  MOV      L,CS1,E.CS1 ;GET EXPECTED CS1
4197 015672 052737 100200 002220  BIS      #RDY!CERR,E.CS1 ;SET READY AND ERROR
4198 015700 042737 000001 002220  BIC      #GO,E.CS1   ;CLEAR GO
4199 015706 023737 002160 002220  CMP      T,CS1,E.CS1 ;CHECK IF CS1 CORRECT
4200 015714 001402              BEQ      6$         ;YES - SKIP
4201 015716 104044      ERROR   44
4202 015720 000410              BR       7$         ;
4203 015722 032737 000100 002174  6$:  BIT      #ECH,T.ER  ;TEST IF ECC HARD ERROR SET
4204 015730 001404              BEQ      7$         ;YES - SKIP
4205 015732 012737 100000 002234  MOV      #DCK,E.ER  ;SET EXPECTED ERROR REG
4206 015740 104045      ERROR   45
4207
4208 015742      7$:
4209
4210 ;*****
4211 ;*TEST 25      18 BIT NPR WRITING OF MEMORY
4212 ;*
4213 ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4214 ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4215 ;*      OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4216 ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4217 ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4218 ;*      A GOOD HEADER, 40 DATA WORDS.  VERIFY THAT THE WORDS
4219 ;*      ARE WRITTEN PROPERLY IN MEMORY.
4220 ;*
4221 ;*****
4222 015742 000004      †ST25: SCOPE
4223 015744 012737 000012 001234  MOV      #10, $TIMES ;;DO 10. ITERATIONS
4224
4225 015752 005037 002352      CLR      BSEDES     ;CLEAR BAD SEC DESIRED
4226 015756 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4227
4228 015764 005037 002344      CLR      ECCSRC     ;CLEAR ECC SOURCE
4229 015770 005037 002354      CLR      HIBITS     ;CLEAR HI ORDER BITS
4230 015774 004437 034026      JSR      R4,LOADRK  ;LOAD "L" REGISTERS
4231 016000 000000      0           ;CYLINDER 0
4232 016002      000      .BYTE 0 ;SECTOR 0
  
```

H07

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 85
 DZR6EA.P11 28-SEP-76 15:31 T25 18 BIT NPR WRITING OF MEMORY

SEQ 0095

4233	016003	000		.BYTE	0		;TRACK 0
4234	016004	050440		IBUFF			;BUFFER ADDRESS IBUFF
4235	016006	177400		-400			;WORD COUNT -400
4236	016010	010021		RDDATA!CFMT			;COMMAND RDDATA!CFMT
4237							
4238	016012	004437	033566	JSR	R4,CLRIBF		;GO CLEAR INPUT BUFFER
4239							
4240	016016	004437	033612	JSR	R4,BLDDAT		;GO BUILD DATA
4241	016022	000003		3			;PATTERN 3
4242							
4243							
4244	016024	004437	034072	JSR	R4,OPSTRT		;START THE OPERATION
4245	016030	004437	025100	JSR	R4,DISEEK		;SIMULATE IMPLIED SEEK
4246	016034	104004		ERROR	4		;CSI MISCOMPARE
4247	016036	104005		ERROR	5		;MR1
4248	016040	104006		ERROR	6		;MR2
4249	016042	104007		ERROR	7		;MR3
4250							
4251	016044	004437	025062	JSR	R4,DSECTR		;SIMULATE SECTOR PULSE
4252							
4253	016050	004437	025504	JSR	R4,DRSYNC		;SIMULATE HEADER PREAMBLE
4254	016054	104010		ERROR	10		;MR1 CONTENTS IN ERROR
4255	016056	104011		ERROR	11		;CSI CONTENTS IN ERROR
4256							
4257	016060	004437	026042	JSR	R4,DHDCMP		;SIMULATE HEADER SEARCH
4258	016064	104012		ERROR	12		;MR1 CONTENTS IN ERROR
4259	016066	104013		ERROR	13		;CSI IN ERROR AFTER SEARCH
4260	016070	104014		ERROR	14		;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4261							
4262	016072	004437	030276	JSR	R4,DRGPSN		;GO READ GAP AND SYNC
4263	016076	104015		ERROR	15		;MR1 IN ERROR READING GAP
4264	016100	104016		ERROR	16		;CSI IN ERROR AFTER READING GAP
4265	016102	104032		ERROR	32		;MR1 IN ERROR READING SYNC
4266	016104	104033		ERROR	33		;CSI IN ERROR AFTER READING SYNC
4267							
4268	016106	012737	000000	MOV	#0,ECPATX		;LOAD EXPECTED PATTERN
4269	016114	012737	005066	MOV	#5066,ECPOSX		;LOAD EXPECTED POSITION
4270	016122	004437	030774	JSR	R4,DRREAD		;GO SIMULATE DATA READ
4271	016126	000060		60			;NUMBER OF WORDS TO SIMULATE
4272	016130	104034		ERROR	34		;MR1 IN ERROR READING DATA OR ECC
4273	016132	104035		ERROR	35		;ECC ERROR READING DATA
4274	016134	104036		ERROR	36		;CSI ERROR AFTER READING DATA OR ECC
4275	016136	104041		ERROR	41		;ECC REG INCORRECT AFTER ECC READ
4276	016140	104042		ERROR	42		;ERR IN ECC PAT CALC.
4277	016142	104043		ERROR	43		;ERR IN ECC POS COUNT
4278							
4279							
4280	016144	012700	051444	MOV	#OBUFF,R0		;SET POINTER TO GOOD DATA
4281	016150	012701	050440	MOV	#IBUFF,R1		;SET POINTER TO INPUT DATA
4282							
4283	016154	012704	000012	MOV	#10,R4		;SET ERROR LIMIT COUNT
4284	016160	005037	001236	CLR	\$ESCAPE		;CLEAR ESCAPE
4285	016164	005037	001220	CLR	\$TMP7		;CLEAR ERROR REPORT SWITCH
4286	016170	012703	000040	MOV	#40,R3		;SET COMPARE LIMIT
4287	016174	005005		CLR	R5		;CLEAR WORD COUNTER
4288							

```

4289 016176 021011      1$:  CMP      (R0),(R1)      ;COMPARE DATA
4290 016200 001005      BNE      2$              ;SKIP IF NOT EQUAL
4291 016202 005303      4$:  DEC      R3              ;ELSE DEC WORD COUNT LIMIT
4292 016204 001424      BEQ      5$              ;IF 0 - SKIP TO EXIT
4293 016206 005205      INC      R5              ;BUMP WORD COUNT
4294 016210 022021      CMP      (R0)+,(R1)+    ;BUMP DATA POINTERS
4295 016212 000771      BR       1$              ;LOOP
4296
4297 016214 005304      2$:  DEC      R4              ;DEC ERROR LIMIT
4298 016216 001417      BEQ      5$              ;IF 0 - SKIP
4299
4300 016220 011037 001162    MOV      (R0), $REG0     ;GET GOOD WORD FOR REPORT
4301 016224 011137 001164    MOV      (R1), $REG1     ;GET BAD WORD
4302 016230 010537 001166    MOV      R5, $REG2      ;GET WORD NUMBER
4303
4304 016234 005737 001220    TST      $TMP7           ;DECIDE WHICH ERROR REPORT TO USE
4305 016240 001004      BNE      3$              ;SKIP IF NOT 0
4306 016242 005237 001220    INC      $TMP7           ;ELSE BUMP $TMP7
4307 016246 104037      ERROR   37              ;REPORT FIRST ERROR LINE
4308 016250 000754      BR       4$              ;SKIP
4309
4310 016252 104040      3$:  ERROR   40              ;REPORT ALL OTHER LINES
4311 016254 000752      BR       4$              ;SKIP
4312
4313 016256      5$:
4314
4315 ;*****
4316 ;*TEST 26      18 BIT READ DATA WITH NO ERROR
4317 ;*
4318 ;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4319 ;* PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4320 ;* OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4321 ;* CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4322 ;* AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4323 ;* A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
4324 ;* VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.
4325 ;*
4326 ;*****
4327 016256 000004      TST26: SCOPE
4328 016260 012737 000001 001234    MOV      #1, $TIMES     ;;DO 1 ITERATION
4329
4330 016266 005037 002352    CLR      BSEDES         ;CLEAR BAD SEC DESIRED
4331 016272 012762 100000 000000    MOV      #CLR, RKCS1(R2) ;CLEAR CONTROLLER
4332 016300 005037 002344    CLR      ECCSRC         ;CLEAR ECC SOURCE FLAG
4333 016304 005037 002344    CLR      ECCSRC         ;CLEAR ECC SOURCE
4334 016310 005037 002354    CLR      HIBITS         ;CLEAR HI ORDER BITS
4335 016314 004437 034026    JSR      R4, LOADRK      ;LOAD "L" REGISTERS
4336 016320 000000      0                      ;CYLINDER 0
4337 016322      000                  .BYTE   0                ;SECTOR 0
4338 016323      000                  .BYTE   0                ;TRACK 0
4339 016324 050440      Ibuff              ;BUFFER ADDRESS Ibuff
4340 016326 177400      -400              ;WORD COUNT -400
4341 016330 010021      Rddata!CFMT        ;COMMAND Rddata!CFMT
4342
4343 016332 004437 033566    JSR      R4, CLRIBF     ;GO CLEAR INPUT BUFFER
4344

```

J07

```

4345 016336 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
4346 016342 000004 4 ;PATTERN 4
4347
4348
4349 016344 004437 034072 JSR R4,OPSTRT ;START THE OPERATION
4350 016350 004437 025100 JSR R4,DISEEK ;SIMULATE IMPLIED SEEK
4351 016354 104004 ERROR 4 ;CSI MISCOMPARE
4352 016356 104005 ERROR 5 ;MR1 ""
4353 016360 104006 ERROR 6 ;MR2 ""
4354 016362 104007 ERROR 7 ;MR3 ""
4355
4356 016364 004437 025762 JSR R4,DSECTR ;SIMULATE SECTOR PULSE
4357
4358 016370 004437 025504 JSR R4,DRSYNC ;SIMULATE HEADER PREAMBLE
4359 016374 104010 ERROR 10 ;MR1 CONTENTS IN ERROR
4360 016376 104011 ERROR 11 ;CSI CONTENTS IN ERROR
4361
4362 016400 004437 026042 JSR R4,DHDCMP ;SIMULATE HEADER SEARCH
4363 016404 104012 ERROR 12 ;MR1 CONTENTS IN ERROR
4364 016406 104013 ERROR 13 ;CSI IN ERROR AFTER SEARCH
4365 016410 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4366
4367 016412 004437 030276 JSR R4,DRGPSN ;GO READ GAP AND SYNC
4368 016416 104015 ERROR 15 ;MR1 IN ERROR READING GAP
4369 016420 104016 ERROR 16 ;CSI IN ERROR AFTER READING GAP
4370 016422 104032 ERROR 32 ;MR1 IN ERROR READING SYNC
4371 016424 104033 ERROR 33 ;CSI IN ERROR AFTER READING SYNC
4372
4373 016426 012737 000000 002346 MOV #0,ECPATX ;LOAD EXPECTED PATTERN
4374 016434 012737 005066 002350 MOV #5066,ECPOSX ;LOAD EXPECTED POSITION
4375 016442 004437 030774 JSR R4,DREAD ;GO SIMULATE DATA READ
4376 016446 000400 400 ;NUMBER OF WORDS TO SIMULATE
4377 016450 104034 ERROR 34 ;MR1 IN ERROR READING DATA OR ECC
4378 016452 104035 ERROR 35 ;ECC ERROR READING DATA
4379 016454 104036 ERROR 36 ;CSI ERROR AFTER READING DATA OR ECC
4380 016456 104041 ERROR 41 ;ECC REG INCORRECT AFTER ECC READ
4381 016460 104042 ERROR 42 ;ERR IN ECC PAT CALC.
4382 016462 104043 ERROR 43 ;ERR IN ECC POS COUNT
4383
4384
4385 016464 012700 003100 MOV #40.*40.,R0 ;SET COUNT TO EMPTY SILO
4386
4387 016470 012762 000440 000026 1S: MOV #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
4388 016476 012762 000040 000026 MOV #DMD,RKMR1(R2)
4389 016504 005300 DEC R0 ;DEC COUNT
4390 016506 001370 BNE 1S ;LOOP UNTIL 0
4391
4392 016510 004437 034154 JSR R4,GETREG ;GET 611 REGS
4393 016514 013737 002260 002220 MOV L.CSI,E.CSI ;GET EXPECTED CSI
4394 016522 042737 000001 002220 BIC #GO,E.CSI ;CLEAR GO BIT
4395 016530 052737 000200 002220 BIS #RDY,E.CSI ;SET READY BIT
4396 016536 023737 002160 002220 CMP T.CSI,E.CSI ;CHECK IF CSI CORRECT
4397 016544 001401 BEQ 2S ;YES - SKIP
4398 016546 104056 ERROR 56 ;"CSI IN ERROR AFTER READ DATA"
4399
4400 2S: ;*****

```

K07

```

4401 ;*TEST 27 18 BIT READ DATA WITH DCK
4402 ;*
4403 ;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4404 ;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
4405 ;* OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4406 ;* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4407 ;* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
4408 ;* A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
4409 ;* A ONE BIT BURST ERROR BEYOND THE 4128TH BIT OF THE SECTOR.
4410 ;* VERIFY THE COUNTING OF THE POSITION REGISTER AND THE
4411 ;* FINAL PATTERN AND POSITION. MAKE SURE DATA CHECK AND
4412 ;* CONTROLLER ERROR SETS.
4413 ;*
4414 ;*****
4415 016550 000004 ;TEST27: SCOPE
4416 016552 012737 000001 001234 MOV #1, $TIMES ; DO 1 ITERATION
4417
4418 016560 005037 002352 CLR BSEDES ; CLEAR BAD SEC DESIRED
4419 016564 012762 100000 000000 MOV #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
4420
4421 016572 005037 002354 CLR HIBITS ; CLEAR HI ORDER BITS
4422 016576 004437 034026 JSR R4, LOADRK ; LOAD "L" REGISTERS
4423 016602 000000 0 ; CYLINDER 0
4424 016604 000 ; SECTOR 0
4425 016605 000 ; TRACK 0
4426 016606 050440 Ibuff ; BUFFER ADDRESS Ibuff
4427 016610 177400 -400 ; WORD COUNT -400
4428 016612 010021 Rddata!CFMT ; COMMAND Rddata!CFMT
4429
4430 016614 004437 033566 JSR R4, CLRIBF ; GO CLEAR INPUT BUFFER
4431
4432 016620 004437 033612 JSR R4, BLDDAT ; GO BUILD DATA
4433 016624 000007 7 ; PATTERN 7
4434
4435
4436 016626 012700 000734 MOV #734, R0 ; SET INDEX INTO BUFFER
4437 016632 012737 006030 052444 MOV #6030, Obuff+1000 ; STORE ECC WORDS
4438 016640 012737 045070 052446 MOV #45070, Obuff+1002
4439 016646 012737 000001 002344 MOV #1, ECCSRC ; SET SOURCE TO INDICATE ECC IN BUFFER
4440 016654 032760 000100 051444 BIT #BIT6, Obuff(R0) ; TEST IF BIT SET
4441 016662 001004 BNE 100$ ; YES - SKIP
4442 016664 052760 000100 051444 BIS #BIT6, Obuff(R0) ; ELSE SET BIT
4443 016672 000403 BR 101$
4444 016674 042760 000100 051444 100$: BIC #BIT6, Obuff(R0) ; CLEAR BIT
4445 016702 101$:
4446
4447 016702 004437 034072 JSR R4, OPSTRT ; START THE OPERATION
4448 016706 004437 025100 JSR R4, DISEEK ; SIMULATE IMPLIED SEEK
4449 016712 104004 ERROR 4 ; CSI MISCOMPARE
4450 016714 104005 ERROR 5 ; MR1
4451 016716 104006 ERROR 6 ; MR2
4452 016720 104007 ERROR 7 ; MR3
4453
4454 016722 004437 025062 JSR R4, DSECTR ; SIMULATE SECTOR PULSE
4455
4456 016726 004437 025504 JSR R4, DRSYNC ; SIMULATE HEADER PREAMBLE

```


4457	016732	104010			ERROR	10		;MR1 CONTENTS IN ERROR
4458	016734	104011			ERROR	11		;CSI CONTENTS IN ERROR
4459								
4460	016736	004437	026042		JSR	R4, DHDCMP		;SIMULATE HEADER SEARCH
4461	016742	104012			ERROR	12		;MR1 CONTENTS IN ERROR
4462	016744	104013			ERROR	13		;CSI IN ERROR AFTER SEARCH
4463	016746	104014			ERROR	14		;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4464								
4465	016750	004437	030276		JSR	R4, DRGPSN		;GO READ GAP AND SYNC
4466	016754	104015			ERROR	15		;MR1 IN ERROR READING GAP
4467	016756	104016			ERROR	16		;CSI IN ERROR AFTER READING GAP
4468	016760	104032			ERROR	32		;MR1 IN ERROR READING SYNC
4469	016762	104033			ERROR	33		;CSI IN ERROR AFTER READING SYNC
4470								
4471	016764	012737	000001	002346	MOV	#1, ECPATX		;LOAD EXPECTED PATTERN
4472	016772	012737	005066	002350	MOV	#5066, ECPOSX		;LOAD EXPECTED POSITION
4473	017000	004437	030774		JSR	R4, DRDREAD		;GO SIMULATE DATA READ
4474	017004	000400			400			;NUMBER OF WORDS TO SIMULATE
4475	017006	104034			ERROR	34		;MR1 IN ERROR READING DATA OR ECC
4476	017010	104035			ERROR	35		;ECC ERROR READING DATA
4477	017012	104036			ERROR	36		;CSI ERROR AFTER READING DATA OR ECC
4478	017014	104041			ERROR	41		;ECC REG INCORRECT AFTER ECC READ
4479	017016	104042			ERROR	42		;ERR IN ECC PAT CALC.
4480	017020	104043			ERROR	43		;ERR IN ECC POS COUNT
4481								
4482								
4483	017022	012737	000022	002326	MOV	#22, BITCNT		;SET BIT COUNT FOR CORRECTION COUNT
4484	017030	012700	000005		MOV	#5, R0		;SET COUNT FOR ECCPOS PASS THRU 0
4485	017034	012701	012672		MOV	#12672, R1		;SET COUNT FOR ECCPOS TO 0 FIRST PASS
4486	017040	005037	001214		CLR	\$TMP5		;CLEAR SWITCH
4487								
4488	017044	004737	033414		3\$: JSR	PC, RDBIT		;GO READ BIT
4489	017050	004737	032260		JSR	PC, ECCGEN		;GENERATE ECC
4490	017054	016237	000032	002214	MOV	RKECPT(R2), T.ECPT		;GET PATTERN
4491	017062	016237	000030	002212	MOV	RKECPS(R2), T.ECPS		;GET POSITION
4492	017070	005237	002350		INC	ECPOSX		;BUMP POSITION EXPECTED
4493	017074	013737	002350	002252	MOV	ECPOSX, E.ECFS		;SET FOR COMPARE AND REPORT
4494	017102	023737	002254	002214	CMP	E.ECPT, T.ECPT		;CHECK IF PATTERN CORRECT
4495	017110	001401			BEQ	1\$;YES - SKIP
4496	017112	104042			ERROR	42		
4497	017114	023737	002252	002212	1\$: CMP	E.ECPS, T.ECPS		;CHECK IF POSITION CORRECT
4498	017122	001401			BEQ	2\$;YES - SKIP
4499	017124	104043			ERROR	43		
4500	017126	005237	002326		2\$: INC	BITCNT		;BUMP BIT COUNT
4501	017132	005301			DEC	R1		;DEC COUNT TO ZERO IN ECCPOS
4502	017134	001343			BNE	3\$;NOT YET ZERO - LOOP
4503	017136	005300			DEC	R0		;DEC PASS THRU 0 COUNT
4504	017140	001406			BEQ	4\$;IF 0 - SKIP
4505	017142	012737	177777	002350	MOV	#-1, ECPOSX		;PRESET EXPECTED POS TO GO TO ZERO
4506	017150	012701	020000		MOV	#20000, R1		;SET COUNT TO NEXT 0 IN ECC POS
4507	017154	000733			BR	3\$;GO DO LOOP AGAIN
4508								
4509	017156	005737	001214		4\$: TST	\$TMP5		;TEST SWITCH
4510	017162	001012			BNE	5\$;ALREADY SET - SKIP
4511	017164	012701	010272		MOV	#10272, R1		;SET COUNT TO FINAL POSITION
4512	017170	012700	000001		MOV	#1, R0		;SET R0 TO RETURN AFTER ONE PASS

M07

```

4513 017174 012737 177777 002350 MOV #1,ECPOSX ;SET EXP POS TO GO TO 0 ON NEXT BIT
4514 017202 005237 001214 INC $TMP5 ;BUMP SWITCH
4515 017206 000716 BR 3$ ;GO DO LOOP AGAIN
4516
4517 017210 004737 033414 5$: JSR PC,RDBIT ;ONE MORE BIT TO SET READY
4518 017214 004437 034154 JSR R4,GETREG ;GET '611 REGISTERS
4519 017220 013737 002260 002220 MOV L.CS1,E.CS1 ;GET EXPECTED CS1
4520 017226 052737 100200 002220 BIS #RDY!CERR,E.CS1 ;SET READY AND ERROR
4521 017234 042737 000001 002220 BIC #GO,E.CS1 ;CLEAR GO
4522 017242 023737 002160 002220 CMP T.CS1,E.CS1 ;CHECK IF CS1 CORRECT
4523 017250 001402 BEQ 6$ ;YES - SKIP
4524 017252 104044 ERROR 44
4525 017254 000410 BR 7$
4526 017256 032737 000100 002174 6$: BIT #ECH,T.ER ;TEST IF ECC HARD ERROR SET
4527 017264 001404 BEQ 7$ ;NO SKIP
4528 017266 012737 100000 002234 MOV #DCK,E.ER ;SET EXPECTED ERROR REG
4529 017274 104045 ERROR 45
4530
4531 017276 7$:
4532
4533 *****
4534 *TEST 30 18 BIT READ DATA WITH ECH
4535 *
4536 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4537 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
4538 * OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4539 * CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4540 * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
4541 * A GOOD HEADER, 400 DATA WORDS AND AND ECC INDICATING
4542 * A 12 BIT BURST ERROR. VERIFY THE COUNTING OF THE
4543 * POSITION REGISTER. MAKE SURE DATE CHECK, ECC HARD,
4544 * AND CONTROLLER ERROR SETS.
4545 *****
4546
4547 017276 000004 ST30: SCOPE
4548 017300 012737 000001 001234 MOV #1,$TIMES ;;DO 1 ITERATION
4549
4550 017306 005037 002352 CLR BSEDES ;CLEAR BAD SEC DESIRED
4551 017312 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4552
4553 017320 005037 002354 CLR HIBITS ;CLEAR HI ORDER BITS
4554 017324 004437 034026 JSR R4,LOADRK ;LOAD "L" REGISTERS
4555 017330 000000 0 ;CYLINDER 0
4556 017332 000 .BYTE 0 ;SECTOR 0
4557 017333 000 .BYTE 0 ;TRACK 0
4558 017334 050440 Ibuff ;BUFFER ADDRESS Ibuff
4559 017336 177400 -400 ;WORD COUNT -400
4560 017340 010021 RDDATA!CFMT ;COMMAND RDDATA!CFMT
4561
4562 017342 004437 033566 JSR R4,CLRIBF ;GO CLEAR INPUT BUFFER
4563
4564 017346 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
4565 017352 000005 5 ;PATTERN 5
4566
4567
4568 017354 012700 000734 MOV #734,R0 ;SET INDEX INTO BUFFER
    
```

NO7

```

4569 017360 012737 004020 052444      MOV      #4020,OBUFF+1000      ;STORE ECC WORDS
4570 017366 012737 071720 052446      MOV      #71720,OBUFF+1002
4571 017374 012737 000001 002344      MOV      #1,ECCSRC          ;SET SOURCE TO INDICATE ECC IN BUFFER
4572 017402 042760 005000 051444      BIC      #BIT9!BIT11,OBUFF(RO) ;CREATE 11 BIT BURST ERROR
4573 017410 052760 050100 051444      BIS      #BIT6!BIT12!BIT14,OBUFF(RO)
4574 017416 062700 000002      ADD      #2,RO              ;BUMP TO NEXT WORD
4575 017422 052760 000001 051444      BIS      #BIT0,OBUFF(RO)
4576 017430 042760 000002 051444      BIC      #BIT1,OBUFF(RO) ;MAKE IT A 12 BIT BURST ERROR
4577
4578 017436 004437 034072      JSR      R4,OPSTRT          ;START THE OPERATION
4579 017442 004437 025100      JSR      R4,DISEEK         ;SIMULATE IMPLIED SEEK
4580 017446 104004      ERROR   4                  ;CS1 MISCOMPARE
4581 017450 104005      ERROR   5                  ;MR1
4582 017452 104006      ERROR   6                  ;MR2
4583 017454 104007      ERROR   7                  ;MR3
4584
4585 017456 004437 025062      JSR      R4,DSECTR         ;SIMULATE SECTOR PULSE
4586
4587 017462 004437 025504      JSR      R4,DRSYNC        ;SIMULATE HEADER PREAMBLE
4588 017466 104010      ERROR   10                 ;MR1 CONTENTS IN ERROR
4589 017470 104011      ERROR   11                 ;CS1 CONTENTS IN ERROR
4590
4591 017472 004437 026042      JSR      R4,DHDCMP        ;SIMULATE HEADER SEARCH
4592 017476 104012      ERROR   12                 ;MR1 CONTENTS IN ERROR
4593 017500 104013      ERROR   13                 ;CS1 IN ERROR AFTER SEARCH
4594 017502 104014      ERROR   14                 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4595
4596 017504 004437 030276      JSR      R4,DRGPSN        ;GO READ GAP AND SYNC
4597 017510 104015      ERROR   15                 ;MR1 IN ERROR READING GAP
4598 017512 104016      ERROR   16                 ;CS1 IN ERROR AFTER READING GAP
4599 017514 104032      ERROR   32                 ;MR1 IN ERROR READING SYNC
4600 017516 104033      ERROR   33                 ;CS1 IN ERROR AFTER READING SYNC
4601
4602 017520 012737 000001 002346      MOV      #1,ECPATX        ;LOAD EXPECTED PATTERN
4603 017526 012737 005066 002350      MOV      #5066,ECPOSX     ;LOAD EXPECTED POSITION
4604 017534 004437 030774      JSR      R4,DREAD         ;GO SIMULATE DATA READ
4605 017540 000400      400                      ;NUMBER OF WORDS TO SIMULATE
4606 017542 104034      ERROR   34                 ;MR1 IN ERROR READING DATA OR ECC
4607 017544 104035      ERROR   35                 ;ECC ERROR READING DATA
4608 017546 104036      ERROR   36                 ;CS1 ERROR AFTER READING DATA OR ECC
4609 017550 104041      ERROR   41                 ;ECC REG INCORRECT AFTER ECC READ
4610 017552 104042      ERROR   42                 ;ERR IN ECC PAT CALC.
4611 017554 104043      ERROR   43                 ;ERR IN ECC POS COUNT
4612
4613
4614 017556 012737 000022 002326      MOV      #22,BITCNT       ;SET BIT COUNT FOR CORRECTION COUNT
4615 017564 012700 000005      MOV      #5,RO           ;SET COUNT FOR ECCPOS PASS THRU 0
4616 017570 012701 012672      MOV      #12672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
4617 017574 005037 001214      CLR      $TMP5           ;CLEAR SWITCH
4618
4619 017600 004737 033414      3$: JSR      PC,RDBIT        ;GO READ BIT
4620 017604 004737 032260      JSR      PC,ECCGEN        ;GENERATE ECC
4621 017610 016237 000032 002214      MOV      RKECPT(R2),T.ECPT ;GET PATTERN
4622 017616 016237 000030 002212      MOV      RKECPS(R2),T.ECPS ;GET POSITION
4623 017624 005237 002350      INC      ECPOSX          ;BUMP POSITION EXPECTED
4624 017630 013737 002350 002252      MOV      ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
    
```

```

4625 017636 023737 002254 002214      CMP      E.ECPT,T.ECPT      :CHECK IF PATTERN CORRECT
4626 017644 001401                    BEQ      15                :YES - SKIP
4627 017646 104042                    ERROR    42
4628 017650 023737 002252 002212 15:    CMP      E.ECPS,T.ECPS    :CHECK IF POSITION CORRECT
4629 017656 001401                    BEQ      25                :YES - SKIP
4630 017660 104043                    ERROR    43
4631 017662 005237 002326      25:    INC      BITCNT          :BUMP BIT COUNT
4632 017666 005301                    DEC      R1              :DEC COUNT TO ZERO IN ECCPOS
4633 017670 001343                    BNE     35                :NOT YET ZERO - LOOP
4634 017672 005300                    DEC      R0              :DEC PASS THRU 0 COUNT
4635 017674 001406                    BEQ      45                :IF 0 - SKIP
4636 017676 012737 177777 002350      MOV     #-1,ECP0SX       :PRESET EXPECTED POS TO GO TO ZERO
4637 017704 012701 020000      MOV     #20000,R1        :SET COUNT TO NEXT 0 IN ECC POS
4638 017710 000733                    BR      35                :GO DO LOOP AGAIN
4639
4640 017712 005737 001214      45:    TST     $TMP5           :TEST SWITCH
4641 017716 001012                    BNE     55                :ALREADY SET - SKIP
4642 017720 012701 011041      MOV     #11041,R1        :SET COUNT TO FINAL POSITION
4643 017724 012700 000001      MOV     #1,R0            :SET R0 TO RETURN AFTER ONE PASS
4644 017730 012737 177777 002350      MOV     #-1,ECP0SX       :SET EXP POS TO GO TO 0 ON NEXT BIT
4645 017736 005237 001214      INC     $TMP5           :BUMP SWITCH
4646 017742 000716                    BR      35                :GO DO LOOP AGAIN
4647
4648 017744 004737 033414      55:    JSR     PC,RDBIT        :ONE MORE BIT TO SET READY
4649 017750 004437 034154      JSR     R4,GETREG        :GET '611 REGISTERS
4650 017754 013737 002260 002220      MOV     L.CS1,E.CS1     :GET EXPECTED CS1
4651 017762 052737 100200 002220      BIS     #RDY!CERR,E.CS1 :SET READY AND ERROR
4652 017770 042737 000001 002220      BIC     #GO,E.CS1       :CLEAR GO
4653 017776 023737 002160 002220      CMP     T.CS1,E.CS1     :CHECK IF CS1 CORRECT
4654 020004 001402                    BEQ     65                :YES - SKIP
4655 020006 104044                    ERROR    44
4656 020010 000410                    BR      75
4657 020012 032737 000100 002174 65:    BIT     #ECH,T.ER        :TEST IF ECC HARD ERROR SET
4658 020020 001004                    BNE     75                :YES - SKIP
4659 020022 012737 100100 002234      MOV     #DCK!ECH,E.ER    :SET EXPECTED ERROR REG
4660 020030 104045                    ERROR    45

```

```

4661
4662 020032      75:
4663
4664      .SBTTL  **SPECIAL READ TESTS
4665
4666      ;*****
4667      ;TEST 31      PARTIAL SECTOR READ
4668      ;
4669      ;      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4670      ;      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4671      ;      OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4672      ;      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4673      ;      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4674      ;      A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
4675      ;      MAKE SURE ONLY 103 WORDS GOT TRANSFERRED TO MEMORY.
4676      ;
4677      ;*****
4678 020032 000004      †ST31:  SCOPE
4679 020034 012737 000001 001234      MOV     #1,$TIMES      ;;DO 1 ITERATION
4680

```

4681	020042	005037	002352		CLR	BSEDES	:CLEAR BAD SEC DESIRED
4682	020046	012762	100000	000000	MOV	#CCLR,RKCSI,R2	:CLEAR CONTROLLER
4683							
4684	020054	005037	002344		CLR	ECCSRC	:CLEAR ECC SOURCE
4685	020060	005037	002354		CLR	HIBITS	:CLEAR HI ORDER BITS
4686	020064	004437	034026		JSR	R4,LOADRK	:LOAD "L" REGISTERS
4687	020070	000000				0	:CYLINDER 0
4688	020072	000			.BYTE	0	:SECTOR 0
4689	020073	000			.BYTE	0	:TRACK 0
4690	020074	050440			IBUFF		:BUFFER ADDRESS IBUFF
4691	020076	177675			-103		:WORD COUNT -103
4692	020100	000021			RDDATA		:COMMAND RDDATA
4693							
4694	020102	004437	033566		JSR	R4,CLRIBF	:GO CLEAR INPLT BUFFER
4695							
4696	020106	004437	033612		JSR	R4,BLDDAT	:GO BUILD DATA
4697	020112	000006			6		:PATTERN 6
4698							
4699							
4700	020114	004437	034072		JSR	R4,OPSTRT	:START THE OPERATION
4701	020120	004437	025100		JSR	R4,DISEEK	:SIMULATE IMPLIED SEEK
4702	020124	104004			ERROR	4	:CSI MISCOMPARE
4703	020126	104005			ERROR	5	:MR1
4704	020130	104006			ERROR	6	:MR2
4705	020132	104007			ERROR	7	:MR3
4706							
4707	020134	004437	025062		JSR	R4,DSECTR	:SIMULATE SECTOR PULSE
4708							
4709	020140	004437	025504		JSR	R4,DRSYNC	:SIMULATE HEADER PREAMBLE
4710	020144	104010			ERROR	10	:MR1 CONTENTS IN ERROR
4711	020146	104011			ERROR	11	:CSI CONTENTS IN ERROR
4712							
4713	020150	004437	026042		JSR	R4,DHDCMP	:SIMULATE HEADER SEARCH
4714	020154	104012			ERROR	12	:MR1 CONTENTS IN ERROR
4715	020156	104013			ERROR	13	:CSI IN ERROR AFTER SEARCH
4716	020160	104014			ERROR	14	:RKDCYL OR RKDA IN ERROR AFTER SEARCH
4717							
4718	020162	004437	030276		JSR	R4,DRGPSN	:GO READ GAP AND SYNC
4719	020166	104015			ERROR	15	:MR1 IN ERROR READING GAP
4720	020170	104016			ERROR	16	:CSI IN ERROR AFTER READING GAP
4721	020172	104032			ERROR	32	:MR1 IN ERROR READING SYNC
4722	020174	104033			ERROR	33	:CSI IN ERROR AFTER READING SYNC
4723							
4724	020176	012737	000000	002346	MOV	#0,ECPATX	:LOAD EXPECTED PATTERN
4725	020204	012737	004066	002350	MOV	#4066,ECPOSX	:LOAD EXPECTED POSITION
4726	020212	004437	030774		JSR	R4,DRREAD	:GO SIMULATE DATA READ
4727	020216	000400			400		:NUMBER OF WORDS TO SIMULATE
4728	020220	104034			ERROR	34	:MR1 IN ERROR READING DATA OR ECC
4729	020222	104035			ERROR	35	:ECC ERROR READING DATA
4730	020224	104036			ERROR	36	:CSI ERROR AFTER READING DATA OR ECC
4731	020226	104041			ERROR	41	:ECC REG INCORRECT AFTER ECC READ
4732	020230	104042			ERROR	42	:ERR IN ECC PAT CALC.
4733	020232	104043			ERROR	43	:ERR IN ECC POS COUNT
4734							
4735							
4736	020234	013700	002164		MOV	T.BA,R0	:GET ACTUAL BA

```

4737 020240 162700 050440 SUB #IBUFF,R0 ;SUB START VALUE OF BA
4738 020244 022700 000206 CMP #206,R0 ;TEST IF CORRECT NUM OF WORDS XFER
4739 020250 001404 BEQ 48 ;YES - SKIP
4740 020252 012737 050646 002224 MOV #IBUFF+206,E.BA ;SET EXPECTED BA
4741 020260 104046 ERROR 46 ;REPORT XFER LENGTH ERROR
4742
4743 020262 012701 000014 48: MOV #3,#4,R1 ;SET COUNT TO END OF OPERATION
4744 020266 012762 000440 000026 38: MOV #0MD!MCLK,RKMR1(R2) ;RUN THE CLOCK
4745 020274 012762 000040 000026 MOV #0MD,RKMR1(R2)
4746 020302 005301 DEC R1 ;DEC THE COUNT
4747 020304 001370 BNE 38 ;LOOP UNTIL ZERO
4748
4749 020306 004437 034154 18: JSR R4,GETREG ;GET 611 REGS
4750 020312 013737 002260 002220 MOV L.CS1,E.CS1 ;GET EXPECTED CS1
4751 020320 042737 000001 002220 BIC #GO,E.CS1 ;CLEAR GO BIT
4752 020326 052737 000200 002220 BIS #RDY,E.CS1 ;SET READY BIT
4753 020334 023737 002160 002220 CMP T.CS1,E.CS1 ;CHECK IF CS1 CORRECT
4754 020342 001401 BEQ 28 ;YES - SKIP
4755 020344 104056 ERROR 56 ;"CS1 IN ERROR AFTER READ DATA"
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772

```

```

*****
:TEST 32 SILO UNLOAD BEFORE HEADER ERROR
*****
:
: CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
: PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
: OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
: CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
: AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
: A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
: AFTER ECC SIMULATE A SECTOR PULSE AND A BAD SECTOR ERROR.
: MAKE SURE THE ENTIRE PREVIOUS SECTOR GETS LOADED INTO
: MEMORY BEFORE BAD SECTOR ERROR AND CONTROLLER ERROR SETS.
*****

```

```

4773 020346 000004 TEST32: SCOPE
4774 020350 012737 000001 001234 MOV #1,STIMES ;;DO 1 ITERATION
4775
4776 020356 012762 100000 000000 MOV #0CLR,RKCS1(R2) ;CLEAR CONTROLLER
4777
4778 020364 005037 002352 CLR BSEDES ;CLEAR BSE DESIRED
4779 020370 005037 002344 CLR ECCSRC ;CLEAR ECC SOURCE
4780 020374 005037 002354 CLR HIBITS ;CLEAR HI ORDER BITS
4781 020400 004437 034026 JSR R4,LOADRK ;LOAD "L" REGISTERS
4782 020404 000000 0 ;CYLINDER 0
4783 020406 000 ;SECTOR 0
4784 020407 000 ;TRACK 0
4785 020410 050440 IBUFF ;BUFFER ADDRESS IBUFF
4786 020412 177377 -401 ;WORD COUNT -401
4787 020414 000021 RDDATA ;COMMAND RDDATA
4788
4789 020416 004437 033566 JSR R4,CLRIBF ;GO CLEAR INPUT BUFFER
4790
4791 020422 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
4792 020426 000002 2 ;PATTERN 2

```

E08

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MC-11-DZ66-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 95
 DZ66A.P11 28-SEP-76 15:31 T32 SILO UNLOAD BEFORE HEADER ERROR

SEG 0095

4793									
4794									
4795	020430	004437	034072		JSR	R4,JPSTRT		:START THE OPERATION	
4796	020434	004437	025100		JSR	R4,DISEEK		:SIMULATE IMPLIED SEEK	
4797	020441	104004			ERROR	4		:CSI MISCOMPARE	
4798	020442	104005			ERROR	5		:MR1	:
4799	020444	104006			ERROR	6		:MR2	:
4800	020446	104007			ERROR	7		:MR3	:
4801									
4802	020450	004437	025062		JSR	R4,DSECTR		:SIMULATE SECTOR PULSE	
4803									
4804	020454	004437	025504		JSR	R4,DRSYNC		:SIMULATE HEADER PREAMBLE	
4805	020460	104010			ERROR	10		:MR1 CONTENTS IN ERROR	
4806	020462	104011			ERROR	11		:CSI CONTENTS IN ERROR	
4807									
4808	020464	004437	026042		JSR	R4,DHDCMP		:SIMULATE HEADER SEARCH	
4809	020470	104012			ERROR	12		:MR1 CONTENTS IN ERROR	
4810	020472	104013			ERROR	13		:CSI IN ERROR AFTER SEARCH	
4811	020474	104014			ERROR	14		:RKDCYL OR RKDA IN ERROR AFTER SEARCH	
4812									
4813	020476	004437	030276		JSR	R4,DRGPSN		:GO READ GAP AND SYNC	
4814	020502	104015			ERROR	15		:MR1 IN ERROR READING GAP	
4815	020504	104016			ERROR	16		:CSI IN ERROR AFTER READING GAP	
4816	020506	104032			ERROR	32		:MR1 IN ERROR READING SYNC	
4817	020510	104033			ERROR	33		:CSI IN ERROR AFTER READING SYNC	
4818									
4819	020512	012737	000000	002346	MOV	#0,ECPATX		:LOAD EXPECTED PATTERN	
4820	020520	012737	004066	002350	MOV	#4066,ECPOSX		:LOAD EXPECTED POSITION	
4821	020526	004437	030774		JSR	R4,DRDAD		:DO SIMULATE DATA REAC	
4822	020532	000400			400			:NUMBER OF WORDS TO SIMULATE	
4823	020534	104034			ERROR	34		:MR1 IN ERROR READING DATA OR ECC	
4824	020536	104035			ERROR	35		:ECC ERROR READING DATA	
4825	020540	104036			ERROR	36		:CSI ERROR AFTER READING DATA OR ECC	
4826	020542	104041			ERROR	41		:ECC REG INCORRECT AFTER ECC READ	
4827	020544	104042			ERROR	42		:ERR IN ECC PAT CALC.	
4828	020546	104043			ERROR	43		:ERR IN ECC POS COUNT	
4829									
4830									
4831	020550	012701	000040		MOV	#8,*4,R1		:SET COUNT TO END OF SECTOR	
4832	020554	012762	000440	000026	MOV	#DMD!MCLK,RKMR1(R2)		:RUN THE CLOCK	
4833	020562	012762	000040	000026	MOV	#DMD,RKMR1(R2)			
4834	020570	005301			DEC	R1		:DEC THE COUNT	
4835	020572	001370			BNE	6\$:LOOP UNTIL ZERO	
4836									
4837	020574	052737	100000	002352	BIS	#BIT15,BSEDES		:SET FOR BAD SECTOR ERROR	
4838									
4839	020602	004437	025062		JSR	R4,DSECTR		:ISSUE SECTOR PULSE	
4840									
4841	020606	004437	025504		JSR	R4,DRSYNC		:READ SYNC	
4842	020612	104010			ERROR	10		:MR1 CONTENTS IN ERROR	
4843	020614	104011			ERROR	11		:CSI CONTENTS IN ERROR	
4844									
4845	020616	004437	026042		JSR	R4,DHDCMP		:DO HEADER COMPARE WITH BSE ERROR	
4846	020622	104012			ERROR	12		:MR1 CONTENTS IN ERROR	
4847	020624	104013			ERROR	13		:CSI IN ERROR AFTER SEARCH	
4848	020626	000411			BR	1\$:EXPECTED RETURN (NO DA INCREMENT)	

F08

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 96
 DZR6EA.F11 28-SEP-76 15:31 T32 SILO UNLOAD BEFORE HEADER ERROR

SEQ 0096

```

4849
4850 : IF THIS CODE IS EXECUTED, THE CONTROLLER DID NOT INHIBIT
4851 : THE ADDRESS INCREMENT BECAUSE OF THE BSE ERROR.
4852
4853 020630 005037 002352 CLR BSEDES ;CLEAR BSE DESIRED SWITCH
4854 020634 013737 002274 002240 MOV L.DCYL,E.DCYL ;SET EXPECTED CYL
4855 020642 013737 002266 002226 MOV L.DA,E.DA ;SET EXPECTED SEC/TRACK
4856 020650 104047 ERROR 47 ;UNEXPECTED ADDRESS INCREMENT
4857
4858 020652 016237 000014 002174 1$: MOV RKER(R2),T.ER ;GET ERROR REG
4859 020660 032737 000200 002174 BIT #BSE,T.ER ;TEST IF BSE SET
4860 020666 001005 BNE 2$ ;YES - SKIP
4861 020670 012737 000200 002234 MOV #BSE,E.ER ;SET EXPECTED ERROR REG
4862 020676 104051 ERROR 51 ;"BSE DID NOT SET"
4863 020700 000453 BR TST33 ;GO TO NEXT TEST
4864
4865 020702 004437 034154 2$: JSR R4,GETREG ;GET 611 REGS
4866 020706 013737 002260 002220 MOV L.CS1,E.CS1 ;SET EXPECTED CS1
4867 020714 023737 002220 002160 CMP E.CS1,T.CS1 ;CHECK IF CS1 OK
4868 020722 001402 BEQ 3$ ;YES - SKIP
4869 020724 104056 ERROR 56 ;"CS1 ERROR AFTER WRITE CHECK"
4870 020726 003440 BR TST33 ;GO TO NEXT TEST
4871
4872 020730 012701 010540 3$: MOV #278,*16.,R1 ;SET COUNT TO NEXT ECC FIELD
4873 020734 005037 002320 CLR PR.BIT ;CLEAR PRESENT AND PREVIOUS BITS
4874 020740 005037 002322 CLR M1.BIT
4875 020744 004737 033414 4$: JSR PC,RDBIT ;SIMULATE READ BIT
4876 020750 005301 DEC R1 ;DEC COUNT
4877 020752 001374 BNE 4$ ;LOOP UNTIL ZERO
4878
4879 020754 004437 034154 JSR R4,GETREG ;GET 611 REGS
4880 020760 042737 000001 002220 BIC #G0,E.CS1 ;SET UP EXPECTED CS1
4881 020766 052737 100200 002220 BIS #RDY!CERR,E.CS1
4882 020774 023737 002220 002160 CMP E.CS1,T.CS1 ;CHECK IF CS1 OK
4883 021002 001402 BEQ 7$ ;YES - SKIP
4884 021004 104056 ERROR 56 ;"CS1 ERROR AFTER READ DATA"
4885 021006 000410 BR TST33 ;GO TO NEXT TEST
4886
4887 021010 012737 051440 002224 7$: MOV #IBUFF+1000,E.BA ;SET EXPECTED BA
4888 021016 023737 002224 002164 CMP E.BA,T.BA ;TEST IF BA CORRECT
4889 021024 001401 BEQ 5$ ;YES - SKIP
4890 021026 104052 ERROR 52 ;BUS ADDRESS INCORRECT
4891
4892 021030 5$:
4893
4894 .SBTTL **WRITE CHECK TESTS
4895
4896 ;*****
4897 ;TEST 33 WRITE CHECK OF 400 WORDS
4898 ;*
4899 ;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4900 ;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4901 ;* CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4902 ;* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4903 ;* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
4904 ;* A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
  
```



```

4905          : *      AND A GOOD 32 BIT ECC.  MAKE SURE NO ERRORS SET.
4906          : *
4907          : * *****
4908 021030 000004          †ST33: SCOPE
4909 021032 012737 000012 001234      MOV      #10, $TIMES      ; DO 10. ITERATIONS
4910 021040 005037 002354          CLR      HIBITS          ; CLEAR HI ORDER BITS
4911 021044 005037 002352          CLR      BSEDES          ; CLEAR BSE DESIRED
4912
4913 021050 005037 002344          CLR      ECCSRC          ; CLEAR ECC SOURCE FLAG
4914 021054 005037 002344          CLR      ECCSRC          ; CLEAR ECC SOURCE
4915 021060 004437 034026          JSR      R4, LOADRK      ; LOAD "L" REGISTERS
4916 021064 000000          0          ; CYLINDER 0
4917 021066          000          .BYTE 0          ; SECTOR 0
4918 021067          000          .BYTE 0          ; TRACK 0
4919 021070 050440          Ibuff          ; BUFFER ADDRESS Ibuff
4920 021072 177400          -400          ; WORD COUNT -400
4921 021074 000031          WRTCHK          ; COMMAND WRTCHK
4922
4923 021076 004437 033566          JSR      R4, CLRIBF      ; GO CLEAR INPUT BUFFER
4924
4925 021102 004437 033612          JSR      R4, BLDDAT      ; GO BUILD DATA
4926 021106 000004          4          ; PATTERN 4
4927
4928 021110 012700 000400          MOV      #400, R0        ; SET A COUNT FOR BUFFER SIZE
4929 021114 012701 050440          MOV      #IBUFF, R1      ; SET ADDRESS OF Ibuff
4930 021120 012703 051444          MOV      #OBUFF, R3      ; SET ADDRESS OF OBUFF
4931 021124 012321          75$: MOV      (R3)+, (R1)+    ; MOV PATTERN TO Ibuff
4932 021126 005300          DEC      R0              ; DEC COUNT
4933 021130 001375          BNE     75$              ; LOOP UNTIL PATTERN IS MOVED
4934
4935 021132 012762 100000 000000 110$: MOV      #CLR, RKCS1(R2)   ; CLEAR CONTROLLER
4936 021140 004437 034072          JSR      R4, OPSTRT      ; START THE OPERATION
4937 021144 004437 025100          JSR      R4, DISEEK      ; SIMULATE IMPLIED SEEK
4938 021150 104004          ERROR 4          ; CS1 MISCOMPARE
4939 021152 104005          ERROR 5          ; MR1 "
4940 021154 104006          ERROR 6          ; MR2 "
4941 021156 104007          ERROR 7          ; MR3 "
4942
4943 021160 004437 025062          JSR      R4, DSECTR      ; SIMULATE SECTOR PULSE
4944
4945 021164 004437 025504          JSR      R4, DRSYNC      ; SIMULATE HEADER PREAMBLE
4946 021170 104010          ERROR 10         ; MR1 CONTENTS IN ERROR
4947 021172 104011          ERROR 11         ; CS1 CONTENTS IN ERROR
4948
4949 021174 004437 026042          JSR      R4, DHDCMP      ; SIMULATE HEADER SEARCH
4950 021200 104012          ERROR 12         ; MR1 CONTENTS IN ERROR
4951 021202 104013          ERROR 13         ; CS1 IN ERROR AFTER SEARCH
4952 021204 104014          ERROR 14         ; RKDCYL OR RKDA IN ERROR AFTER SEARCH
4953
4954 021206 004437 030276          JSR      R4, DRGPSN      ; GO READ GAP AND SYNC
4955 021212 104015          ERROR 15         ; MR1 IN ERROR READING GAP
4956 021214 104016          ERROR 16         ; CS1 IN ERROR AFTER READING GAP
4957 021216 104032          ERROR 32         ; MR1 IN ERROR READING SYNC
4958 021220 104033          ERROR 33         ; CS1 IN ERROR AFTER READING SYNC
4959
4960 021222 012737 000000 002346      MOV      #0, ECPATX      ; LOAD EXPECTED PATTERN

```

H08

```

4961 021230 012737 004066 002350      MOV      #4066,ECPOSX      ;LOAD EXPECTED POSITION
4962 021236 004437 030774              JSR      R4,DWRTCK        ;GO SIMULATE WRITE CHECK
4963 021242 000400              400                      ;NUMBER OF WORDS TO SIMULATE
4964 021244 104053      ERROR    53              ;MR1 IN ERROR ON WRT CK OR ECC READ
4965 021246 104054      ERROR    54              ;ECC ERROR IN WRITE CHECK OP
4966 021250 104055      ERROR    55              ;CSI ERROR AFTER WRT CHK OR ECC READ
4967 021252 104041      ERROR    41              ;ECC REG INCORRECT AFTER ECC READ
4968 021254 104042      ERROR    42              ;ERR IN ECC PAT CALC.
4969 021256 104043      ERROR    43              ;ERR IN ECC POS COUNT

```

```

4970
4971
4972 021260 012700 002710      MOV      #40.*37.,R0      ;SET COUNT TO EMPTY SILO

```

```

4973
4974 021264 012762 000440 000026 15:    MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
4975 021272 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4976 021300 005300      DEC      R0              ;DEC COUNT
4977 021302 001370      BNE      15              ;LOOP UNTIL 0

```

```

4978
4979 021304 004437 034154      JSR      R4,GETREG        ;GET 611 REGS
4980 021310 013737 002260 002220      MOV      L.CSI,E.CSI     ;GET EXPECTED CSI
4981 021316 042737 000001 002220      BIC      #GO,E.CSI       ;CLEAR GO BIT
4982 021324 052737 000200 002220      BIS      #RDY,E.CSI      ;SET READY BIT
4983 021332 023737 002160 002220      CMP      T.CSI,E.CSI     ;CHECK IF CSI CORRECT
4984 021340 001401      BEQ      25              ;YES - SKIP
4985 021342 104057      ERROR    57              ;"CSI IN ERROR AFTER WRITE CHECK"

```

```

4986 021344
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002

```

```

*****
;TEST 34      WRITE CHECK WITH DCK
;
;      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
;      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
;      CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
;      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
;      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
;      A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
;      AND A 32 BIT ECC INDICATING A 1 BIT BURST ERROR.
;      VERIFY THAT THE POSITION REGISTER INCREMENTS PROPERLY
;      AND THAT THE FINAL PATTERN AND POSITION ARE CORRECT.
;      CHECK THAT DATA CHECK AND CONTROLLER ERROR ARE SET.
*****

```

```

5003 021344 000004      ST34:   SCOPE
5004 021346 012737 000001 001234      MOV      #1,$TIMES      ;;DO 1 ITERATION
5005
5006 021354 005037 002352      CLR      BSEDES         ;CLEAR BAD SECTOR DESIRED
5007 021360 005037 002354      CLR      HIBITS        ;CLEAR HI ORDER BITS
5008 021364 004437 034026      JSR      R4,LOADRK      ;LOAD "L" REGISTERS
5009 021370 000000      0                      ;CYLINDER 0
5010 021372 000          .BYTE    0              ;SECTOR 0
5011 021373 000          .BYTE    0              ;TRACK 0
5012 021374 050440      Ibuff          ;BUFFER ADDRESS Ibuff
5013 021376 177400      -400           ;WORD COUNT -400
5014 021400 000031      WRTCHK          ;COMMAND WRTCHK
5015
5016 021402 004437 033566      JSR      R4,CLRIBF      ;GO CLEAR INPUT BUFFER

```

```

5017
5018 021406 004437 033612 JSR R4,BLDDAT ;GO BUILD DATA
5019 021412 000005 S ;PATTERN 5
5020
5021 021414 012700 000400 MOV #400,R0 ;SET A COUNT FOR BUFFER SIZE
5022 021420 012701 050440 MOV #IBUFF,R1 ;SET ADDRESS OF IBUFF
5023 021424 012703 051444 MOV #OBUFF,R3 ;SET ADDRESS OF OBUFF
5024 021430 012321 75$: MOV (R3)+,(R1)+ ;MOV PATTERN TO IBUFF
5025 021432 005300 DEC R0 ;DEC COUNT
5026 021434 001375 BNE 75$ ;LOOP UNTIL PATTERN IS MOVED
5027
5028 021436 012700 001002 MOV #1002,R0 ;SET INDEX INTO BUFFER
5029 021442 012737 126073 052444 MOV #126073,OBUFF+1000 ;STORE ECC WORDS
5030 021450 012737 151052 052446 MOV #151052,OBUFF+1002
5031 021456 012737 000001 002344 MOV #1,ECCSRC ;SET SOURCE TO INDICATE ECC IN BUFFER
5032 021464 032760 000100 051444 BIT #BIT6,OBUFF(R0) ;TEST IF BIT SET
5033 021472 001004 BNE 100$ ;YES - SKIP
5034 021474 052760 000100 051444 BIS #BIT6,OBUFF(R0) ;ELSE SET BIT
5035 021502 000403 BR 101$
5036 021504 042760 000100 051444 100$: BIC #BIT6,OBUFF(R0) ;CLEAR BIT
5037 021512 101$:
5038
5039 021512 012762 100000 000000 110$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5040 021520 004437 034072 JSR R4,OPSTRAT ;START THE OPERATION
5041 021524 004437 025100 JSR R4,DISEEK ;SIMULATE IMPLIED SEEK
5042 021530 104004 ERROR 4 ;CS1 MISCOMPARE
5043 021532 104005 ERROR 5 ;MR1 "
5044 021534 104006 ERROR 6 ;MR2 "
5045 021536 104007 ERROR 7 ;MR3 "
5046
5047 021540 004437 025062 JSR R4,DSECTR ;SIMULATE SECTOR PULSE
5048
5049 021544 004437 025504 JSR R4,DRSYNC ;SIMULATE HEADER PREAMBLE
5050 021550 104010 ERROR 10 ;MR1 CONTENTS IN ERROR
5051 021552 104011 ERROR 11 ;CS1 CONTENTS IN ERROR
5052
5053 021554 004437 026042 JSR R4,DHDCMP ;SIMULATE HEADER SEARCH
5054 021560 104012 ERROR 12 ;MR1 CONTENTS IN ERROR
5055 021562 104013 ERROR 13 ;CS1 IN ERROR AFTER SEARCH
5056 021564 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5057
5058 021566 004437 030276 JSR R4,DRGPSN ;GO READ GAP AND SYNC
5059 021572 104015 ERROR 15 ;MR1 IN ERROR READING GAP
5060 021574 104016 ERROR 16 ;CS1 IN ERROR AFTER READING GAP
5061 021576 104032 ERROR 32 ;MR1 IN ERROR READING SYNC
5062 021600 104033 ERROR 33 ;CS1 IN ERROR AFTER READING SYNC
5063
5064 021602 012737 000001 002346 MOV #1,ECPATX ;LOAD EXPECTED PATTERN
5065 021610 012737 004066 002350 MOV #4066,ECPOSX ;LOAD EXPECTED POSITION
5066 021616 004437 030774 JSR R4,DWRTCK ;GO SIMULATE WRITE CHECK
5067 021622 000400 400 ;NUMBER OF WORDS TO SIMULATE
5068 021624 104053 ERROR 53 ;MR1 IN ERROR ON WRT CK OR ECC READ
5069 021626 104054 ERROR 54 ;ECC ERROR IN WRITE CHECK OP
5070 021630 104055 ERROR 55 ;CS1 ERROR AFTER WRT CHK OR ECC READ
5071 021632 104041 ERROR 41 ;ECC REG INCORRECT AFTER ECC READ
5072 021634 104042 ERROR 42 ;ERR IN ECC PAT CALC.
    
```

J08

```

5073 021636 104043          ERROR 43          ;ERR IN ECC POS COUNT
5074
5075
5076 021640 012737 000020 002326      MOV    #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
5077 021646 012700 000005          MOV    #5,R0          ;SET COUNT FOR ECCPOS PASS THRU 0
5078 021652 012701 013672          MOV    #13672,R1     ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
5079 021656 005037 001214          CLR    $TMP5         ;CLEAR SWITCH
5080
5081 021662 004737 033414          3$:   JSR    PC,RDBIT      ;GO READ BIT
5082 021666 004737 032260          JSR    PC,ECCGEN     ;GENERATE ECC
5083 021672 016237 000032 002214      MOV    RKECPT(R2),T.ECPT ;GET PATTERN
5084 021700 016237 000030 002212      MOV    RKECPS(R2),T.ECPS ;GET POSITION
5085 021706 005237 002350          INC    ECPOX         ;BUMP POSITION EXPECTED
5086 021712 013737 002350 002252      MOV    ECPOX,E.ECPS  ;SET FOR COMPARE AND REPORT
5087 021720 023737 002254 002214      CMP    E.ECPT,T.ECPT ;CHECK IF PATTERN CORRECT
5088 021726 001401          BEQ    1$           ;YES - SKIP
5089 021730 104042          ERROR 42
5090 021732 023737 002252 002212 1$:   CMP    E.ECPS,T.ECPS  ;CHECK IF POSITION CORRECT
5091 021740 001401          BEQ    2$           ;YES - SKIP
5092 021742 104043          ERROR 43
5093 021744 005237 002326          2$:   INC    BITCNT        ;BUMP BIT COUNT
5094 021750 005301          DEC    R1           ;DEC COUNT TO ZERO IN ECCPOS
5095 021752 001343          BNE    3$          ;NOT YET ZERO - LOOP
5096 021754 005300          DEC    R0           ;DEC PASS THRU 0 COUNT
5097 021756 001406          BEQ    4$          ;IF 0 - SKIP
5098 021750 012737 177777 002350      MOV    #-1,ECPOX    ;PRESET EXPECTED POS TO GO TO ZERO
5099 021766 012701 020000          MOV    #20000,R1    ;SET COUNT TO NEXT 0 IN ECC POS
5100 021772 000733          BR     3$          ;GO DO LOOP AGAIN
5101
5102 021774 005737 001214          4$:   TST    $TMP5         ;TEST SWITCH
5103 022000 001012          BNE    5$          ;ALREADY SET - SKIP
5104 022002 012701 010016          MOV    #10016,R1    ;SET COUNT TO FINAL POSITION
5105 022006 012700 000001          MOV    #1,R0        ;SET R0 TO RETURN AFTER ONE PASS
5106 022012 012737 177777 002350      MOV    #-1,ECPOX    ;SET EXP POS TO GO TO 0 ON NEXT BIT
5107 022020 005237 001214          INC    $TMP5        ;BUMP SWITCH
5108 022024 000716          BR     3$          ;GO DO LOOP AGAIN
5109
5110 022026 004737 033414          5$:   JSR    PC,RDBIT      ;ONE MORE BIT TO SET READY
5111 022032 004437 034154          JSR    R4,GETREG    ;GET '611 REGISTERS
5112 022036 013737 002260 002220      MOV    L.CS1,E.CS1  ;GET EXPECTED CS1
5113 022044 052737 100200 002220      BIS    #RDY!CERR,E.CS1 ;SET READY AND ERROR
5114 022052 042737 000001 002220      BIC    #GO,E.CS1    ;CLEAR GO
5115 022060 023737 002160 002220      CMP    T.CS1,E.CS1  ;CHECK IF CS1 CORRECT
5116 022066 001402          BEQ    6$          ;YES - SKIP
5117 022070 104044          ERROR 44
5118 022072 000410          BR     7$          ;
5119 022074 032737 000100 002174 6$:   BIT    #ECH,T.ER    ;TEST IF ECC HARD ERROR SET
5120 022102 001404          BEQ    7$          ;YES - SKIP
5121 022104 012737 100000 002234      MOV    #DCK,E.ER    ;SET EXPECTED ERROR REG
5122 022112 104045          ERROR 45
5123
5124 022114          7$:
5125
5126 ;*****
5127 ;*TEST 35 WRITE CHECK OF 18 BIT WORDS
5128 ;*

```

K08

```

5129 ;* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5130 ;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
5131 ;* CHECK OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
5132 ;* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5133 ;* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
5134 ;* A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
5135 ;* AND A GOOD 32 BIT ECC. MAKE SURE NO ERRORS SET.
5136 ;*
5137 ;*
5138 *****
5138 022114 000004 T35: SCOPE
5139 022116 012737 000001 001234 MOV #1, $TIMES ;;DO 1 ITERATION
5140
5141 022124 005037 002352 CLR BSEDES ;CLEAR BAD SECTOR DESIRED
5142 022130 005037 002354 CLR HIBITS ;CLEAR HI ORDER BITS
5143 022134 005037 002344 CLR ECCSRC ;CLEAR ECC SOURCE
5144 022140 004437 034026 JSR R4, LOADRK ;LOAD "L" REGISTERS
5145 022144 000000 0 ;CYLINDER 0
5146 022146 000 .BYTE 0 ;SECTOR 0
5147 022147 000 .BYTE 0 ;TRACK 0
5148 022150 050440 IBUFF ;BUFFER ADDRESS IBUFF
5149 022152 177400 -400 ;WORD COUNT -400
5150 022154 013031 WRTCHK!CFMT ;COMMAND WRTCHK!CFMT
5151
5152 022156 004437 033566 JSR R4, CLRIBF ;GO CLEAR INPUT BUFFER
5153
5154 022162 004437 033612 JSR R4, BLDDAT ;GO BUILD DATA
5155 022166 000004 4 ;PATTERN 4
5156
5157 022170 012700 000400 MOV #400, R0 ;SET A COUNT FOR BUFFER SIZE
5158 022174 012701 050440 MOV #IBUFF, R1 ;SET ADDRESS OF IBUFF
5159 022200 012703 051444 MOV #OBUFF, R3 ;SET ADDRESS OF OBUFF
5160 022204 012321 75$: MOV (R3)+, (R1)+ ;MOV PATTERN TO IBUFF
5161 022206 005300 R0 ;DEC COUNT
5162 022210 001375 BNE 75$ ;LOOP UNTIL PATTERN IS MOVED
5163
5164 022212 012762 100000 000000 110$: MOV #CCLR, RKCS1(R2) ;CLEAR CONTROLLER
5165 022220 004437 034072 JSR R4, OPSTRT ;START THE OPERATION
5166 022224 004437 025100 JSR R4, DISEEK ;SIMULATE IMPLIED SEEK
5167 022230 104004 ERROR 4 ;CS1 MISCOMPARE
5168 022232 104005 ERROR 5 ;MR1 ""
5169 022234 104006 ERROR 6 ;MR2 ""
5170 022236 104007 ERROR 7 ;MR3 ""
5171
5172 022240 004437 025062 JSR R4, DSECTR ;SIMULATE SECTOR PULSE
5173
5174 022244 004437 025504 JSR R4, DRSYNC ;SIMULATE HEADER PREAMBLE
5175 022250 104010 ERROR 10 ;MR1 CONTENTS IN ERROR
5176 022252 104011 ERROR 11 ;CS1 CONTENTS IN ERROR
5177
5178 022254 004437 026042 JSR R4, DHDCMP ;SIMULATE HEADER SEARCH
5179 022260 104012 ERROR 12 ;MR1 CONTENTS IN ERROR
5180 022262 104013 ERROR 13 ;CS1 IN ERROR AFTER SEARCH
5181 022264 104014 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5182
5183 022266 004437 030276 JSR R4, DRGPSN ;GO READ GAP AND SYNC
5184 022272 104015 ERROR 15 ;MR1 IN ERROR READING GAP
    
```

L08

```

5185 022274 104016 ERROR 16 ;CS1 IN ERROR AFTER READING GAP
5186 022276 104032 ERROR 32 ;MR1 IN ERROR READING SYNC
5187 022300 104033 ERROR 33 ;CS1 IN ERROR AFTER READING SYNC
5188
5189 022302 012737 000000 002346 MOV #0,ECPTX ;LOAD EXPECTED PATTERN
5190 022310 012737 005066 002350 MOV #5066,ECPOSX ;LOAD EXPECTED POSITION
5191 022316 004437 030774 JSR R4,DWRTCK ;GO SIMULATE WRITE CHECK
5192 022322 000400 400 ;NUMBER OF WORDS TO SIMULATE
5193 022324 104053 ERROR 53 ;MR1 IN ERROR ON WRT CK OR ECC READ
5194 022326 104054 ERROR 54 ;ECC ERROR IN WRITE CHECK OP
5195 022330 104055 ERROR 55 ;CS1 ERROR AFTER WRT CHK OR ECC READ
5196 022332 104041 ERROR 41 ;ECC REG INCORRECT AFTER ECC READ
5197 022334 104042 ERROR 42 ;ERR IN ECC PAT CALC.
5198 022336 104043 ERROR 43 ;ERR IN ECC POS COUNT
5199
5200

```

```

5201 022340 012700 003100 MOV #40.*40.,R0 ;SET COUNT TO EMPTY SILO
5202
5203 022344 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
5204 022352 012762 000040 000026 MOV #DMD,RKMR1(R2)
5205 022360 005300 DEC R0 ;DEC COUNT
5206 022362 001370 BNE 1$ ;LOOP UNTIL 0
5207

```

```

5208 022364 004437 034154 JSR R4,GETREG ;GET 611 REGS
5209 022370 013737 002260 002220 MOV L.CS1,E.CS1 ;GET EXPECTED CS1
5210 022376 042737 000001 002220 BIC #GO,E.CS1 ;CLEAR GO BIT
5211 022404 052737 000200 002220 BIS #RDY,E.CS1 ;SET READY BIT
5212 022412 023737 002160 002220 CMP T.CS1,E.CS1 ;CHECK IF CS1 CORRECT
5213 022420 001401 BEQ 2$ ;YES - SKIP
5214 022422 104057 ERROR 57 ;"CS1 IN ERROR AFTER WRITE CHECK"
5215 022424

```

```

2$:
*****
*TEST 36 WRITE CHECK OF A PARTIAL SECTOR
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
* A GOOD HEADER, TAKE ONE WORD TO BE WRITE-CHECKED, 377 WORDS
* NOT TO BE WRITE CHECKED, AND A GOOD 32 BIT ECC.
* MAKE SURE NO ERRORS SET.
*****

```

```

5228
5229 022424 000004 ST36: SCOPE
5230 022426 012737 000001 001234 MOV #1,STIMES ;;DO 1 ITERATION
5231
5232 022434 005037 002352 CLR BSEDES ;CLEAR BAD SECTOR DESIRED
5233 022440 005037 002354 CLR HIBITS ;CLEAR HI ORDER BITS
5234 022444 005037 002344 CLR ECCSRC ;CLEAR ECC SOURCE
5235 022450 004437 034026 JSR R4,LOADRK ;LOAD "L" REGISTERS
5236 022454 000000 0 ;CYLINDER 0
5237 022456 000 .BYTE 0 ;SECTOR 0
5238 022457 000 .BYTE 0 ;TRACK 0
5239 022460 050440 Ibuff ;BUFFER ADDRESS Ibuff
5240 022462 177777 -1 ;WORD COUNT -1

```

M08

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 103
 DZR6EA.P11 28-SEP-76 15:31 T36 WRITE CHECK OF A PARTIAL SECTOR

SEG 0103

5241	022464	000031				WRTCHK		;COMMAND WRTCHK
5242								
5243	022466	004437	033566			JSR	R4,CLRIBF	;GO CLEAR INPUT BUFFER
5244								
5245	022472	004437	033612			JSR	R4,BLDDAT	;GO BUILD DATA
5246	022476	000006				6		;PATTERN 6
5247								
5248	022500	012700	000400			MOV	#400,R0	;SET A COUNT FOR BUFFER SIZE
5249	022504	012701	050440			MOV	#IBUFF,R1	;SET ADDRESS OF IBUFF
5250	022510	012703	051444			MOV	#OBUFF,R3	;SET ADDRESS OF OBUFF
5251	022514	012321			75\$:	MOV	(R3)+,(R1)+	;MOV PATTERN TO IBUFF
5252	022516	005300				DEC	R0	;DEC COUNT
5253	022520	001375				BNE	75\$;LOOP UNTIL PATTERN IS MOVED
5254								
5255	022522	012762	100000	000000	110\$:	MOV	#CCLR,RKCS1(R2)	;CLEAR CONTROLLER
5256	022530	004437	034072			JSR	R4,OPSTRT	;START THE OPERATION
5257	022534	004437	025100			JSR	R4,DISEEK	;SIMULATE IMPLIED SEEK
5258	022540	104004				ERROR	4	;CS1 MISCOMPARE
5259	022542	104005				ERROR	5	;MR1
5260	022544	104006				ERROR	6	;MR2
5261	022546	104007				ERROR	7	;MR3
5262								
5263	022550	004437	025062			JSR	R4,DSECTR	;SIMULATE SECTOR PULSE
5264								
5265	022554	004437	025504			JSR	R4,DRSYNC	;SIMULATE HEADER PREAMBLE
5266	022560	104010				ERROR	10	;MR1 CONTENTS IN ERROR
5267	022562	104011				ERROR	11	;CS1 CONTENTS IN ERROR
5268								
5269	022564	004437	026042			JSR	R4,DHDCMP	;SIMULATE HEADER SEARCH
5270	022570	104012				ERROR	12	;MR1 CONTENTS IN ERROR
5271	022572	104013				ERROR	13	;CS1 IN ERROR AFTER SEARCH
5272	022574	104014				ERROR	14	;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5273								
5274	022576	004437	030276			JSR	R4,DRGPSN	;GO READ GAP AND SYNC
5275	022602	104015				ERROR	15	;MR1 IN ERROR READING GAP
5276	022604	104016				ERROR	16	;CS1 IN ERROR AFTER READING GAP
5277	022606	104032				ERROR	32	;MR1 IN ERROR READING SYNC
5278	022610	104033				ERROR	33	;CS1 IN ERROR AFTER READING SYNC
5279								
5280	022612	012737	000000	002346		MOV	#0,ECPATX	;LOAD EXPECTED PATTERN
5281	022620	012737	004066	002350		MOV	#4066,ECPOSX	;LOAD EXPECTED POSITION
5282	022626	004437	030774			JSR	R4,DWRTCK	;GO SIMULATE WRITE CHECK
5283	022632	000400				400		;NUMBER OF WORDS TO SIMULATE
5284	022634	104053				ERROR	53	;MR1 IN ERROR ON WRT CK OR ECC READ
5285	022636	104054				ERROR	54	;ECC ERROR IN WRITE CHECK OP
5286	022640	104055				ERROR	55	;CS1 ERROR AFTER WRT CHK OR ECC READ
5287	022642	104041				ERROR	41	;ECC REG INCORRECT AFTER ECC READ
5288	022644	104042				ERROR	42	;ERR IN ECC PAT CALC.
5289	022646	104043				ERROR	43	;ERR IN ECC POS COUNT
5290								
5291								
5292	022650	013700	002164			MOV	T.BA,R0	;GET ACTUAL BA
5293	022654	162700	050440			SUB	#IBUFF,R0	;SUB START VALUE OF BA
5294	022660	022700	000002			CMP	#2,R0	;TEST IF CORRECT NUM OF WORDS XFER
5295	022664	001404				BEQ	4\$;YES - SKIP
5296	022666	012737	050442	002224		MOV	#IBUFF+2,E.BA	;SET EXPECTED BA

N08

```

5297 022674 104046          ERROR 46          ;REPORT XFER LENGTH ERROR
5298
5299 022676 012701 000040    4$:  MOV      #8.,R1          ;SET COUNT TO END OF OPERATION
5300 022702 012762 000440    3$:  MOV      #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
5301 022710 012762 00004C    000026  MOV      #DMD,RKMR1(R2)
5302 022716 005301          DEC      R1          ;DEC COUNT
5303 022720 001370          BNE     3$          ;LOOP UNTIL ZERO
5304
5305 022722 004437 034154    1$:  JSR      R4,GETREG    ;GET 611 REGS
5306 022726 013737 002260    002220  MOV      L.CS1,E.CS1  ;GET EXPECTED CS1
5307 022734 042737 000001    002220  BIC     #GO,E.CS1    ;CLEAR GO BIT
5308 022742 052737 000200    002220  BIS     #RDY,E.CS1   ;SET READY BIT
5309 022750 023737 002160    002220  CMP     T.CS1,E.CS1  ;CHECK IF CS1 CORRECT
5310 022756 001401          BEQ     2$          ;YES - SKIP
5311 022760 104057          ERROR 57          ;"CS1 IN ERROR AFTER WRITE CHECK"
5312 022762
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333

```

```

*****
;TEST 37      WRITE CHECK ERROR (PART 1)
;
;      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
;      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
;      CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT.
;      CLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
;      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
;      A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 000000.
;      WRITE CHECKED DATA IS 000001.  MAKE SURE WRITE CHECK ERROR SETS.
;      REPEAT USING EACH OF THE FOLLOWING CONFIGURATIONS
;      AS WRITE CHECKED DATA:
;
;      000002 000040 001000 020000
;      000004 000100 002000 040000
;      000010 000200 004000 100000
;      000020 000400 010000
*****

```

```

5334 022762 000004          TST37: SCOPE
5335 022764 012737 000001    001234  MOV     #1,$TIMES    ;;DO 1 ITERATION
5336
5337 022772 005037 002352          CLR     BSEDES       ;CLEAR BAD SECTOR DESIRED
5338 022776 005037 002354          CLR     HIBITS      ;CLEAR HI ORDER BITS
5339 023002 005037 002344          CLR     ECCSRC      ;CLEAR ECC SOURCE
5340 023006 004437 034026          JSR     R4,LOADRK    ;LOAD "L" REGISTERS
5341 023012 000000          0          ;CYLINDER 0
5342 023014          000          .BYTE 0          ;SECTOR 0
5343 023015          000          .BYTE 0          ;TRACK 0
5344 023016 050440          IBUFF          ;BUFFER ADDRESS IBUFF
5345 023020 177777          -1          ;WORD COUNT -1
5346 023022 000031          WRTCHK          ;COMMAND WRTCHK
5347
5348 023024 012705 000020          MOV     #16.,R5     ;SET PATTERN COUNT
5349 023030 005037 051444          CLR     OBUF        ;CLEAR FIRST WORD OF OBUF
5350 023034 012737 000001    050440  MOV     #1,IBUFF     ;SET IN FIRST PATTERN
5351 023042 032737 000001    050440  BIT     #BIT0,IBUFF  ;IS BIT 0 SET
5352 023050 001003          BNE     112$        ;YES - PATTERN SHOULD BE 0

```



```

5353 023052 012737 177777 051444      MOV      # -1,OBUFF      ;ELSE PATTERN SHOULD BE ALL ONES
5354 023060 012737 023066 001110 1128:  MOV      #1108,$LPERR  ;SET LOCAL LOOP ON ERROR
5355
5356 023066 012762 100000 000000 1108:  MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5357 023074 004437 034072      JSR      R4,OPSTR1     ;START THE OPERATION
5358 023100 004437 025100      JSR      R4,DISEEK    ;SIMULATE IMPLIED SEEK
5359 023104 104004      ERROR   4             ;CSI MISCOMPARE
5360 023106 104005      ERROR   5             ;MR1
5361 023110 104006      ERROR   6             ;MR2
5362 023112 104007      ERROR   7             ;MR3
5363
5364 023114 004437 025062      JSR      R4,DSECTR    ;SIMULATE SECTOR PULSE
5365
5366 023120 004437 025504      JSR      R4,DRSYNC    ;SIMULATE HEADER PREAMBLE
5367 023124 104010      ERROR   10            ;MR1 CONTENTS IN ERROR
5368 023126 104011      ERROR   11            ;CSI CONTENTS IN ERROR
5369
5370 023130 004437 026042      JSR      R4,DHDCMP    ;SIMULATE HEADER SEARCH
5371 023134 104012      ERROR   12            ;MR1 CONTENTS IN ERROR
5372 023136 104013      ERROR   13            ;CSI IN ERROR AFTER SEARCH
5373 023140 104014      ERROR   14            ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5374
5375 023142 004437 030276      JSR      R4,DRGPSN    ;GO READ GAP AND SYNC
5376 023146 104015      ERROR   15            ;MR1 IN ERROR READING GAP
5377 023150 104016      ERROR   16            ;CSI IN ERROR AFTER READING GAP
5378 023152 104032      ERROR   32            ;MR1 IN ERROR READING SYNC
5379 023154 104033      ERROR   33            ;CSI IN ERROR AFTER READING SYNC
5380
5381 023156 012737 000000 002346      MOV      #0,ECPATX    ;LOAD EXPECTED PATTERN
5382 023164 012737 004066 002350      MOV      #4066,ECPOSX ;LOAD EXPECTED POSITION
5383 023172 004437 030774      JSR      R4,DWRATCK   ;GO SIMULATE WRITE CHECK
5384 023176 000001      1          ;NUMBER OF WORDS TO SIMULATE
5385 023200 104053      ERROR   53            ;MR1 IN ERROR ON WRT CK OR ECC READ
5386 023202 104054      ERROR   54            ;ECC ERROR IN WRITE CHECK OP
5387 023204 104055      ERROR   55            ;CSI ERROR AFTER WRT CHK OR ECC READ
5388 023206 104041      ERROR   41            ;ECC REG INCORRECT AFTER ECC READ
5389 023210 104042      ERROR   42            ;ERR IN ECC PAT CALC.
5390 023212 104043      ERROR   43            ;ERR IN ECC POS COUNT
5391
5392 023214 012701 000040      MOV      #9,*4,R1     ;SET COUNT
5393 023220 012762 000440 000026 15:  MOV      #DMD,MCV,RKMR1(R2) ;RUN CLOCK
5394 023226 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5395 023234 005301      DEC      R1           ;DEC COUNT
5396 023236 001370      BNE     15           ;LOOP UNTIL ZERO
5397
5398 023240 004437 034154      JSR      R4,GETREG    ;GET REGS
5399 023244 032737 040000 002170      BIT      #WCE,T.CS2   ;TEST IF WCE SET
5400 023252 001012      BNE     25           ;YES - SKIP
5401 023254 013737 002170 001162      MOV      T.CS2,$REG0  ;SET CS2 FOR REPORT
5402 023262 013737 050440 001164      MOV      IBUFF,$REG1  ;SET WORD READ
5403 023270 013737 051444 001166      MOV      OBUFF,$REG2  ;SET WORD FROM BUFFER
5404 023276 104060      ERROR   60           ;"WCE FAILED TO SET"
5405
5406 023300 104415      25:  SCOP1              ;LOCAL LOOP ON ERROR
5407
5408 023302 006337 050440      ASL     IBUFF        ;SHIFT FOR NEXT TEST PATTERN
    
```

023306 005305 DEC R5 :DEC PATTERN COUNT
023310 001266 BNE 1105 :LOOP UNTIL ALL PATTERNS TESTED

YES 40 WRITE CHECK ERROR (PART 2)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 177777.
WRITE CHECKED DATA IS 177776. MAKE SURE WRITE
CHECK ERROR SETS. REPEAT USING EACH OF THE FOLLOWING
CONFIGURATION AS WRITE CHECKED DATA:

177775 177737 176777 157777
177773 177677 175777 137777
177767 177577 173777 077777
177757 177377 167777

023312 000004
023314 012737 000001 001234

YES 40: SCOPE
MOV #1,5TIMES :DO 1 ITERATION

023322 005037 002352 CLR BSEDES :CLEAR BAD SECTOR DESIRED
023326 005037 002354 CLR HIBITS :CLEAR HI ORDER BITS
023332 005037 002344 CLR ECCSRC :CLEAR ECC SOURCE
023336 004437 034026 JSR R4,LOADRK :LOAD "L" REGISTERS
023342 000000 0 :CYLINDER 0
023344 000 :SECTOR 0
023346 000 :TRACK 0
023348 050440 Ibuff :BUFFER ADDRESS Ibuff
023350 177777 -1 :WORD COUNT -1
023352 000031 WATCHK :COMMAND WATCHK

023354 012705 000020 MOV #16,R5 :SET PATTERN COUNT
023360 005037 051444 CLR Obuff :CLEAR FIRST WORD OF Obuff
023364 012737 177776 050440 MOV #177776,IBuff :SET IN FIRST PATTERN
023372 032737 000001 050440 BIT #BIT0,IBuff :IS BIT 0 SET
023400 001003 BNE 1125 :YES - PATTERN SHOULD BE 0
023402 012737 177777 051444 MOV #-1,Obuff :ELSE PATTERN SHOULD BE ALL ONES
023410 012737 023416 001110 1125: MOV #1105,\$LPERA :SET LOCAL LOOP ON ERROR

023416 012762 100000 000000 1105: MOV #CLR,RKCS1(R2) :CLEAR CONTROLLER
023424 004437 034072 JSR R4,OPSTAT :START THE OPERATION
023430 004437 025100 JSR R4,DISEEK :SIMULATE IMPLIED SEEK
023434 104004 ERROR 4 :CS1 MISCOMPARE
023436 104005 ERROR 5 :MR1
023440 104006 ERROR 6 :MR2
023442 104007 ERROR 7 :MR3

023444 004437 025062 JSR R4,DSECTR :SIMULATE SECTOR PULSE

023450 004437 025504 JSR R4,DRSYNC :SIMULATE HEADER PREAMBLE
023454 104010 ERROR 10 :MR1 CONTENTS IN ERROR

22-SEP-76 15:31

T4C WRITE CHECK ERROR (PART 2)

```

023456 104011 ERROR 11 ;CSI CONTENTS IN ERROR
023460 004437 026242 JSR R4,DHOCMP ;SIMULATE HEADER SEARCH
023464 004437 026242 ERROR 12 ;MRI CONTENTS IN ERROR
023468 004437 026242 ERROR 13 ;CSI IN ERROR AFTER SEARCH
023472 004437 026242 ERROR 14 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
023476 004437 033276 JSR R4,DRGSPN ;GO READ GAP AND SYNC
023480 004437 033276 ERROR 15 ;MRI IN ERROR READING GAP
023484 004437 033276 ERROR 16 ;CSI IN ERROR AFTER READING GAP
023488 004437 033276 ERROR 22 ;MRI IN ERROR READING SYNC
023492 004437 033276 ERROR 33 ;CSI IN ERROR AFTER READING SYNC
023500 000000 000000 MOV R0,ECPATX ;LOAD EXPECTED PATTERN
023504 000000 000000 MOV R4066,ECPOSX ;LOAD EXPECTED POSITION
023508 000000 000000 JSR R4,DWATCK ;GO SIMULATE WRITE CHECK
023512 000000 000000 ERROR 53 ;NUMBER OF WORDS TO SIMULATE
023516 000000 000000 ERROR 54 ;MRI IN ERROR ON WRT CK OR ECC REAC
023520 000000 000000 ERROR 55 ;ECC ERROR IN WRITE CHECK OP
023524 000000 000000 ERROR 41 ;CSI ERROR AFTER WRT CK OR ECC REAC
023528 000000 000000 ERROR 42 ;ECC REG INCORRECT AFTER ECC REAC
023532 000000 000000 ERROR 43 ;ERR IN ECC PAT CALC
023536 000000 000000 ;ERR IN ECC POS COUNT
023540 000000 000000 MOV R8,*4,R1 ;SET COUNT
023544 000000 000000 MOV R0MC,MCLK,RKMR1,R2) ;RUN CLOCK
023548 000000 000000 MOV R0MC,RKMR1,R2)
023552 000000 000000 DEC R1 ;DEC COUNT
023556 000000 000000 BNE 18 ;LOOP UNTIL ZERO
023560 004437 034154 JSR R4,GETREG ;GET REGS
023564 004437 040000 002170 BIT R4CE,T,CS2 ;TEST IF WCE SET
023568 004437 001012 BNE 28 ;YES - SKIP
023572 004437 002170 001162 MOV R1,CS2,$REG0 ;SET CS2 FOR REPORT
023576 004437 050440 001164 MOV IBUFF,$REG1 ;SET WORD READ
023580 004437 051444 001166 MOV OBUFF,$REG2 ;SET WORD FROM BUFFER
023584 004437 104060 ERROR 60 ;"WCE FAILED TO SET"
023630 104415 28: SCOP1 ;LOCAL LOOP ON ERROR
023632 006337 050440 ASL IBUFF ;SHIFT FOR NEXT TEST PATTERN
023636 005537 050440 ADC IBUFF ;PUT CARRY BACK IN
023640 005305 DEC R5 ;DEC PATTERN COUNT
023644 001264 BNE 1108 ;LOOP UNTIL ALL PATTERNS TESTED

```

*TEST 41 WRITE CHECK ERROR (PART 3)

* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
* A GOOD HEADER, AND ONE WORD OF DATA CONSISTING
* OF 200000. WRITE CHECK DATA IS 000000. MAKE SURE
* WRITE CHECK ERROR SETS. REPEAT USING 400000 AS

SIMULATED DATA.

```

*****
41: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
CLR BSEDES ;:CLEAR BAD SECTOR DESIRED
CLR ECCSRC ;:CLEAR ECC SOURCE
JSR R4,LOADRK ;:LOAD "L" REGISTERS
D ;:CYLINDER 0
.BYTE 0 ;:SECTOR 0
.BYTE 0 ;:TRACK 0
IBUFF ;:BUFFER ADDRESS IBUFF
-! ;:WORD COUNT -!
WRTCHK!CFMT ;:COMMAND WRTCHK!CFMT

42: MOV #40000,HIBITS ;:SET HIBITS FOR PATTERN OF 200000
CLR IBUFF ;:SET LOW BITS FOR READ
CLR OBUFF ;:SET 0 IN BUFFER
MOV #1105,$LPERR ;:SET LOCAL LOOP ON ERROR
MOV #CCLR,RKCSI(R2) ;:CLEAR CONTROLLER

43: MOV #2000,R0 ;:SET STALL COUNT
DEC R0 ;:DEC COUNT
BNE 45 ;:LOOP UNTIL ZERO

44: JSR R4,OPSTRT ;:START THE OPERATION
JSR R4,DISEEK ;:SIMULATE IMPLIED SEEK
ERROR 4 ;:CSI MISCOMPARE
ERROR 5 ;:MR1
ERROR 6 ;:MR2
ERROR 7 ;:MR3

45: JSR R4,DSECTR ;:SIMULATE SECTOR PULSE

46: JSR R4,DRSYNC ;:SIMULATE HEADER PREAMBLE
ERROR 10 ;:MR1 CONTENTS IN ERROR
ERROR 11 ;:CSI CONTENTS IN ERROR

47: JSR R4,DHDCMP ;:SIMULATE HEADER SEARCH
ERROR 12 ;:MR1 CONTENTS IN ERROR
ERROR 13 ;:CSI IN ERROR AFTER SEARCH
ERROR 14 ;:RKDCYL OR RKDA IN ERROR OFTER SEARCH

48: JSR R4,DRGPSN ;:GO READ GAP AND SYNC
ERROR 15 ;:MR1 IN ERROR READING GAP
ERROR 16 ;:CSI IN ERROR AFTER READING GAP
ERROR 32 ;:MR1 IN ERROR READING SYNC
ERROR 33 ;:CSI IN ERROR AFTER READING SYNC

49: MOV #0,ECPATX ;:LOAD EXPECTED PATTERN
MOV #5066,ECPOSX ;:LOAD EXPECTED POSITION
JSR R4,DWRTCK ;:GO SIMULATE WRITE CHECK
1 ;:NUMBER OF WORDS TO SIMULATE
ERROR 53 ;:MR1 IN ERROR ON WRT CK OR ECC READ
ERROR 54 ;:ECC ERROR IN WRITE CHECK CP

```

F09

```

5577 024056 104055          ERROR 55          :CS1 ERROR AFTER WRT CHK OR ECC REAC
5578 024060 104041          ERROR 41          :ECC REG INCORRECT AFTER ECC READ
5579 024062 104042          ERROR 42          :ERR IN ECC PAT CALC.
5580 024064 104043          ERROR 43          :ERR IN ECC POS COUNT
5581
5582 024066 012701 000040          MOV #8,*4,R1          :SET COUNT
5583 024072 012762 000440 000026 1S: MOV #DMD!MCLK,RKMR1(R2) :RUN CLOCK
5584 024100 012762 000040 000026          MOV #DMD,RKMR1(R2)
5585 024106 005301          DEC R1              :DEC COUNT
5586 024110 001370          BNE 1S              :LOOP UNTIL ZERO
5587
5588 024112 004437 034154          JSR R4,GETREG       :GET REGS
5589 024116 032737 040000 002170          BIT #WCE,T.CS2     :TEST IF WCE SET
5590 024124 001025          BNE 2S              :YES - SKIP
5591 024126 013737 002170 001162          MOV T.CS2,$REG0    :SET CS2 FOR REPORT
5592 024134 013737 050440 001164          MOV Ibuff,$REG1    :SET WORD READ
5593 024142 013737 051444 001166          MOV Obuff,$REG2    :SET WORD FROM BUFFER
5594 024150 005037 001170          CLR $REG3           :SET UP FOR REPORT
5595 024154 013737 002354 001172          MOV HIBITS,$REG4
5596 024162 012700 000016          MOV #14,R0
5597 024166 006037 001172          3S: ROR $REG4
5598 024172 005300          DEC R0
5599 024174 001374          BNE 3S
5600 024176 104063          ERROR 63          : "WCE FAILED TO SET"
5601
5602 024200 104415          2S: SCOP1          :LOCAL LOOP ON ERROR
5603
5604 024202 023727 002354 100000          CMP HIBITS,#100000 :TEST IF SECOND PATTERN DONE
5605 024210 001404          BEQ 5S              :YES - EXIT
5606 024212 012737 100000 002354          MOV #100000,HIBITS :SET FOR PATTERN 400000
5607 024220 000643          BR 110S             :LOOP UNTIL ALL PATTERNS TESTED
5608
5609
5610
5611 *****
5612 *TEST 42 WRITE CHECK ERROR (PART 4)
5613 *
5614 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5615 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
5616 * CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT.
5617 * CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5618 * AND DRIVE CLEAR MESSAGES SIMULATE A SECTOR PULSE,
5619 * A GOOD HEADER, 125 GOOD WORDS, AND A BAD WORD.
5620 * MAKE SURE WRITE CHECK ERROR SET. CHECK WORD COUNT
5621 * AND BUS ADDRESS.
5622 *****
5623 024222 000004          TST42: SCOPE
5624 024224 012737 000001 001234          MOV #1,$TIMES      ;;DO 1 ITERATION
5625
5626 024232 005037 002354          CLR HIBITS         :CLEAR HI BITS
5627 024236 005037 002352          CLR BSEDES        :CLEAR BAD SECTOR DESIRED
5628 024242 004437 034026          JSR R4,LOADRK     :LOAD "L" REGISTERS
5629 024246 000000          0                 :CYLINDER 0
5630 024250 000          .BYTE 0           :SECTOR 0
5631 024251 000          .BYTE 0           :TRACK 0
5632 024252 050440          Ibuff            :BUFFER ADDRESS IBLFF

```



```

5725 .SBTTL END OF PASS ROUTINE
5726
5727 ;*****
5728 ;*INCREMENT THE PASS NUMBER ($PASS)
5729 ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
5730 ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
5731 ;*IF THERES A MONITOR GO TO IT
5732 ;*IF THERE ISN'T JUMP TO NEWPAS
5733
5734 $EOP:
5735 024652 000004 SCOPE
5736 024652 005037 001102 CLR $TSTNM ;; ZERO THE TEST NUMBER
5737 024654 005037 001234 CLR $TIMES ;; ZERO THE NUMBER OF ITERATIONS
5738 024664 005237 001256 INC $PASS ;; INCREMENT THE PASS NUMBER
5739 024670 042737 100000 001256 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
5740 024676 005327 DEC (PC)+ ;; LOOP?
5741 024700 000001 $EOPCT: .WORD 1
5742 024702 003063 BGT $DOAGN ;; YES
5743 024704 012737 MOV (PC)+,2(PC)+ ;; RESTORE COUNTER
5744 024706 000001 $ENDCT: .WORD 1
5745 024710 024700 $EOPCT
5746 024712 104401 024720 TYPE ,65S ;; TYPE ASCIZ STRING
5747 024716 000407 BR ,64S ;; GET OVER THE ASCIZ
5748
5749 024736 024736 013746 001256 ;;65S: .ASCIZ <12><15>/END PASS #/
5750 024736 013746 001256 64S: MOV $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
5751
5752 024742 104405 TYPDS ;; TYPE PASS NUMBER
5753 024744 104401 024752 TYPE ,67S ;; GO TYPE--DECIMAL ASCII WITH SIGN
5754 024750 000421 BR ,66S ;; TYPE ASCIZ STRING
5755
5756 025014 025014 013746 001112 ;;67S: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
5757 025014 013746 001112 66S: MOV $ERTTL,-(SP) ;; SAVE $ERTTL FOR TYPEOUT
5758
5759 025020 104405 TYPDS ;; TOTAL NUMBER OF ERRORS
5760 025022 104401 001245 TYPE $CRLF ;; GO TYPE--DECIMAL ASCII WITH SIGN
5761 025026 005037 001112 CLR $ERTTL ;; TYPE CARRIAGE RETURN, LINE FEED
5762 025032 013700 000042 $GET42: MOV ,2*42,RO ;; CLEAR ERROR TOTAL
5763 025036 001405 BEQ $DOAGN ;; GET MONITOR ADDRESS
5764 025040 000005 RESET ;; BRANCH IF NO MONITOR
5765 025042 004710 $ENDAD: JSR PC,(RO) ;; CLEAR THE WORLD
5766 025044 000240 NOP ;; GO TO MONITOR
5767 025046 000240 NOP ;; SAVE ROOM
5768 025050 000240 NOP ;; FOR
5769 025052 $DOAGN: ;; ACT11
5770 025052 000137 JMP ,2(PC)+ ;; RETURN
5771 025054 003376 $RTNAD: .WORD NEWPAS
5772 025056 377 377 000 $ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
5773 025062 .EVEN
5774
5775 .SBTTL DIAGNOSTIC MODE SECTOR PULSE
5776 ;* THE SECTOR PULSE IS PROVIDED BY THIS ROUTINE
5777 025062 012762 000140 000026 DSECTR: MOV #DMD!MSP,RKMRI(R2) ;SET SECTOR PULE
5778 025070 012762 000040 000026 MOV #DMD,RKMRI(R2) ;RESET SECTOR PULSE
5779 025076 000204 RTS R4
5780

```



```

5781 .SBTTL DIAGNOSTIC MODE IMPLIED SEEK
5782 :* THE CONTROLLER IS CLOCKED THROUGH THE SEEK MESSAGES
5783 :* GETTING STATUS, AND CLEARING THE DRIVE. SINCE THIS REQUIRES A
5784 :* DIFFERENT COUNT FOR WRITE DATA AND READ DATA/WRITE CHECK,
5785 :* THE ROUTINE LOOKS AT THE COMMAND THAT WAS
5786 :* STARTED (L.CS1) AND SETS THE CLOCK COUNT ACCORDINGLY.
5787 :*
5788 :* AT THE END OF THE IMPLIED SEEK THE CONTENTS OF CS1, MR1, MR2
5789 :* AND MR3 ARE CHECKED TO VERIFY THE OPERATION. ALL DISCREPENCIES
5790 :* ARE REPORTED IN THE ORDER LISTED.
5791 :*
5792 :* CALL:
5793 :* JSR R4,DISEEK
5794 :* ERROR 4 ;CS1 MISCOMPARE ERROR
5795 :* ERROR 5 ;MR1 MISCOMPARE ERROR
5796 :* ERROR 6 ;MR2 MISCOMPARE ERROR
5797 :* ERROR 7 ;MR3 MISCOMPARE ERRON
5798
5799 025100 DISEEK:
5800 025100 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
5801 025102 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
5802 025104 032737 000002 002260 BIT #BIT1,L.CS1 ;:CHECK IF WRITE DATA
5803 025112 001003 BNE 10$ ;YES - SKIP
5804 025114 012700 000316 MOV #51.*4+2,R0 ;:SET COUNT FOR READ OR WRITE CHECK
5805 025120 000412 BR 1$
5806
5807 025122 012700 004612 10$: MOV #66.*37.,R0 ;:COUNT FOR CLOCK THROUGH SILO LOAD
5808 025126 032737 010000 002260 BIT #CFMT,L.CS1 ;:TEST IF 22 SECTOR MODE
5809 025134 001402 BEQ 11$ ;NO - SKIP
5810 025136 012700 005222 MOV #66.*41.,R0 ;:ELSE CHANGE FOR 18 BIT WORDS
5811 025142 062700 000016 11$: ADD #4*3+2,R0 ;:AND SHIFT REGISTER LOAD
5812
5813 025146 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ;:SET CLOCK
5814 025154 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:CLEAR CLOCK
5815 025162 005300 DEC R0 ;:LOOP UNTIL R0 IS ZERO
5816 025164 001370 BNE 1$
5817
5818 025166 013703 002274 MOV L.DCYL,R3 ;:GET DESIRED CYLINDER
5819 025172 004437 034242 JSR R4,FSBLVV ;:REVERSE ORDER OF BITS
5820 025176 010337 002246 MOV R3,E.MR2 ;:STORE AS EXPECTED MR2
5821 025202 113703 002267 MOV# L.DT,R3 ;:GET DESIRED TRACK/SECTOR
5822 025206 042703 177400 BIC #177400,R3 ;:CLEAR UNUSED BITS
5823 025212 012700 000005 MOV #5,R0 ;:SET SHIFT COUNT
5824 025216 006303 15$: ASL R3 ;:SHIFT FOR TRACK ALIGNMENT
5825 025220 005300 DEC R0 ;:DEC COUNT
5826 025222 001375 BNE 15$ ;:LOOP UNTIL ZERO
5827 025224 153703 002266 BISB L.DS,R3 ;:INSERT SECTOR NUMBER
5828 025230 032737 010000 002260 BIT #CFMT,L.CS1 ;:TEST IF 18 BIT MODE
5829 025236 001402 BEQ 16$ ;NO - SKIP
5830 025240 052703 001000 BIS #BIT9,R3 ;:ELSE SET FORMAT BIT IN HDR WRD
5831 025244 004437 034242 16$: JSR R4,FSBLVV ;:REVERSE BIT ORDER
5832 025250 010337 002250 MOV R3,E.MR3 ;:STORE AS EXPECTED MR3
5833
5834 025254 013737 002260 002220 MOV L.CS1,E.CS1 ;:GET EXPECTED CS1
5835 025262 012737 022040 002244 MOV #DMD!ECCW!MEWD,E.MR1 ;:SET UP EXPECTED MR1
5836 025270 004437 034154 JSR R4,GETREG ;:GET RK REGS
    
```

```

5837
5838 025274 023737 002160 002220      CMP      T.CS1,E.CS1      ;CHECK CS1
5839 025302 001411                      BEQ      2$             ; OK - SKIP
5840 025304 012737 025326 001236      MOV      #2$, $ESCAPE  ;SET ESCAPE TO RETURN FOR
5841                                ;REMAINING TESTS
5842 025312 032777 001000 153620      BIT      #BIT9, $SWR   ;CHECK IF LOOP ON ERROR
5843 025320 001056                      BNE      9$             ;YES - SKIP
5844
5845 025322 010446                      4$:     MOV      R4, -(SP) ;SET STACK FOR RETURN TO SUB-ROUTINE
5846 025324 000204                      RTS
5847
5848 025326 005724                      2$:     TST      (R4)+     ;BUMP TO NEXT ERROR CALL
5849 025330 023737 002244 002204      CMP      E.MR1,T.MR1   ;CHECK MR1
5850 025336 001410                      BEQ      3$             ;OK - SKIP
5851 025340 032777 001000 153572      BIT      #BIT9, $SWR   ;CHECK IF LOOP ON ERROR
5852 025346 001043                      BNE      9$             ;YES - SKIP
5853 025350 012737 025360 001236      MOV      #3$, $ESCAPE  ;SET ESCAPE FOR RETURN TO SUB-ROUTINE
5854 025356 000761                      BR
5855
5856 025360 005724                      3$:     TST      (R4)+     ;SAME OPERATION AS ABOVE TO
5857 025362 023737 002246 002206      CMP      E.MR2,T.MR2   ;CHECK MR2
5858 025370 001410                      BEQ      5$
5859 025372 032777 001000 153540      BIT      #BIT9, $SWR
5860 025400 001026                      BNE      9$
5861 025402 012737 025412 001236      MOV      #5$, $ESCAPE
5862 025410 000744                      BR
5863
5864 025412 005724                      5$:     TST      (R4)+     ;SAME OPERATION AS ABOVE TO
5865 025414 023737 002250 002210      CMP      E.MR3,T.MR3   ;CHECK MR3
5866 025422 001414                      BEQ      8$
5867 025424 032777 001000 153506      BIT      #BIT9, $SWR
5868 025432 001011                      BNE      9$
5869 025434 012737 025444 001236      MOV      #6$, $ESCAPE
5870 025442 000727                      BR
5871
5872 025444 032777 001000 153466      6$:     BIT      #BIT9, $SWR ;CHECK IF LOOP ON ERROR
5873 025452 001001                      BNE      9$             ;YES - SKIP
5874 025454 005724                      8$:     TST      (R4)+     ;BUMP TO NO ERROR RETURN
5875 025456 005037 001236      9$:     CLR      $ESCAPE    ;CLEAR ESCAPE
5876 025462 012603                      MOV      (SP)+, R3     ;POP STACK INTO R3
5877 025464 012600                      MOV      (SP)+, R0     ;POP STACK INTO R0
5878 025466 000204                      RTS      R4
5879
5880 025470 005037 001236      7$:     CLR      $ESCAPE    ;CLEAR ESCAPE
5881 025474 012706 001100      MOV      #STACK, SP   ;CLEAN OFF STACK
5882 025500 000177 153404      JMP      @ $LPERR     ;GO TO LOOP START
5883
5884                                .SBTTL  DIAGNOSTIC MODE READ SYNC PATTERN (PREAMBLE)
5885                                ;*      THE 255 "0" BITS AND 1 "1" BIT ARE CLOCKED THROUTH THE READ.
5886                                ;*      THE CONTENTS OF MR1 ARE MONITORED TO WATCH FOR AN ERROR.
5887                                ;*      AT THE END OF THE SYNC READ, CSI IS CHECKED TO INSURE IT
5888                                ;*      DID NOT GET CHANGED.
5889                                ;*
5890                                ;*      CALL:
5891                                ;*      JSR      R4, DRSYNC
5892                                ;*      ERROR  10      ;MR1 CONTENTS IN ERROR

```

```

5893      ;*      ERROR 11      ;CSI CONTENTS IN ERROR
5894      ;*
5895      ;*      IF AN ERROR IS DETECTED, THE SUBROUTINE WILL SET UP TO ABORT
5896      ;*      THE PRESENT TEST UNLESS LOOP ON ERROR IS SET.
5897
5898      025504      DRSYNC:
5899      025504      010046      MOV      RD, -(SP)      ;; PUSH RD ON STACK
5900      025506      005037      002326      CLR      BITCNT      ;; CLEAR BIT COUNT
5901      025512      005037      001236      CLR      $ESCAPE      ;; CLEAR ESCAPE
5902      025516      012700      000200      MOV      #128, RO      ;; SET COUNT TO CLOCK TO READ GATE
5903      025522      005037      002320      CLR      PR.BIT      ;; CLEAR PRESENT BIT
5904      025526      005037      002322      CLR      M1.BIT      ;; CLEAR PREVIOUS BIT
5905      025532      012737      022040      002244      MOV      #DMD!MEWD!ECCW.E.MR1 ; SET UP EXPECTED MR1
5906
5907      025540      004737      033414      3$:      JSR      PC,RDBIT      ;; GO READ A BIT
5908      025544      016237      000026      002204      MOV      RKMRI(R2), T.MR1 ; GET MR1
5909      025552      023737      002204      002244      CMP      T.MR1,E.MR1      ;; CHECK IF IT IS CORRECT
5910      025560      001424      BEQ      2$      ;; YES - SKIP
5911
5912      025562      012737      025632      001236      MOV      #2$, $ESCAPE      ;; SET ESCAPE FOR CONTINUE TEST
5913
5914      025570      005037      001236      4$:      CLR      $ESCAPE      ;; CLEAR ESCAPE
5915      025574      032777      001000      153336      BIT      #BIT9, $SWR      ;; CHECK IF LOOP ON ERROR
5916      025602      001011      BNE      1$      ;; YES SKIP
5917      025604      013700      001254      MOV      $TESTN, RO      ;; ELSE SET UP TO SKIP TO NEXT TEST
5918      025610      006300      ASL      RO
5919      025612      016037      035270      001236      MOV      $SWO8TB(RO), $ESCAPE
5920      025620      162737      000002      001236      SUB      #2, $ESCAPE
5921
5922      025626      012600      1$:      MOV      (SP)+, RO      ;; POP STACK INTO RO
5923      025630      000204      RTS      R4
5924
5925      025632      005237      002326      2$:      INC      BITCNT      ;; INCREMENT THE BIT COUNT
5926      025636      005300      DEC      RO      ;; LOOP UNTIL READY FOR READ GATE
5927      025640      001337      BNE      3$
5928
5929      025642      012700      000177      MOV      #127, RO      ;; SET COUNT FOR REST OF SYNC
5930      025646      012737      122040      002244      MOV      #DMD!MEWD!ECCW!RDGATE.E.MR1 ; SET NEW EXPECTED MR1
5931
5932      025654      004737      033414      5$:      JSR      PC,RDBIT      ;; GO READ BIT
5933      025660      016237      000026      002204      MOV      RKMRI(R2), T.MR1 ; GET MR1
5934      025666      023737      002204      002244      CMP      T.MR1,E.MR1      ;; CHECK IF CORRECT
5935      025674      001404      BEQ      6$      ;; YES - SKIP
5936      025676      012737      025706      001236      MOV      #6$, $ESCAPE      ;; SET FOR CONTINUE TESTING
5937      025704      000445      BR      50$      ;; GO TO ERROR EXIT
5938      025706      005237      002326      6$:      INC      BITCNT      ;; BUMP BIT COUNTER
5939      025712      005300      DEC      RO      ;; LOOP UNTIL READY FOR SYNC 1
5940      025714      001357      BNE      5$
5941
5942      025716      013737      002320      002322      MOV      PR.BIT, M1.BIT      ;; MOV PRESENT TO PREVIOUS
5943      025724      012737      000001      002320      MOV      #1, PR.BIT      ;; SET SYNC 1 BIT
5944      025732      004737      033414      JSR      PC,RDBIT      ;; GO READ BIT
5945      025736      016237      000026      002204      MOV      RKMRI(R2), T.MR1 ; GET MR1
5946      025744      023737      002204      002244      CMP      T.MR1,E.MR1      ;; CHECK IF CORRECT
5947      025752      001404      BEQ      7$      ;; YES - SKIP
5948
    
```

```

5949 025754 012737 025764 001236      MOV    #7$, $ESCAPE      ;SET RETURN FOR CONTINUE TEST
5950 025762 000416                      BR     50$               ;GO TO ERROR EXIT
5951
5952 025764 004437 034154      7$:   JSR    R4, GETREG    ;GO GET RK611 REGISTERS
5953 025770 005237 002326      INC    BITCNT           ;BUMP BIT COUNTER
5954 025774 005724                      TST    (R4)+            ;BUMP TO NEXT ERROR CALL
5955 025776 013737 002260 002220    MOV    L.CS1, E.CS1     ;GET EXPECTED CS1
5956 026004 023737 002160 002220    CMP    T.CS1, E.CS1    ;CHECK IF CORRECT
5957 026012 001266                      BNE    4$               ;NO - SKIP
5958
5959 026014 005724                      TST    (R4)+            ;BUMP TO NO ERROR RETURN
5960 026016 000703                      BR     1$               ;SKIP TO RETURN
5961
5962 026020 032777 001000 153112 50$:   BIT    #BIT9, $SWR     ;TEST IF LOOP ON ERROR
5963 026026 001403                      BEQ    51$              ;NO - SKIP
5964 026030 005037 001236      CLR    $ESCAPE         ;CLEAR FOR SKIP TO NEXT TEST
5965 026034 000674                      BR     1$
5966 026036 010446      51$:   MOV    R4, -(SP)       ;SET UP STACK FOR RETURN
5967 026040 000204                      RTS    R4
5968
5969      .SBTTL  DIAGNOSTIC MODE HEADER READ AND COMPARE
5970      *      THIS ROUTINE SIMULATES THE THREE HEADER WORDS READ AND
5971      *      COMPARE THAT IS PART OF ALL DATA TRANSFER OPERATIONS.
5972      *
5973      *      THE EXPECTED HEADER IS CALCULATED FROM THE CYLINDER, TRACK,
5974      *      AND SECTOR SPECIFIED IN THE "L" REGISTERS. THE READING OF THE
5975      *      3 WORDS IS DONE AND MRI IS MONITORED TO INSURE IT DOES
5976      *      NOT CHANGE. AT THE END OF THE READ/COMPARE, THE CONTENTS
5977      *      OF RKCS1 IS CHECKED AND THE INCREMENT OF RKDCYL AND RKDA IS
5978      *      VERIFIED. IF THE INCREMENT IS GOOD, THE "L" REGISTERS ARE
5979      *      UPDATED TO THE NEW PACK ADDRESS IN CASE OF MULTI-SECTOR
5980      *      OPERATIONS. THE BAD SECTOR ERROR DESIRED (BSEDES) SWITCH
5981      *      IS TESTED AND IF IT IS NON-ZERO, THE 2ND AND 3RD HEADER
5982      *      WORDS ARE CHANGED ACCORDING TO BSEDES CONTENTS. THIS
5983      *      CAN BE USED TO FORCE HEADER TYPE ERROR.
5984      *
5985      *      CALL:
5986      *      JSR    R4, DHDCMP
5987      *      ERROR 12      ;ERROR IN MRI
5988      *      ERROR 13      ;ERROR IN CS1
5989      *      ERROR 14      ;ERROR IN PACK ADDRESS
5990      *
5991      DHDCMP:
5992      MOV    R0, -(SP)    ;; PUSH R0 ON STACK
5993      MOV    R1, -(SP)    ;; PUSH R1 ON STACK
5994      MOV    R3, -(SP)    ;; PUSH R3 ON STACK
5995      MOV    R5, -(SP)    ;; PUSH R5 ON STACK
5996      CLR    BITCNT      ;CLEAR BIT COUNTER
5997      MOV    #5, R0       ;SET A SHIFT COUNT
5998      MOVB  L.DT, R1      ;GET DESIRED TRACK
5999      BIC   #177400, R1   ;CLEAR UNUSED BITS
6000
6001 026072 006301      1$:   ASL    R1              ;SHIFT TRACK TO ALIGN TO
6002 026074 005300      DEC    R0              ;HEADER WORD FORMAT
6003 026076 001375      BNE    1$
6004

```

6005	026100	153701	002266		BISB	L.DS,R1	; INSERT DESIRED SECTOR	
6006	026104	052701	140000		BIS	#BIT15:BIT14,R1	; SET BS BITS	
6007	026110	013700	002274		MOV	L.DCYL,R0	; GET DESIRED CYLINDER	
6008	026114	032737	010000	002260	BIT	#CFMT,L.CS1	; TEST IF 22 SECTOR FORMAT	
6009	026122	001402			BEQ	2\$; NO - SKIP	
6010	026124	052701	001000		BIS	#BIT9,R1	; ELSE SET FORMAT BIT	
6011	026130	010003		2\$:	MOV	R0,R3	; COMPUTE VRC	
6012	026132	010105			MOV	R1,R5		
6013	026134	040005			BIC	R0,R5		
6014	026136	040103			BIC	R1,R3		
6015	026140	050503			BIS	R5,R3		
6016	026142	005737	002352		TST	BSEDES	; TEST IF BSE DESIRED	
6017	026146	001404			BEQ	23\$; NO - SKIP	
6018	026150	043701	002352		BIC	BSEDES,R1	; CLEAR BIT IN WORD 2	
6019	026154	043703	002352		BIC	BSEDES,R3	; CORRECT HVRC	
6020	026160			23\$:				
6021								
6022								
6023								
6024	026160	005037	001204		CLR	\$TMP1	; CLEAR A PASS COUNTER	
6025	026164	012737	122040	002244	MOV	#DMD!MEWD!ECCW!R0GATE,E.MR1	; SET EXPECTED MR1	
6026								
6027	026172	012705	000001	002322	11\$:	MOV	#BIT0,R5	; ESTABLISH BEGINNING OF DATA WORD
6028	026176	013737	002320	7\$:	MOV	PR.BIT,M1.BIT	; STORE PREVIOUS BIT	
6029	026204	005037	002320		CLR	PR.BIT	; PRESET FOR PRESENT BIT 0	
6030	026210	030500			BIT	R5,R0	; TEST IF BIT TO BE A ONE	
6031	026212	001403			BEQ	3\$; NO - SKIP	
6032	026214	012737	000001	002320	MOV	#BIT0,PR.BIT	; ELSE CHANGE TO ONE	
6033	026222	004737	033414		3\$:	JSR	PC,R0BIT	; GO READ A BIT
6034	026226	016237	000026	002204	MOV	RKMR1(R2),T.MR1	; GET MR1 FOR TESTING	
6035	026234	023737	002204	002244	CMP	T.MR1,E.MR1	; CHECK IF CORRECT	
6036	026242	001431			BEQ	6\$; YES - SKIP	
6037	026244	012737	026326	001236	MOV	#6\$,\$ESCAPE	; SET RETURN TO CONTINUE TEST	
6038	026252	000137	026756		JMP	50\$		
6039								
6040	026256	005037	001236		4\$:	CLR	\$ESCAPE	; CLEAR TO JUMP TO NEXT TEST
6041	026262	032777	001000	152650	BIT	#BIT9,\$SWR	; CHECK IF LOOP ON ERROR	
6042	026270	001011			BNE	5\$; YES - SKIP	
6043	026272	013705	001254		MOV	\$TESTN,R5	; GET CURRENT TEST NUMBER	
6044	026276	006305			ASL	R5	; AND SET UP TO ESCAPE TO NEXT TEST	
6045	026300	016537	035270	001236	MOV	\$SW08TB(R5),\$ESCAPE		
6046	026306	162737	000002	001236	SUB	#2,\$ESCAPE		
6047								
6048	026314				5\$:			
6049	026314	012605			MOV	(SP)+,R5	:: POP STACK INTO R5	
6050	026316	012603			MOV	(SP)+,R3	:: POP STACK INTO R3	
6051	026320	012601			MOV	(SP)+,R1	:: POP STACK INTO R1	
6052	026322	012600			MOV	(SP)+,R0	:: POP STACK INTO R0	
6053	026324	000204			RTS	R4		
6054								
6055	026326	005237	002326		6\$:	INC	BITCNT	; BUMP BIT COUNT
6056	026332	005705			TST	R5	; CHECK IF LAST BIT OF THIS WORD	
6057	026334	100402			BMI	8\$; HAS BEEN SIMULATED - YES - SKIP	
6058	026336	006305			ASL	R5	; SHIFT TO NEXT BIT	
6059	026340	000716			BR	7\$; GO DO THIS BIT	
6060								

B10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZ86E-A MACY11 271006, 05-OCT-76 09:12 PAGE 118
 DZ86EA.P11 28-SEP-76 15:31 DIAGNOSTIC MODE HEADER READ AND COMPARE

SEG 0118

```

6061 026342 005737 001204      8$:  TST      STMP1      ;TEST IF PASS FOR 1ST WORD JUST DONE
6062 026346 001004              BNE      9$        ;NO - SKIP
6063 026350 010100              MOV      R1,R0     ;PUT WORD 2 IN R0 FOR SIMULATION
6064 026352 005237 001204      INC      STMP1     ;BUMP PASS COUNT
6065 026356 000705              BR       11$
6066
6067 026360 023727 001204 000002  9$:  CMP      STMP1,#2   ;JUST DONE WITH WORD 2?
6068 026366 002004              BGE     10$       ;NO SKIP
6069 026370 010300              MOV      R3,R0     ;PUT 3RD WORD IN R0
6070 026372 005237 001204      INC      STMP1     ;BUMP PASS COUNT
6071 026376 000675              BR       11$
6072
6073 026400 023727 001204 000003 10$:  CMP      STMP1,#3   ;JUST DONE WITH WORD 3?
6074 026406 002006              BGE     16$       ;NO - SKIP
6075 026410 005000              CLR      R0        ;PUT IN WORD 0 OF GAP
6076 026412 012705 100000      MOV      #BIT15,R5 ;SET TO GIVE IT 1ST GAP BIT
6077 026416 005237 001204      INC      STMP1     ;BUMP PASS COUNT
6078 026422 000665              BR       7$
6079
6080 026424 023727 001204 000004 16$:  CMP      STMP1,#4   ;TEST IF READY FOR GAP
6081 026432 001445              BEQ     12$       ;YES - SKIP
6082
6083          ;
6084          ; THE FOLLOWING BLOCK OF CODE COMPUTES THE NUMBER OF CLOCK
6085          ; PULSES (WHICH ARE ACTUALLY CALLS TO RDBIT) THE OPERATION
6086          ; BEING PERFORMED REQUIRES TO UPDATE THE ADDRESS COUNTERS
6087          ; (RKDC AND RKDA). THIS CALCULATION IS BASED ON THE WHAT WAS
6088          ; IN THESE COUNTERS WHEN THE OPERATION STARTED (L.DC AND L.DA).
6089
6089 026434 012701 000025      MOV      #25,R1    ;PRESET R1 FOR 24 SECTOR MODE
6090 026440 032737 010000 002260  BIT      #CFMT,L.CS1 ;IS IT 24 SECTOR MODE OPERATION?
6091 026446 001402              BEQ     22$       ;YES - SKIP
6092 026450 012701 000023      MOV      #23,R1    ;ELSE CHANGE FOR 22 SECTOR MODE
6093 026454 032737 002000 002260 22$:  BIT      #CDT,L.CS1 ;RK06?
6094 026462 001403              BEQ     21$       ;YES - SKIP
6095 026464 052701 002000      BIS      #BIT10,R1 ;ELSE SET FOR MAX HEAD OF 4
6096 026470 000402              BR       20$     ;SKIP
6097 026472 052701 001000      BIS      #BIT9,R1  ;SET FOR MAX HEAD OF 2
6098 026476 023701 002266      CMP      L.DS,R1   ;TEST IF TRACK/SECTOR ADDRESS WAS MAX
6099 026502 001003              BNE     17$       ;NO - SKIP
6100 026504 012705 000100      MOV      #BIT6,R5  ;SET FOR 8 BITS OF GAP TO UPDATE
6101 026510 000410              BR       19$
6102 026512 123701 002266      CMPB    L.DS,R1    ;TEST IF SECTOR WAS MAX
6103 026516 001003              BNE     18$       ;NO - SKIP
6104 026520 012705 000400      MOV      #BIT8,R5  ;SET FOR 6 BITS OF GAP TO UPDATE
6105 026524 000402              BR       19$
6106 026526 012705 010000      MOV      #BIT12,R5 ;SET TO GIVE 1ST 4 BITS OF GAP
6107 026532 005237 001204      INC      STMP1     ;BUMP PASS COUNT
6108 026536 042737 100000 002244  BIC      #RDGATE,E.MR1 ;CLEAR READ GATE
6109 026544 000614              BR       7$
6110          ;
6111          ; END OF BLOCK
6112
6112 026546 004437 034154      12$:  JSR      R4,GETREG ;GET RK611 REGISTERS
6113 026552 005724              TST     (R4)+      ;BUMP TO NEXT ERROR RETURN
6114 026554 013737 002260 002220  MOV      L.CS1,E.CS1 ;SET EXPECTED CS1
6115 026562 023737 002160 002220  CMP      T.CS1,E.CS1 ;CHECK IF CORRECT
6116 026570 001232              BNE     4$        ;NO SKIP
  
```

C10

```

6117
6118 026572 005724          TST      (R4)+          ;BUMP TO NEXT ERROR RETURN
6119 026574 013737 002274 002240  MOV      L.DCYL,E.DCYL ;GET STARTING CYLINDER
6120 026602 013737 002266 002226  MOV      L.DA,E.DA    ;          TRACK/SECTOR
6121 026610 105237 002226          INCB     E.DA          ;BUMP SECTOR
6122 026614 012700 000026          MOV      #26,RO       ;PRESET FOR 26 SEC/TRACK
6123 026620 032737 010000 002260  BIT      #CFMT,L.CS1   ;TEST IF IN 26 SECTOR MODE
6124 026626 001402          BEQ      13$          ;YES - SKIP
6125 026630 012700 000024          MOV      #24,RO       ;ELSE CHANGE FOR 24 SECTOR MODE
6126 026634 120037 002226          CMPB    RO,E.DA       ;CHECK IF SECTOR TO LARGE
6127 026640 001023          BNE     15$          ;NO - SKIP
6128 026642 105037 002226          CLRB   E.DA          ;SET SECTOR TO 0
6129 026646 105237 002227          INCB   E.DA+1        ;BUMP TRACK
6130 026652 012700 000003          MOV      #3,RO        ;PRESET FOR 3 TRACKS (RK06)
6131 026656 032737 002000 002260  BIT      #COT,L.CS1   ;TEST IF IT IS RK06
6132 026664 001402          BEQ      14$          ;YES - SKIP
6133 026666 012700 000005          MOV      #5,RO        ;ELSE CHANGE
6134 026672 123700 002227          CMPB    E.DA+1,RO     ;CHECK IF TRACK INC TO LARGE
6135 026676 001004          BNE     15$          ;NO - SKIP
6136
6137 026700 105037 002227          CLRB   E.DA+1        ;CLEAR TRACK TO 0
6138 026704 005237 002240          INC     E.DCYL        ;BUMP CYLINDER
6139
6140 026710 023737 002240 002200 15$:  CMP      E.DCYL,T.DCYL ;CHECK IF CYL OK
6141 026716 001004          BNE     25$          ;NO - SKIP
6142 026720 023737 002226 002166  CMP      E.DA,T.DA    ;CHECK IF DA OK
6143 026726 001402          BEQ     26$          ;
6144 026730 000137 026256          JMP     4$           ;
6145
6146 026734 013737 002240 002274 26$:  MOV      E.DCYL,L.DCYL ;STORE COMPUTED NEXT CYL
6147 026742 013737 002226 002266  MOV      E.DA,L.DA    ;AND NEXT DA
6148
6149 026750 005724          TST      (R4)+          ;BUMP TO NO ERROR RETURN
6150 026752 000137 026314          JMP     5$           ;GO TO GOOD EXIT
6151
6152 026756 032777 001000 152154 50$:  BIT      #BIT9,DSWR   ;TEST IF LOOP ON ERROR
6153 026764 001404          BEQ     51$          ;NO - SKIP
6154 026766 005037 001236          CLR     $ESCAPE      ;CLEAR TO LOOP ON ERROR
6155 026772 000137 026314          JMP     5$           ;
6156
6157 026776 010446          51$:  MOV      R4,-(SP)     ;SET STACK TO CONTINUE TEST
6158 027000 000204          RTS      R4
6159
6160          .SBTTL  DIAGNOSTIC MODE GAP READ AND SYNC WRITE
6161          *      THE READING OF THE GAP AND WRITING OF THE SYNC (OR PREAMBLE)
6162          *      IS HANDLED IN THIS ROUTINE.
6163          *
6164          *      THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
6165          *      ROUTINE AND THE REMAINDER ARE PROCESSED HERE. THEN A
6166          *      CHECK IS MADE THAT WRITE GATE CAME ON AND 255 "0" BITS
6167          *      AND A SINGLE "1" BIT IS WRITTEN.
6168          *
6169          *      CALL:
6170          *      JSR      R4,DWGPSN
6171          *      ERROR  15      ;MR1 IN ERROR IN READ GAP
6172          *      ERROR  16      ;CS1 IN ERROR AFTER READ GAP
  
```

D10

```

6173      : *      ERROR 20      ;CSI IN ERROR AFTER SYNC WRITE
6174      : *
6175      : *      ROUTINE CALLED:
6176      : *      JSR      PC,WRTBIT
6177      : *      RETURN POINT IF ERROR IN WRTBIT
6178      : *      NO ERROR RETURN
6179
6180      027002      010046      DRIPSN:
6181      027002
6182
6183
6184
6185      027004      005737      002266
6186      027010      001006
6187      027012      012700      000066
6188      027016      012737      000011      002326
6189      027024      000416
6190      027026      105737      002266      9$:
6191      027032      001006
6192      027034      012700      000070
6193      027040      012737      000007      002326
6194      027046      000405
6195      027050      012737      000005      002326      9$:
6196      027056      012700      000074
6197
6198
6199      027062      012737      022040      002244      10$:
6200      027070      005037      002320
6201      027074      005037      002322
6202      027100      004737      033414
6203      027104      016237      000026      002204      1$:
6204      027112      023737      002244      002204
6205      027120      001404
6206      027122      012737      027144      001236
6207      027130      000513
6208      027132      005237      002326      11$:
6209      027136      005300
6210      027140      001357
6211      027142      000421
6212
6213      027144      005037      001236      2$:
6214      027150      022777      001000      1517E2
6215      027156      001011
6216      027160      013700      001254
6217      027164      006300
6218      027166      016037      035270      001236
6219      027174      162737      000002      001236
6220
6221      027202
6222      027202      012600
6223      027204      000204
6224
6225      027206      004437      034154      4$:
6226      027212      005724
6227      027214      013737      002260      002220
6228      027222      023737      002160      002220
6229      027230      001345

```

```

MOV      R0, -(SP)      ;; PUSH R0 ON STACK
THE FOLLOWING CODE ADJUST THE COUNT FOR THE REMAINDER OF
THE GAP. THE ADJUSTMENT DEPENDS ON HOW MANY GAP BITS WERE
READ IN THE HEADER COMPARE ROUTINE FOR ADDRESS REGISTER UPDATE.
TST      L,DA      ;; TEST IF DESIRED TRACK/SECTOR NOW 0
BNE      B$      ;; NO - SKIP
MOV      #54, R0      ;; ELSE ADJUST COUNT FOR CYL CHANGE
MOV      #11, BITCNT      ;; PRESET BIT COUNT FOR REST OF GAP
BR       10$
TSTB     L,DS      ;; TEST IF NEW SECTOR IS 0
BNE      9$      ;; NO - SKIP
MOV      #56, R0      ;; ELSE ADJUST COUNT FOR TRACK CHANGE
MOV      #7, BITCNT      ;; PRESET BIT COUNT FOR REST OF GAP
BR       10$
MOV      #5, BITCNT      ;; PRESET BIT COUNTER FOR REST OF GAP
MOV      #60, R0      ;; SET COUNT FOR 60 BITS
END OF BLOCK

MOV      #DMD!ECCW!MEWD, E.MR1      ;; SET EXPECTED MR1
CLR      PR.BIT      ;; CLEAR PRESENT BIT
CLR      MI.BIT      ;; CLEAR PREVIOUS BIT
JSR      PC,RDBIT      ;; GO READ BIT
MOV      R4,MR1(R2), T.MR1      ;; GET MR1
CMP      E.MR1, T.MR1      ;; CHECK IF CORRECT
BEQ      11$      ;; YES - SKIP
MOV      #25, $ESCAPE      ;; SET RETURN TO CONT TEST
BR       50$
INC      BITCNT      ;; BUMP BIT COUNT
DEC      R0      ;; DEC LOOP COUNT
BNE      1$      ;; LOOP IF NOT ZERO
BR       4$      ;; ELSE SKIP

CLR      $ESCAPE      ;; CLEAR TO EXIT OR LOOP
CMP      #BIT9, $SWR      ;; ERROR RETURN: CHECK IF LOOP ON ERROR
BNE      3$      ;; YES - SKIP
MOV      $TESTN, R0      ;; GET NUMBER OF NEXT TEST
ASL      R0      ;; SHIFT FOR INDEXING
MOV      $SWOBTB(R0), $ESCAPE      ;; SET ESCAPE FOR GOT TO NEXT TEST
SJB      #2, $ESCAPE      ;; SET TO FIRST INST (SCOPE)

MOV      (SP)+, R0      ;; POP STACK INTO R0
RTS      R4

JSR      R4, GETREG      ;; GET RK611 REGS
TST      (R4)+      ;; BUMP TO NEXT ERROR RETURN
MOV      L.CSI, E.CSI      ;; SET EXPECTED CSI
CMP      T.CSI, E.CSI      ;; CHECK IF CSI OK
BNE      2$      ;; NO - SKIP

```


E10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZ6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 121
 DZ6EA.P11 28-SEP-76 15:31 DIAGNOSTIC MODE GAP READ AND SYNC WRITE

SEQ 0121

```

6229
6230 027232 005724          TST      (R4)+          ;BUMP RETURN POINTER
6231
6232 027234 005037 002326    CLR      BITCNT        ;CLEAR BIT COUNTER
6233 027240 005037 002322    CLR      M1.BIT        ;CLEAR ALL WRITE BIT INDICATORS
6234 027244 005037 002324    CLR      M2.BIT
6235 027250 005037 002316    CLR      P1.BIT
6236 027254 005037 002320    CLR      PR.BIT
6237 027260 012737 062040 002244  MOV      #DMD!ECCW!MEWD!WRTGATE,E.MR1 ;SET EXPECTED MR1
6238 027266 012700 000400    MOV      #256.,R0      ;SET COUNT TO WRITE 256 "0"
6239
6240 027272 004737 032450    5$:     JSR      PC,WRTBIT ;CALL WRITE BIT
6241 027276 000405          BR       6$           ;ERROR RETURN - SKIP TO HANDLER
6242 027300 005237 002326    INC      BITCNT        ;BLMP BIT COUNT
6243 027304 005300          DEC      R0            ;LOOP UNTIL ALL 256
6244 027306 001371          BNE     5$           ;BITS ARE WRITTEN
6245 027310 000402          BR       7$
6246
6247 027312 010446          6$:     MOV      R4,-(SP)     ;WRITE ERROR HANDLER-SET
6248 027314 000204          RTS     R4            ;STACK FOR REPORTING SUBSEQUENT ERRORS
6249
6250 027316 012737 000001 002316  7$:     MOV      #1,P1.BIT    ;SET SYNC BIT OF 1 AS NEXT BIT
6251 027324 004737 032450    JSR      PC,WRTBIT    ;GO WRITE IT
6252 027330 000770          BR       6$           ;ERROR RETURN
6253 027332 005237 002326    INC      BITCNT        ;BUMP BIT COUNT
6254
6255 027336 004437 034154    JSR      R4,GETREG    ;GET RK611 REGS
6256 027342 005724          TST      (R4)+        ;BUMP TO NEXT ERROR RETURN
6257 027344 023737 002160 002220  CMP      T.CS1,E.CS1  ;CHECK IF CORRECT
6258 027352 001274          BNE     2$           ;NO - SKIP
6259
6260 027354 005724          TST      (R4)+        ;BUMP PAST ERROR RETURNS
6261 027356 000711          BR       3$           ;GO TO EXIT
6262
6263 027360 032777 001000 151552  50$:    BIT      #BIT9,DSWP   ;TEST IF LOOP ON ERROR
6264 027366 001403          BEQ     51$          ;YES - SKIP
6265 027370 005037 001236    CLR      $ESCAPE      ;ELSE SET TO LOOP
6266 027374 000702          BR       3$
6267 027376 010446          51$:    MOV      R4,-(SP)     ;PRESET STACK FOR RETURN
6268 027400 000204          RTS     R4
6269
6270          .SBTTL  DIAGNOSTIC WRITE DATA AND ECC ROUTINE
6271          *      THE DATA IN THE OUPUT BUFFER IS WRITE SIMULATED, ECC IS
6272          *      GENERATED AND WRITTEN, AND THE POSTAMBLE IS WRITTEN.
6273          *
6274          *      CALL:
6275          *      JSR      R4,DWRITE
6276          *      ERROR 17      ;MR1 IN ERROR IN WRITING
6277          *      ERROR 21      ;ECC ERROR IN WRITING
6278          *      ERROR 22      ;CS1 ERROR AFTER WRITE
6279          *      NO ERROR RETURN
6280          *
6281          *      ROUTINE CALLED:
6282          *      JSR      PC,WRTBIT
6283          *      RETURN POINT IF ERROR IN WRTBIT
6284          *      NO ERROR RETURN FROM WRTBIT
  
```

F10

```

6285 027402          DWRITE:
6286 027402 010046      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
6287 027404 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
6288 027406 010346      MOV    R3,-(SP)      ;;PUSH R3 ON STACK
6289 027410 010546      MOV    R5,-(SP)      ;;PUSH R5 ON STACK
6290 027412 005037 002326  CLR    BITCNT      ;;CLEAR BIT COUNTER
6291 027416 005037 002342  CLR    ECCLO       ;;CLEAR ECC LOW
6292 027422 005037 002340  CLR    ECCHI       ;;CLEAR ECC HI
6293 027426 012700 000400  MOV    #400,R0      ;;SET WORD COUNT FOR ONE SECTOR
6294 027432 012703 051444  MOV    #0BUFF,R3    ;;GET POINTER TO OUTPUT DATA
6295 027436 010437 001216  MOV    R4,$TMP6     ;;STORE MRI ERROR RETURN
6296 027442 005724          TST    (R4)+        ;;BUMP TO NEXT ERROR RETURN
6297 027444 010437 001220  MOV    R4,$TMP7     ;;STORE ECC ERROR RETURN
6298
6299
6300
6301
6302
6303 027450 005037 001214      CLR    $TMP5       ;;CLEAR SWITCH
6304 027454 012701 000001 3$:  MOV    #BIT0,R1    ;;LOAD BIT POINTER
6305 027460 012305          MOV    (R3)+,R5    ;;GET A WORD FOR WRITING
6306
6307 027462 013737 002322 002324 4$:  MOV    M1.BIT,M2.BIT ;;SHIFT CURRENT-1 TO CURRENT-2
6308 027470 013737 002320 002322  MOV    PR.BIT,M1.BIT ;;PRESENT TO CURRENT-1
6309 027476 013737 002316 002320  MOV    P1.BIT,PR.BIT ;;NEXT TO PRESENT
6310 027504 005037 002316      CLR    P1.BIT      ;;CLEAR NEXT BIT
6311 027510 030105          BIT    R1,R5       ;;TEST IF NEXT BIT IS ONE
6312 027512 001403          BEQ    $S         ;;NO - SKIP
6313 027514 012737 000001 002316  MOV    #1,P1.BIT    ;;ELSE SET NEXT BIT
6314 027522 004737 032450 5$:  JSR    PC,WRTBIT    ;;GO WRITE BIT
6315 027526 000441          BR     33$         ;;ERROR RETURN - SKIP
6316 027530 004737 032260 43$:  JSR    PC,ECCGEN    ;;GO GENERATE ECC FOR THIS BIT
6317 027534 016237 000032 002214  MOV    RKECPT(R2),T.ECPT ;;GET PATTERN
6318 027542 023737 002254 002214  CMP    E.ECPT,T.ECPT ;;CHECK IF ECC CORRECT
6319 027550 001402          BEQ    40$        ;;YES - SKIP
6320 027552 000137 030150          JMP    13$        ;;JUMP TO ERROR EXIT
6321
6322 027556 005237 002326 40$:  INC    BITCNT      ;;BUMP BIT COUNT
6323 027562 005701          TST    R1         ;;TEST BIT POINTER
6324 027564 100424          BMI    7$        ;;IF NEGATIVE - SKIP
6325 027566 006301          ASL    R1         ;;ELSE SHIFT LEFT AND LOOP
6326 027570 020027 000001  CMP    R0,#1      ;;TEST IF WRITING LAST WORD
6327 027574 001332          BNE    4$        ;;NO - GO TO LOOP
6328 027576 032737 010000 002260  BIT    #CFMT,L.CS1 ;;TEST IF 22 SECTOR MODE
6329 027604 001404          BEQ    32$        ;;NO - SKIP
6330 027606 005737 001214          TST    $TMP5     ;;TEST IF 18 BIT SWITCH SET
6331 027612 001001          BNE    32$        ;;YES - SKIP
6332 027614 000722          BR     4$        ;;ELSE LOOP
6333 027616 005701 32$:  TST    R1         ;;ELSE TEST IF LAST BIT OF LAST WORD
6334 027620 100320          BPL    4$        ;;NO - GO TO LOOP
6335 027622 042737 020000 002244  BIC    #ECCW,E.MR1 ;;ELSE SET MRI FOR ECC WRITE
6336 027630 000714          BR     4$
6337 027632 000137 030172 33$:  JMP    20$
6338
6339 027636 032737 010000 002260 7$:  BIT    #CFMT,L.CS1 ;;TEST IF 22 SECTOR MODE
6340 027644 001413          BEQ    31$        ;;NO - SKIP

```

G10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PE MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 123
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC WRITE DATA AND ECC ROUTINE

SEQ 0123

6341	027646	005737	001214		TST	\$TMP5		; TEST IF 18 BIT SWITCH SET
6342	027652	001006			BNE	30\$; YES - SKIP
6343	027654	012701	040000		MOV	#BIT14,R1		; SET TO SUPPLY 2 MORE BITS (16 & 17)
6344	027660	005005			CLR	R5		; MAKE SURE THEY ARE 0'S
6345	027662	005137	001214		COM	\$TMP5		; SET THE 18 BIT SWITCH
6346	027666	000675			BR	4\$; GO WRITE THE BITS
6347								
6348	027670	005037	001214	30\$:	CLR	\$TMP5		; CLEAR THE SWITCH
6349	027674	005300		31\$:	DEC	R0		; DEC WORD COUNT
6350	027676	001266			BNE	3\$; IF NOT ZERO - GO GET NEXT WORD
6351	027700	005037	001214		CLR	\$TMP5		; CLEAR FOR PASS CONTROL
6352	027704	013705	002340		MOV	ECCHI,R5		; GET FIRST ECC WORD
6353	027710	012701	000001		MOV	#BIT0,R1		; SET BIT POINTER
6354								
6355	027714	013737	002322	002324	8\$:	MOV	M1.BIT,M2.BIT	; SHIFT BIT INDICATORS AGAIN
6356	027722	013737	002320	002322		MOV	PR.BIT,M1.BIT	
6357	027730	013737	002316	002320		MOV	P1.BIT,PR.BIT	
6358	027736	005037	002316		CLR	P1.BIT		; CLEAR NEXT BIT
6359	027742	030105			BIT	R1,R5		; TEST IF BIT IS ZERO
6360	027744	001403			BEQ	9\$; YES - SKIP
6361	027746	012737	000001	002316	MOV	#1,P1.BIT		; ELSE CHANGE TO ONE
6362	027754	004737	032450	9\$:	JSR	PC,WRTBIT		; GO WRITE BIT
6363	027760	000501			BR	14\$; WRTBIT ERROR - SKIP
6364								
6365	027762	005237	002326	41\$:	INC	BITCNT		; BUMP BIT COUNT
6366	027766	005701			TST	R1		; TEST BIT POINTER
6367	027770	100413			BMI	10\$; IF NEGATIVE - SKIP
6368	027772	006301			ASL	R1		; ELSE SHIFT AND LOOP
6369	027774	023727	001214	000001	CMP	\$TMP5,#1		; TEST IF WRITING LAST ECC WORD
6370	030002	001344			BNE	8\$; NO - SKIP TO LOOP
6371	030004	005701			TST	R1		; ELSE TEST IF WRITING LAST BIT OF LAST ECC
6372	030006	100342			BPL	8\$; NO - SKIP TO LOOP
6373	030010	052737	020000	002244	BIS	#ECCW,E.MR1		; ELSE SET ECCW IN MR1
6374	030016	000736			BR	8\$		
6375								
6376	030020	005737	001214	10\$:	TST	\$TMP5		; TEST PASS 0 JUST DONE
6377	030024	001007			BNE	11\$; NO - SKIP
6378	030026	005237	001214		INC	\$TMP5		; BUMP PASS COUNT
6379	030032	012701	000001		MOV	#BIT0,R1		; PRESET BIT POINTER
6380	030036	013705	002342		MOV	ECCL0,R5		; GET SECOND ECC WORD
6381	030042	000724			BR	8\$; LOOP
6382								
6383	030044	012701	000017	11\$:	MOV	#15.,R1		; SET A BIT COUNT
6384								
6385	030050	013737	002322	002324	12\$:	MOV	M1.BIT,M2.BIT	; SHIFT BIT INDICATORS
6386	030056	013737	002320	002322		MOV	PR.BIT,M1.BIT	
6387	030064	013737	002316	002320		MOV	P1.BIT,PR.BIT	
6388	030072	005037	002316		CLR	P1.BIT		; SET ZERO FOR WRITE POSTAMBLE
6389	030076	004737	032450		JSR	PC,WRTBIT		; GO WRITE BIT
6390	030102	000436			BR	15\$; WRITE BIT ERROR - SKIP
6391	030104	005237	002326	42\$:	INC	BITCNT		; BUMP BIT COUNT
6392	030110	005301			DEC	R1		; DEC BIT COUNT
6393	030112	001356			BNE	12\$; LOOP UNTTIL BIT COUNT 0
6394								
6395	030114	004437	034154		JSR	R4,GETREG		; GET RK611 REGS
6396	030120	013704	001220		MOV	\$TMP7,R4		; SET R4 FOR ECC ERROR RETURN

H10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 124
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC WRITE DATA AND ECC ROUTINE

SEQ 0124

```

6397 030124 005724          TST   (R4)+          ;BUMP TO CS1 ERROR RETURN
6398 030126 013737 002260 002220  MOV   L.CS1,E.CS1    ;GET EXPECTED CS1
6399
6400 030134 023737 002160 002220  CMP   T.CS1,E.CS1    ;CHECK IF CORRECT
6401 030142 001031          BNE   18$            ;NO - SKIP
6402 030144 005724          TST   (R4)+          ;BUMP TO GOOD RETURN
6403 030146 000446          BR    19$            ;GO TO RETURN
6404
6405 030150 013704 001220          MOV   $TMP7,R4       ;SET FOR ECC ERROR RETURN
6406 030154 012737 027556 001236  MOV   #40$,$ESCAPE   ;STORE RETURN POINT FOR NO LOOP ON ERROR
6407 030162 000410          BR    16$
6408
6409 030164 013704 001216          MOV   $TMP6,R4       ;SET FOR MRI ERROR RETURN
6410 030170 000405          BR    16$
6411
6412 030172 013704 001216          MOV   $TMP6,R4       ;SET FOR MRI ERROR RETURN
6413 030176 000402          BR    16$
6414
6415 030200 013704 001216          MOV   $TMP6,R4       ;SET FOR MRI ERROR RETURN
6416
6417 030204 032777 001000 150726  BIT   #BIT9,$SWR     ;CHECK IF LOOP ON ERROR
6418 030212 001403          BEQ   17$            ;NO - SKIP
6419
6420 030214 005037 001236          CLR   $ESCAPE        ;CLEAR ESCAPE
6421 030220 000421          BR    19$            ;SKIP TO EXIT
6422
6423 030222 010446          MOV   R4,-(SP)       ;CONDITION STACK
6424 030224 000204          RTS   R4              ;GO REPORT ERROR
6425
6426 030226 005037 001236          CLR   $ESCAPE        ;CLEAR FOR LOOP OR EXIT
6427 030232 032777 001000 150700  BIT   #BIT9,$SWR     ;CHECK IF LOOP ON ERROR
6428 030240 001011          BNE   19$            ;YES - SKIP
6429 030242 013700 001254          MOV   $TESTN,R0      ;SET UP TO SKIP TO NEXT TEST
6430 030246 006300          ASL   R0
6431 030250 016037 035270 001236  MOV   $$W08TB(R0),$ESCAPE
6432 030256 162737 000002 001236  SUB   #2,$ESCAPE
6433 030264          19$:
6434 030264 012605          MOV   (SP)+,R5       ;;POP STACK INTO R5
6435 030266 012603          MOV   (SP)+,R3       ;;POP STACK INTO R3
6436 030270 012601          MOV   (SP)+,R1       ;;POP STACK INTO R1
6437 030272 012600          MOV   (SP)+,R0       ;;POP STACK INTO R0
6438 030274 000204          RTS   R4
6439
6440          .SBTTL  DIAGNOSTIC MODE READ GAP AND DATA SYNC
6441          ;* THE READING OF THE GAP AND READING OF THE SYNC (OR PREAMBLE)
6442          ;* IS HANDLED IN THIS ROUTINE.
6443          ;*
6444          ;* THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
6445          ;* ROUTINE AND THE REMAINING GAP BITS ARE PROCESSED HERE.
6446          ;* THEN A CHECK IS MADE THAT READ GATE CAME ON
6447          ;* AT THE 129TH BIT OF SYNC.*
6448          ;* CALL:
6449          ;* JSR   R4,DRGPSN
6450          ;* ERROR 15 ;MRI IN ERROR IN READ GAP
6451          ;* ERROR 16 ;CSI IN ERROR AFTER READ GAP
6452          ;* ERROR 32 ;MRI IN ERROR IN READ SYNC

```

```

6453          ;*      ERROR 33      ;CS1 IN ERROR AFTER SYNC READ
6454          ;*
6455          ;*      ROUTINE CALLED:
6456          ;*      JSR      PC,RDBIT
6457
6458 030276          DRGPSN:
6459 030276 010046  MOV      RD,-(SP)          ;;PUSH RD ON STACK
6460          ; THE FOLLOWING CODE ADJUST THE COUNT FOR THE REMAINDER OF
6461          ; THE GAP. THE ADJUSTMENT DEPENDS ON HOW MANY GAP BITS WERE
6462          ; READ IN THE HEADER COMPARE ROUTINE FOR ADDRESS REGISTER UPDATE.
6463 030300 005737 002266  TST      L.DA          ;TEST IF DESIRED TRACK/SECTOR NOW 0
6464 030304 001096          BNE      8$          ;NO - SKIP
6465 030306 012700 000066  MOV      #54,RD          ;ELSE ADJUST COUNT FOR CYL CHANGE
6466 030312 012737 000011 002326  MOV      #11,BITCNT     ;PRESET BIT COUNT FOR REST OF GAP
6467 030320 000416          BR       10$
6468 030322 105737 002266 8$:  TSTB   L.DS          ;TEST IF NEW SECTOR IS 0
6469 030326 001096          BNE      9$          ;NO - SKIP
6470 030330 012700 000070  MOV      #56,RD          ;ELSE ADJUST COUNT FOR TRACK CHANGE
6471 030334 012737 000007 002326  MOV      #7,BITCNT     ;PRESET BIT COUNT FOR REST OF GAP
6472 030342 000405          BR       10$
6473 030344 012737 000005 002326 9$:  MOV      #5,BITCNT     ;PRESET BIT COUNTER FOR REST OF GAP
6474 030352 012700 000074          MOV      #60,RD          ;SET COUNT FOR 60 BITS
6475          ;
6476          ;
6477 030356 012737 022040 002244 10$:  MOV      #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
6478 030364 005037 002320          CLR      PR.BIT        ;CLEAR PRESENT BIT
6479 030370 005037 002322          CLR      M1.BIT        ;CLEAR PREVIOUS BIT
6480 030374 004737 033414          JSR      PC,RDBIT      ;GO READ BIT
6481 030400 016237 000026 002204 1$:  MOV      RKMRI(R2),T.MR1 ;GET MR1
6482 030406 023737 002244 002204  CMP      E.MR1,T.MR1    ;CHECK IF CORRECT
6483 030414 001404          BEQ     11$          ;YES - SKIP
6484 030416 012737 030440 001236  MOV      #2,$ESCAPE    ;SET RETURN TO CONT TEST
6485 030424 000552          BR       50$
6486 030426 005237 002326          11$:  INC      BITCNT        ;BUMP BIT COUNT
6487 030432 005300          DEC      RD           ;DEC LOOP COUNT
6488 030434 001357          BNE     1$           ;LOOP IF NOT ZERO
6489 030436 000421          BR       4$           ;ELSE SKIP
6490
6491 030440 005037 001236          2$:  CLR      $ESCAPE      ;CLEAR TO EXIT OR LOOP
6492 030444 022777 001000 150466  CMP      #BIT9,$SWR    ;ERROR RETURN: CHECK IF LOOP ON ERROR
6493 030452 001011          BNE     3$           ;YES - SKIP
6494 030454 013700 001254          MOV      $TESTN,RD    ;GET NUMBER OF NEXT TEST
6495 030460 006300          ASL     RD           ;SHIFT FOR INDEXING
6496 030462 016037 035270 001236  MOV      $$SWOBTB(RD),$ESCAPE ;SET ESCAPE FOR GOT TO NEXT TEST
6497 030470 162737 000002 001236  SUB     #2,$ESCAPE    ;SET TO FIRST INST (SCOPE)
6498          3$:
6499 030476 012600          MOV     (SP)+,RD      ;;POP STACK INTO RD
6500 030500 000204          RTS     R4
6501
6502 030502 004437 034154          4$:  JSR     R4,GETREG     ;GET RK611 REGS
6503 030506 005724          TST     (R4)+        ;BUMP TO NEXT ERROR RETURN
6504 030510 013737 002260 002220  MOV     L.CS1,E.CS1   ;SET EXPECTED CS1
6505 030516 023737 002160 002220  CMP     T.CS1,E.CS1   ;CHECK IF CS1 OK
6506 030524 001345          BNE     2$          ;NO - SKIP
6507
6508 030526 005724          TST     (R4)+        ;BUMP RETURN POINTER
    
```

J10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PS MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 126
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC MODE READ GAP AND DATA SYNC

SEG 0126

```

6509
6510 030530 005037 002326          CLR      BITCNT          ;CLEAR BIT COUNTER
6511 030534 012737 022040 002244    MOV      #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
6512 030542 012700 000200          MOV      #128.,RO        ;SET COUNT TO READ 128 "0"
6513
6514 030546 004737 033414          5$:     JSR      PC,RDBIT      ;CALL READ BIT
6515 030552 016237 000026 002204    MOV      RKMRI(R2),T.MR1    ;GET MR1 FOR TESTING
6516 030550 023737 002244 002204    CMP      E.MR1,T.MR1      ;TEST IF CORRECT
6517 030566 001404          BEQ      22$              ;YES - SKIP
6518
6519 030570 012737 030600 001236    MOV      #22$, $ESCAPE     ;SET RETURN FOR CONTINUE TEST
6520 030576 000465          BR       50$              ;GO TO ERROR EXIT
6521 030600 005237 002326          22$:    INC      BITCNT          ;BUMP BIT COUNT
6522 030604 005300          DEC      RO                ;LOOP UNTIL ALL 128
6523 030606 001357          BNE      5$                ;BITS ARE READ
6524 030610 012700 000177          MOV      #127.,RO        ;SET COUNT FOR REST OF SYNC
6525 030614 052737 100000 002244    BIS      #RDGATE,E.MR1     ;SET EXPECTED MR1
6526 030622 004737 033414          6$:     JSR      PC,RDBIT      ;GO CALL READ BIT
6527 030626 016237 000026 002204    MOV      RKMRI(R2),T.MR1    ;GET MR1 FOR TESTING
6528 030634 023737 002244 002204    CMP      E.MR1,T.MR1      ;TEST IF CORRECT
6529 030642 001404          BEQ      23$              ;YES - SKIP
6530 030644 012737 030654 001236    MOV      #23$, $ESCAPE     ;ELSE SET ESCAPE
6531 030652 000437          BR       50$              ;GO TO ERROR REPORT EXIT
6532
6533 030654 005237 002326          23$:    INC      BITCNT          ;BUMP BIT COUNTER
6534 030660 005300          DEC      RO                ;DEC BIT COUNT
6535 030662 001357          BNE      6$                ;LOOP UNTIL ALL 127 READ
6536
6537 030664 012737 000001 002320    MOV      #1,PR.BIT        ;SET SYNC BIT OF 1 AS NEXT BIT
6538 030672 004737 033414          JSR      PC,RDBIT      ;GO READ IT
6539 030676 016237 000026 002244    MOV      RKMRI(R2),E.MR1    ;GET MR1 FOR TESTING
6540 030704 023737 002244 002204    CMP      E.MR1,T.MR1      ;CHECK IF CORRECT
6541 030712 001404          BEQ      24$              ;YES - SKIP
6542 030714 012737 030724 001236    MOV      #24$, $ESCAPE     ;ELSE SET RETURN FOR CONTINUE TEST
6543 030722 000413          BR       50$
6544
6545 030724 005237 002326          24$:    INC      BITCNT          ;BUMP BIT COUNT
6546
6547 030730 004437 034154          JSR      R4,GETREG        ;GET RK611 REGS
6548 030734 005724          TST      (R4)+            ;BUMP TO NEXT ERROR RETURN
6549 030736 023737 002160 002220    CMP      T.CS1,E.CS1      ;CHECK IF CORRECT
6550 030744 001235          BNE      2$                ;NO - SKIP
6551
6552 030746 005724          TST      (R4)+            ;BUMP PAST ERROR RETURNS
6553 030750 000652          BR       3$                ;GO TO EXIT
6554
6555 030752 032777 001000 150160 50$:    BIT      #BIT9, $SWR      ;TEST IF LOOP ON ERROR
6556 030760 001403          BEQ      51$              ;YES - SKIP
6557 030762 005037 001236          CLR      $ESCAPE          ;ELSE SET TO LOOP
6558 030766 000643          BR       3$
6559 030770 010446          51$:    MOV      R4,-(SP)        ;PRESET STACK FOR RETURN
6560 030772 000204          RTS      R4
6561 .SBTTL  DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE
6562 ;*      TH'S ROUTINE IS USED TO SIMULATE THE READ OR WRITE CHECK
6563 ;*      OF DATA. THE ROUTINE WILL TRANSFER A FULL OR PARTIAL SECTOR.
6564 ;*      THE FIRST WORD AFTER THE CALL SPECIFIES HOW MANY WORDS
  
```

K10

```

6565      ;*      ARE TO BE TRANSFERRED.
6566      ;*
6567      ;*      IF LESS THAN A FULL SECTOR, THE ECC AT THE END
6568      ;*      OF THE TRANSFER IS NOT CHECKED. THE ECC IS CHECKED AT THE TIME
6569      ;*      EACH BIT IS TRANSFERED.
6570      ;*
6571      ;*      EITHER 16 OR 18 BIT WORDS CAN BE SIMULATED
6572      ;*      DEPENDING ON THE STATE OF CFMT BIT IN L.CS1. IF 18 BITS
6573      ;*      ARE REQUIRED, BITS 16 AND 17 ARE ASSUMED TO BE ZERO.
6574      ;*
6575      ;*      THE CONTENTS OF LOCATION ECPOSX MUST BE LOADED WITH THE ECC
6576      ;*      COUNT EXPECTED AT THE END OF THE DATA TRANSFER. THIS
6577      ;*      LOADING MUST OCCUR BEFORE THE ROUTINE IS CALLED.
6578      ;*
6579      ;*      ANOTHER LOCATION, ECPATX, IS SET TO 0 IF THE ECC IS EXPECTED
6580      ;*      TO BE CORRECT AND SET TO 1 IF AN ERROR IS BEING FORCED.
6581      ;*
6582      ;*      A 3RD LOCATION, ECCSRC, SPECIFIES THE SOURCE OF THE ECC
6583      ;*      WORDS TO BE USED IN CHECKING THE OPERATION. IF ECCSRC IS A
6584      ;*      1, THE ECC WORDS FED INTO THE DECODER ARE TAKEN FROM THE
6585      ;*      LAST TWO LOCATIONS OF OBUFF, IF ECCSRC IS 0, THE COMPUTED
6586      ;*      ECC WORDS FROM ECCHI AND ECCLO ARE USED.
6587      ;*      CALL:
6588      ;*      JSR      R4,DREAD
6589      ;*      LENGTH OF TRANSFER
6590      ;*      ERROR 34      ;MR1 IN ERROR READING DATA OR ECC
6591      ;*      ERROR 35      ;ECC ERROR READING DATA
6592      ;*      ERROR 36      ;CS1 ERROR AFTER READING DATA OR ECC
6593      ;*      ERROR 41      ;ECC REGISTER INCORRECT AFTER READ ECC
6594      ;*      ERROR 42      ;ERR IN ECC PAT CALCULATION
6595      ;*      ERROR 43      ;ERR IN ECC POS COUNTING
6596      ;*
6597      ;*      OR
6598      ;*
6599      ;*      JSR      R4,DWRTCK
6600      ;*      LENGTH OF TRANSFER
6601      ;*      ERROR 53      ;MR1 IN ERROR WRITE CHK OR ECC READ
6602      ;*      ERROR 54      ;ECC ERROR IN WRITE CHECK
6603      ;*      ERROR 55      ;CS1 ERROR AFTER IN WRT CHK OR ECC READ
6604      ;*      ERROR 41      ;ECC REGISTER INCORRECT AFTER READ ECC
6605      ;*      ERROR 42      ;ERR IN ECC PAT CALCULATION
6606      ;*      ERROR 43      ;ERR IN ECC POS COUNTING
6607      ;*
6608      ;*      ROUTINES CALLED:
6609      ;*      RDBIT
6610      ;*      ECCGEN
6611      ;*
6612      ;*
6613      ;*      DWRTCK:
6614      ;*      DREAD:
6615      ;*      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
6616      ;*      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
6617      ;*      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
6618      ;*      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
6619      ;*      MOV      (R4)+,R0      ;STORE WORD COUNT
6620      ;*      MOV      R4,$TMP6      ;STORE MR1 ERROR RETURN

```

001216

L10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 128
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE

SEG 0128

6621	031012	005724				TST	(R4)+	;BUMP TO NEXT RETURN
6622	031014	010437	001220			MOV	R4,\$TMP7	;STORE ECC ERROR RETURN
6623	031020	005724				TST	(R4)+	;BUMP TO NEXT ERROR RETURN
6624	031022	010437	001210			MOV	R4,\$TMP3	;STORE CS1 ERROR RETURN
6625	031026	005724				TST	(R4)+	;BUMP TO NEXT ERROR RETURN
6626	031030	010437	001212			MOV	R4,\$TMP4	;STORE ECC REG ERROR RETURN
6627	031034	005724				TST	(R4)+	;BUMP TO NEXT ERROR RETURN
6628	031036	010437	001222			MOV	R4,\$TMP10	;STORE ECC PAT ERR RETURN
6629	031042	005724				TST	(R4)+	;BUMP TO NEXT ERROR RETURN
6630	031044	010437	001224			MOV	R4,\$TMP11	;SOTER ECC POS ERROR RETURN
6631	031050	012703	051444			MOV	#0BUFF,R3	;GET ADDRESS OF BLUFFER
6632	031054	005037	002326			CLR	BITCNT	;CLEAR BIT COUNTER
6633	031060	005037	002340			CLR	ECCHI	;CLEAR ECC COMPUTED LOCATIONS
6634	031064	005037	002342			CLR	ECCLO	
6635	031070	005037	001214			CLR	\$TMP5	;CLEAR FOR 18 BIT MODE SWITCH
6636	031074	032737	010000	002260		BIT	#CFMT,L.CS1	;TEST IF 18 BIT MODE
6637	031102	001412				BEQ	32\$;NO - SKIP
6638	031104	012737	011000	001226		MOV	#11000,\$TMP12	;ELSE SET UP BIT COUNTS FOR 18 BIT MODE
6639	031112	012737	011001	001230		MOV	#11001,\$TMP13	
6640	031120	012737	011041	001232		MOV	#11041,\$TMP14	
6641	031126	000411				BR	3\$	
6642	031130	012737	010000	001226	32\$:	MOV	#10000,\$TMP12	;SET BIT COUNTS FOR 16 BIT MODE
6643	031136	012737	010001	001230		MOV	#10001,\$TMP13	
6644	031144	012737	010041	001232		MOV	#10041,\$TMP14	
6645								
6646	031152	012701	000001		3\$:	MOV	#BIT0,R1	;PRESET BIT POINTER
6647	031156	012305				MOV	(R3)+,R5	;GET DATA WORD
6648								
6649	031160	012737	002320	002322	4\$:	MOV	PR.BIT,M1.BIT	;SHIFT PRESENT INTO PREVIOUS
6650	031166	005037	002320			CLR	PR.BIT	;CLEAR PRESENT
6651	031172	030105				BIT	R1,R5	;TEST IF PRESENT BIT IS 0
6652	031174	001403				BEQ	1\$;YES - SKIP
6653	031176	012737	000001	002320		MOV	#1,PR.BIT	;ELSE INSERT A 1
6654								
6655	031204	004737	033414		1\$:	JSR	PC,RDBIT	;GO READ BIT
6656	031210	016237	000026	002204		MOV	RKMR1(R2),T.MR1	;GET MR1
6657	031216	023737	002204	002244		CMP	T.MR1,E.MR1	;CHECK IF CORRECT
6658	031224	001402				BEQ	43\$;YES - SKIP
6659	031226	000137	032060			JMP	9\$	
6660								
6661	031232	005737	002326		43\$:	TST	BITCNT	;TEST IF FIRST BIT OF DATA
6662	031236	001413				BEQ	44\$;YES - SKIP ECC GEN
6663	031240	004737	032260			JSR	PC,ECCGEN	;GO GENERATE ECC
6664	031244	016237	000032	002214		MOV	RKECPT(R2),T.ECPT	;TEST IF ECC CORRECT
6665	031252	023737	002214	002254		CMP	T.ECPT,E.ECPT	
6666	031260	001402				BEQ	44\$	
6667	031262	000137	032074			JMP	10\$	
6668								
6669	031266	005237	002326		44\$:	INC	BITCNT	;BUMP BIT COUNTER
6670	031272	005701				TST	R1	;TEST IF THIS WORD IS DONE
6671	031274	100402				BMI	2\$;YES - SKIP
6672	031276	006301				ASL	R1	;ELSE SHIFT BIT POINTER
6673	031300	000727				BR	4\$	
6674								
6675	031302	032737	010000	002260	2\$:	BIT	#CFMT,L.CS1	;TEST IF 18 BIT MODE
6676	031310	001412				BEQ	30\$;NO - SKIP

M10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 129
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE

SEQ 0129

6677	031312	005737	001214		TST	\$TMP5		;TEST IF PROCESSING BIT 16 OR 17 DONE
6678	031316	001007			BNE	30\$;YES - SKIP
6679	031320	012701	040000		MOV	#BIT14,R1		;SET POINTER FOR 2 MORE BITS
6680	031324	013705	002354		MOV	HIBITS,R5		;INSERT HI ORDER BITS
6681	031330	005137	001214		COM	\$TMP5		;SET 18 BIT SWITCH
6682	031334	000711			BR	4\$		
6683	031336	005037	001214	30\$:	CLR	\$TMP5		;CLEAR 18 BIT SWITCH
6684								
6685	031342	005300		31\$:	DEC	RO		;DECREMENT WORD COUNT
6686	031344	001302			BNE	3\$;IF NOT ZERO - LOOP
6687								
6688	031346	023737	002326	001226	CMP	BITCNT,\$TMP12		;ELSE TEST IF FULL SECTOR WAS READ
6689	031354	001402			BEQ	33\$;YES - SKIP
6690	031356	000137	032032		JMP	8\$;ELSE JUMP TO EXIT - DO NOT CHECK ; ECC WORDS
6691								
6692								
6693	031362	013737	002320	002322	33\$:	MOV	PR.BIT,M1.BIT	;MOVE LAST DATA BIT FOR ECC GEN
6694	031370	004737	032260		JSR	PC,ECCGEN		;GENERATE ECC
6695	031374	005037	001214		CLR	\$TMP5		;CLEAR FOR PASS COUNTING
6696	031400	005737	002344		TST	ECCSRC		;TEST WHERE ECC WORDS ARE
6697	031404	001006			BNE	36\$;THEY ARE IN BUFFER - SKIP
6698	031406	013737	002340	052444	MOV	ECCHI,0BUFF+1000		;ELSE MOVE THEM INTO BUFFER
6699	031414	013737	002342	052446	MOV	ECCL0,0BUFF+1002		
6700								
6701	031422	012305		36\$:	MOV	(R3)+,R5		;GET ECC FROM BUFFER
6702	031424	012701	000001		MOV	#BIT0,R1		;SET BIT POINTER
6703								
6704	031430	013737	002320	002322	5\$:	MOV	PR.BIT,M1.BIT	;SHIFT PRESENT TO PREVIOUS
6705	031436	005037	002320		CLR	PR.BIT		;AND SET UP PRESENT
6706	031442	030105			BIT	R1,R5		
6707	031444	001403			BEQ	34\$		
6708	031446	012737	000001	002320	MOV	#1,PR.BIT		
6709	031454	004737	033414	34\$:	JSR	PC,RDBIT		;GO READ BIT
6710	031460	016237	000026	002204	MOV	RKMR1(R2),T.MR1		;GET MR1
6711	031466	023737	002204	002244	CMP	T.MR1,E.MR1		;CHECK IF CORRECT
6712	031474	001402			BEQ	38\$;YES - SKIP
6713	031476	000137	032110		JMP	11\$;ELSE JUMP TO REPORT EXIT
6714	031502	023737	002326	001226	38\$:	CMP	BITCNT,\$TMP12	;TEST IF FIRST ECC BIT
6715	031510	001402			BEQ	49\$;YES - SKIP ECC GEN (ALREADY DONE)
6716	031512	004737	032260		JSR	PC,ECCGEN		;GENERATE ECC FOR BIT
6717	031516	016237	000032	002214	49\$:	MOV	RKECPT(R2),T.ECPT	;GET PATTERN
6718	031524	023737	002214	002254	CMP	T.ECPT,E.ECPT		;TEST IF ECC PAT CORRECT
6719	031532	001402			BEQ	45\$;YES SKIP
6720	031534	000137	032074		JMP	10\$;ELSE GO REPORT ERROR
6721								
6722	031540	005237	002326	45\$:	INC	BITCNT		;BUMP BIT COUNT
6723	031544	023737	002326	001230	CMP	BITCNT,\$TMP13		;TEST IF 1ST BIT OF ECC
6724	031552	001003			BNE	46\$;NO - SKIP
6725	031554	042737	020000	002244	BIC	#ECCW,E.MR1		;ELSE CLR ECCW FOR ECC READ
6726	031562	023737	002326	001232	46\$:	CMP	BITCNT,\$TMP14	;TEST IF 1ST BIT OF POSTAMBLE
6727	031570	001027			BNE	47\$;NO - SKIP
6728	031572	052737	020000	002244	BIS	#ECCW,E.MR1		;ELSE SET ECCW FOR ECC DONE
6729	031600	042737	100000	002244	BIC	#RDGATE,E.MR1		;CLEAR READ GATE
6730								
6731	031606	016237	000032	002214	MOV	RKECPT(R2),T.ECPT		;STORE ACTUAL PATTERN
6732	031614	016237	000030	002212	MOV	RKECPS(R2),T.ECPS		;STORE ACTUAL POSITION

N10

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 130
 DZR6EA.P11 28-SEP-76 15:31 DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE

SEQ 0130

6733	031622	013737	002350	002252		MOV	ECPOX,E.ECPS	;GET EXPECTED POSITION FOR REPORT
6734	031630	023737	002214	002254		CMP	T.ECPT,E.ECPT	;CHECK IF CORRECT
6735	031636	001163				BNE	20\$;NO SKIP
6736	031640	023737	002212	002252		CMP	T.ECPS,E.ECPS	;CHECK IF POSITION CORRECT
6737	031646	001157				BNE	20\$;NO - SKIP
6738								
6739	031650	023737	002326	001232	47\$:	CMP	BITCNT,\$TMP14	;TEST IF ECC DONE
6740	031656	002436				BLT	48\$;NO - SKIP
6741	031660	005737	002346			TST	ECPATX	;TEST IF EXPECTED ECC IS 0
6742								; (NO ERROR)
6743	031664	001433				BEQ	48\$;YES - SKIP
6744	031666	023737	001232	002326		CMP	\$TMP14,BITCNT	;TEST IF FIRST POSTAMBLE BIT
6745	031674	001402				BEQ	52\$;YES - SKIP
6746	031676	005237	002350			INC	ECPOX	;ELSE BUMP POS COUNT EXPECTED
6747	031702	016237	000032	002214	52\$:	MOV	RKECPT(R2),T.ECPT	;GET PATTERN
6748	031710	016237	000030	002212		MOV	RKECPS(R2),T.ECPS	;GET POSITION
6749	031716	013737	002350	002252		MOV	ECPOX,E.ECPS	;GET EXPECTED POSITION
6750	031724	023737	002214	002254		CMP	T.ECPT,E.ECPT	;TEST IF PATTERN CORRECT
6751	031732	001402				BEQ	51\$;YES - SKIP
6752	031734	000137	032124			JMP	22\$;ELSE GO REPORT
6753	031740	023737	002212	002252	51\$:	CMP	T.ECPS,E.ECPS	;CHECK IF POSITION CORRECT
6754	031746	001402				BEQ	48\$;YES - SKIP
6755	031750	000137	032140			JMP	23\$;ELSE GO REPORT
6756								
6757	031754	005701			48\$:	TST	R1	;TEST IF WORD DONE
6758	031756	100403				BMI	6\$;YES SKIP
6759	031760	006301				ASL	R1	;ELSE SHIFT BIT POINTER
6760	031762	000137	031430			JMP	5\$;AND LOOP
6761								
6762	031766	005737	001214		6\$:	TST	\$TMP5	;TEST PASS COUNT. IF NOT 0
6763								; (NOT THE FIRST ECC WORD JUST WRITTE)
6764	031772	001004				BNE	7\$; - SKIP
6765								
6766	031774	005237	001214			INC	\$TMP5	;BUMP PASS COUNT
6767	032000	000137	031422			JMP	36\$	
6768								
6769	032004	023727	001214	000001	7\$:	CMP	\$TMP5,#1	;TEST PASS COUNT. IF NOT 1 (NOT THE
6770	032012	001007				BNE	8\$;2ND ECC WORD JUST READ) - SKIP
6771	032014	005005				CLR	R5	;CLEAR WORD (POSTAMBLE)
6772	032016	012701	000001			MOV	#BIT0,R1	;SET BIT POINTER
6773	032022	005237	001214			INC	\$TMP5	;BUMP PASS COUNT
6774	032026	000137	031430			JMP	5\$	
6775								
6776	032032	004437	034154		8\$:	JSR	R4,GETREG	;GET RK611 REGS
6777	032036	013737	002260	002220		MOV	L.CS1,E.CS1	;SET EXPECTED CS1
6778	032044	023737	002220	002160		CMP	E.CS1,T.CS1	;CHECK IF CORRECT
6779	032052	001060				BNE	21\$;NO - SKIP
6780	032054	005724				TST	(R4)+	;ELSE BUMP RETURN POINTER TO NO ERR
6781	032056	000446				BR	13\$	
6782								
6783	032060	013704	001216		9\$:	MOV	\$TMP6,R4	;SET RETURN TOR MR1 ERR.
6784	032064	012737	031232	001236		MOV	#43\$,\$ESCAPE	;SET RETURN TO CONTINUE TEST
6785	032072	000427				BR	12\$	
6786								
6787	032074	013704	001220		10\$:	MOV	\$TMP7,R4	;SET RETURN FOR ECC ERROR
6788	032100	012737	031266	001236		MOV	#44\$,\$ESCAPE	;SET RETURN TO CONTINUE TEST

B11

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZ6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 131
 DZ6EA.P11 28-SEP-76 15:31 DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE

SEG 0131

```

6789 032106 000421          BR      12$
6790
6791 032110 013704 001216 11$: MOV   STMP6,R4      ;SET RETURN FOR MR1 ERROR
6792 032114 012737 031540 001236  MOV   #45$, $ESCAPE ;SET RETURN TO CONTINUE TEST
6793 032122 000413          BR      12$
6794
6795 032124 013704 001222 22$: MOV   STMP10,R4     ;GET RETURN FOR PATTERN ERROR
6796 032130 012737 031740 001236  MOV   #51$, $ESCAPE ;SET RETURN TO CONTINUE TEST
6797 032136 000405          BR      12$
6798
6799 032140 013704 001224 23$: MOV   STMP11,R4     ;RET RETURN FOR POSITION ERROR
6800 032144 012737 031754 001236  MOV   #48$, $ESCAPE ;SET RETURN TO CONTINUE TEST
6801
6802
6803 032152 013746 001224 12$: MOV   STMP11, -(SP)  ;PREP STACK FOR RETURN
6804 032156 032777 001000 146754  BIT   #BIT9, $SWR    ;TEST IF LOOP ON ERROR
6805 032164 001407          BEQ   14$           ;NO SKIP
6806
6807 032166 005726          TST   (SP)+         ;REMOVE LAST ENTRY FROM STACK
6808 032170 005037 001236  CLR   $ESCAPE      ;NO LOOP - SLEAR ESCAPE
6809
6810 032174          13$:
6811 032174 012605          MOV   (SP)+, R5     ;; POP STACK INTO R5
6812 032176 012603          MOV   (SP)+, R3     ;; POP STACK INTO R3
6813 032200 012601          MOV   (SP)+, R1     ;; POP STACK INTO R1
6814 032202 012600          MOV   (SP)+, R0     ;; POP STACK INTO R0
6815 032204 000204          14$: RTS   R4
6816
6817 032206 013704 001212 20$: MOV   STMP4, R4     ;GET RETURN FOR ECC ERROR
6818 032212 000402          BR      15$
6819 032214 013704 001210 21$: MOV   STMP3, R4     ;GET CSI ERROR RETURN
6820 032220 005037 001236 15$: CLR   $ESCAPE      ;TEST IF LOOP ON ERROR
6821 032224 032777 001000 146706  BIT   #BIT9, $SWR    ;TEST IF LOOP ON ERROR
6822 032232 001360          BNE   13$           ;YES - SKIP
6823
6824 032234 013700 001254          MOV   $TESTN, R0    ;SET REGISTERS TO LOOP ON ERROR OR
6825 032240 006300          ASL   R0            ;ESCAPE TO NEXT TEST.
6826 032242 016037 035270 001236  MOV   $$W08TB, R0), $ESCAPE ;SET UP TO ESCAPE TO NEXT TEST
6827
6828 032250 162737 000002 001236  SUB   #2, $ESCAPE   ;ADJUST RETURN
6829 032256 000746          BR      13$         ;GO TO EXIT
6830
6831          .SBTTL  GENERATE ECC WORD
6832          ;*
6833          ;* EACH TIME THIS ROUTINE IS CALLED THE ECC IS CALCULATED FOR THE
6834          ;* NEXT BIT TO BE READ OR WRITTEN AND THE BITS PREVIOUSLY
6835          ;* READ OR WRITTEN (STORED IN ECCHI AND ECCLO). THE RESULTS OF
6836          ;* THE CALCULATION ARE PLACED IN ECCHI AND ECCLO. THE LOW ORDER
6837          ;* 11 BITS OF ECCHI ARE PLACED IN E.ECPT FOR EASY COMPARISON
6838          ;* TO RKECPT.
6839          ;*
6840          ;* CALL:   JSR      PC, ECCGEN
6841          ;* RETURN: RTS      R4
6842          ;*
6843          ;* ECCGEN: MOV    RO, -(SP)      ;STORE RO
6844          ;*          MOV    #PI, BIT, R0    ;GET ADDRESS OF NEXT BIT
6845          ;*          BIT   #BIT1, L, CSI   ;CHECK IF WRITING
6846          ;*          BNE   12$           ;YES - SKIP
  
```

```

6845 032276 012700 002322          MOV      #M1.BIT,RO      ;ELSE CHANGE TO PRESENT BIT
6846
6847 032302 005710          12$:    TST      (RO)      ;CHECK IF NEXT BIT 0
6848 032304 001410          BEQ      3$             ;YES - SKIP
6849 032306 032737 000001 002340      BIT      #1,ECCHI      ;NO, CHECK IF BIT 31 = 1
6850 032314 001410          BEQ      5$             ;NO, SET ECC XORED BIT
6851 032316 005037 002336          1$:    CLR      ECCXOR    ;CLEAR ECC XORED BIT
6852 032322 000241          CLC      ;CLEAR CARRY FLOP
6853 032324 000410          BR       7$           ;SHIFT IN ECC BIT
6854
6855 032326 032737 000001 002340      3$:    BIT      #1,ECCHI      ;CHECK IF BIT 31 = 1
6856 032334 001770          BEQ      1$           ;NO, CLEAR ECC XORED BIT
6857 032336 012737 000001 002336      5$:    MOV      #1,ECCXOR    ;SET ECC XOR
6858 032344 000261          SEC      ;SET CARRY FLOP
6859 032346 006037 002342          7$:    ROR      ECCLO      ;
6860 032352 006037 002340          ROR      ECCHI         ;
6861 032356 005737 002336          TST      ECCXOR        ;CHECK IF XOR NEEDED
6862 032362 001422          BEQ      10$          ;NO, RETURN
6863 032364 012746 020020          MOV      #BIT13:BIT4,-(SP) ;DO XOR OF ECC BITS
6864 032370 043716 002342          BIC      ECCLO,(SP)    ; 0, 2, 11, 21, 23
6865 032374 042737 020020 002342      BIC      #BIT13:BIT4,ECCLO
6866 032402 052637 002342          BIS      (SP)+,ECCLO
6867 032406 012746 002400          MOV      #BIT10:BIT8,-(SP)
6868 032412 043716 002340          BIC      ECCHI,(SP)
6869 032416 042737 002400 002340      BIC      #BIT10:BIT8,ECCHI
6870 032424 052637 002340          BIS      (SP)+,ECCHI
6871 032430 013737 002340 002254      10$:   MOV      ECCHI,E.ECPT ;STORE ECC PATTERN
6872 032436 042737 174000 002254      BIC      #174000,E.ECPT ;MASK UNSEEN BITS
6873 032444 012600          MOV      (SP)+,RO      ;RESTORE RO
6874 032446 000207          RTS      PC            ;RETURN
6875
6876          .SBTTL  SIMULATE ONE BIT OF WRITE DATA IN MAINTENANCE MODE
6877          :*    ONE BIT OF DATA IS WRITE SIMULATED. THE CONTENTS OF
6878          :*    RKMRI IS COMPUTED (PRECOMP ADVANCE AND DELAY, MAINTENANCE
6879          :*    ENCODED WRITE DATA) FOR THE PROPER TRANSITION OF THE
6880          :*    MAINTENANCE CLOCK AND CHECKED AT EACH TRANSITION. IF MARI
6881          :*    IS INCORRECT AT ANY TRANSITION, AN ERROR IS SET UP TO
6882          :*    CONTAIN A MESSAGE THAT IDENTIFIES THE TRANSITION WHEN THE
6883          :*    ERROR OCCURRED.
6884          :*
6885          :*    CALL:  JSR      PC,WRTBIT
6886          :*    RETURN: RTS      PC+2      FOR NO ERROR RETURN
6887          :*           RTS      PC        FOR ERROR IN MARI
6888 032450 052737 002400 002244      WRTBIT: BIS      #MCLK!MEWD,E.MRI ;CREATE EXPECTED MAINT. REG. 1
6889 032456 012762 000440 000026      MOV      #DMD!MCLK,RKMRI(R2) ;PROVIDE 1ST UPWARD TRANSITION
6890 032464 016237 000026 002204      MOV      RKMRI(R2),T.MRI ;STORE MAINT. REG. 1
6891 032472 023737 002244 002204      CMP      E.MRI,T.MRI ;CHECK IF MAINT REG 1 CORRECT
6892 032500 001416          BEQ      3$           ;YES, PROVIDE DOWNWARD TRANSITION
6893 032502 012737 032522 001236      MOV      #1$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
6894 032510 012737 045452 001512      MOV      #EMW1,EMW+2 ;LOAD ERROR MESSAGE
6895 032516 011646          MOV      (SP),-(SP) ;SAVE RETURN
6896 032520 000207          RTS      PC            ;MARI INCORRECT ON UPWARD TRANSITION
6897
6898 032522 032777 001000 146410      1$:    BIT      #SW9,$SWR    ;CHECK IF LOOP ON ERROR
6899 032530 001402          BEQ      3$           ;NO, CONTINUE
6900 032532 000137 033404          JMP      63$          ;YES, LOOP ON ERROR

```


F11

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 135
 DZR6EA.P11 28-SEP-76 15:31 SIMULATE ONE BIT OF WRITE DATA IN MAINTANANCE MODE

SEQ 0135

```

7013 033404 012706 001100      63$:  MOV    #STACK,SP      ;FORCE STACK
7014 033410 000177 145474      JMP    @SLPERR          ;LOOP ON ERROR
7015
7016      .SBTTL  SIMULATE ONE BIT OR READ DATA IN MAINTANANCE MODE
7017      :*
7018      :*      ONE BIT OF DATA IS READ SIMULATED. NO CHECKING IS
7019      :*      DONE BY THIS ROUTINE.
7020      :*
7021      :*      CALL:   JSR      PC,RDBIT
7022      :*      RETURN: RTS      PC
7023      RDBIT: TST     PR,BIT      ;CHECK IF ONE
7024      BNE    10$,      ;YES, SIMULATE ONE
7025      TST     M1,BIT     ;CHECK IF PREVIOUS ONE
7026      BEQ    4$,      ;NO, INSERT TRANSITION
7027 033430 012762 000440 000026  MOV    #DMD!MCLK,RKMR1(R2) ;YES, DO NOT INSERT TRANSITION
7028 033436 000403      BR     5$,      ;CLOCK IN ZERO
7029
7029 033440 012762 001440 000026  4$:   MOV    #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
7030 033446 012762 000040 000026  5$:   MOV    #DMD,RKMR1(R2) ;CLOCK IN ZERO
7031 033454 012762 000440 000026  MOV    #DMD!MCLK,RKMR1(R2)
7032 033462 012762 000040 000026  MOV    #DMD,RKMR1(R2)
7033 033470 000207      RTS     PC          ;RETURN
7034
7035 033472 012762 000440 000026  10$:  MOV    #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
7036 033500 012762 000040 000026  MOV    #DMD,RKMR1(R2)
7037 033506 012762 001440 000026  MOV    #DMD!MCLK!MERD,RKMR1(R2)
7038 033514 012762 000040 000026  MOV    #DMD,RKMR1(R2)
7039 033522 000207      RTS     PC          ;RETURN
7040
7041      .SBTTL  MEMORY CHECK ENABLE TRAP
7042      :*
7043      :*      IF THE PROCESSOR IN USE HAS MEMORY PARITY OPTION,
7044      :*      IT WILL BE ENABLED. THIS ROUTINE WILL PROCESS TRAPS CAUSED
7045      :*      BY MEMORY PARITY ERRORS.
7045 033524 012737 033540 001236  MEMERR: MOV    #10$, $ESCAPE ;LOAD ESCAPE
7046 033532 011637 002310      MOV    (SP),TRAPPC ;STORE PC
7047 033536 104001      ERROR  1 ;REPORT MEM PARITY ERROR
7048 033540 005037 001236 10$:  CLR    $ESCAPE ;CLEAR ESCAPE
7049 033544 032777 001000 145366  BIT    #SW9,@SWR ;CHECK IF LOOP ON ERROR
7050 033552 001001      BNE    15$,      ;YES, FORCE STACK AND TRY AGAIN
7051 033554 000002      RTI     ;NO, RETURN
7052
7053 033556 012706 001100      15$:  MOV    #STACK,SP      ;INITIALIZE STACK
7054 033562 000177 145322      JMP    @SLPERR          ;LOOP ON ERROR
7055
7056      .SBTTL  CLEAR INPUT BUFFER
7057 033566 012700 050440      CLRIBF: MOV    #IBUFF,R0 ;GET BUFFER POINTER
7058 033572 010146      MOV    R1,-(SP) ;STORE R1
7059 033574 012701 000400      MOV    #400,R1 ;SET COUNT FOR CLEAR
7060 033600 005020      1$:   CLR    (R0)+ ;CLEAR BUFFER WORD
7061 033602 005301      DEC    R1 ;DEC COUNT
7062 033604 001375      BNE    1$,      ;LOOP UNTIL COUNT IS ZERO
7063 033606 012601      MOV    (SP)+,R1 ;RESTORE R1
7064 033610 000204      RTS     R4 ;RETURN
7065
7066      .SBTTL  BUILD DATA BUFFER
7067      :*
7068      :*      THE PATTERN SPECIFIED IN THE CALL IS LOADED INTO THE DATA
7069      :*      BUFFER. THE ENTIRE BUFFER IS ALWAYS LOADED (400(8) WORDS).
706e 033612      B_LOADAT:

```


G11

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A MACY11 27(1006) 05-OCT-76 09:12 PAGE 136
 DZR6EA.P11 28-SEP-76 15:31 BUILD DATA BUFFER

SEG 0136

```

7069 033612 010046      MOV      R0,-(SP)      ;;PLSH R0 ON STACK
7070 033614 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7071 033616 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7072 033620 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7073 033622 012700 051444      MOV      #OBUFF,R0     ;LOAD R0 TO POINT TO BUFFER
7074 033626 012701 000400      MOV      #400,R1      ;SET DATA LENGTH
7075 033632 021427 000001      CMP      (R4),#1      ;TEST IF PATTERN 1(ALL ZERO)
7076 033636 001004      BNE      2$          ;NO - SKIP
7077 033640 005020      1$: CLR      (R0)+      ;ELSE CLEAR OBUFF
7078 033642 005301      DEC      R1
7079 033644 001375      BNE      1$
7080 033646 000461      BR       13$
7081 033650 021427 000007      2$: CMP      (R4),#7      ;TEST IF PAT 7 (ALL ONES)
7082 033654 001006      BNE      4$          ;NO SKIP
7083 033656 012703 177777      MOV      #177777,R3   ;ELSE SET BUFFER TO ALL ONES
7084 033662 010320      3$: MOV      R3,(R0)+
7085 033664 005301      DEC      R1
7086 033666 001375      BNE      3$
7087 033670 000450      BR       13$
7088 033672 021427 000006      4$: CMP      (R4),#6      ;TEST IF PAT 6 (COMPOSIT ROTATING)
7089 033676 001003      BNE      5$          ;NO - SKIP
7090 033700 012705 050400      MOV      #PAT6,R5     ;ELSE GET ADDRESS OF PATTERN 6
7091 033704 000427      BR       10$         ;GO LOAD BUFFER
7092 033706 021427 000005      5$: CMP      (R4),#5      ;TEST IF PATTERN 5 (ALT 1 & 0)
7093 033712 001006      BNE      7$          ;NO SKIP
7094 033714 012703 125252      MOV      #125252,R3   ;ELSE LOAD OBUFF WITH PAT 5
7095 033720 010320      6$: MOV      R3,(R0)+
7096 033722 005301      DEC      R1
7097 033724 001375      BNE      6$
7098 033726 000431      BR       13$
7099 033730 021427 000004      7$: CMP      (R4),#4      ;TEST IF PATTERN 4 (HI-LO FREQ MIX)
7100 033734 001003      BNE      8$          ;NO - SKIP
7101 033736 012705 050340      MOV      #PAT4,R5     ;LOAD POINTER WITH ADD OF PAT 4
7102 033742 000410      BR       10$
7103 033744 021427 000003      8$: CMP      (R4),#3      ;TEST IF PAT 3 (MAX PRECOMP PHASE MIX)
7104 033750 001003      BNE      9$          ;NO - SKIP
7105 033752 012705 050300      MOV      #PAT3,R5     ;LOAD POINTER WITH ADD OF PAT 3
7106 033756 000402      BR       10$         ;GO LOAD BUFFER
7107 033760 012705 050240      9$: MOV      #PAT2,R5     ;GET ADDRESS OF PAT 2
7108 033764 012703 000020      10$: MOV     #16,R3      ;SET PATTERN LENGTH COUNT
7109 033770 012520      11$: MOV     (R5)+,(R0)+ ;MOV PATTERN INTO OBUFF
7110 033772 005301      DEC      R1          ;DEC COUNTERS
7111 033774 005303      DEC      R3
7112 033776 001374      BNE      11$         ;LOOP UNTIL PATTERN IS MOVED
7113 034000 012705 051444      MOV      #OBUFF,R5   ;SET R5 TO START OF BUFFER
7114 034004 012520      12$: MOV     (R5)+,(R0)+ ;REPEAT PATTERN THROUGH BUFFER
7115 034006 005301      DEC      R1
7116 034010 001375      BNE      12$
7117 034012 005724      13$: TST      (R4)+      ;BUMP PAST PARAMETER
7118 034014 012605      MOV      (SP)+,R5     ;POP STACK INTO R5
7119 034016 012603      MOV      (SP)+,R3     ;POP STACK INTO R3
7120 034020 012601      MOV      (SP)+,R1     ;POP STACK INTO R1
7121 034022 012600      MOV      (SP)+,R0     ;POP STACK INTO R0
7122 034024 000204      RTS      R4
7123
7124 .SBTTL  LOAD "L" REGISTERS

```


H11

```

7125      ;*      THE "L" REGISTERS ARE LOADED FORM THE PARAMETERS FOLLOWING
7126      ;*      THE SUBROUTINE CALL. DIAGNOSTIC MODE IS SET IN L.MR1
7127      ;*      AND L.CS1 IS LOADED WITH THE COMMAND.
7128      ;*
7129      ;*      CALL:
7130      ;*      JSR      R4,LOADRK
7131      ;*      .WORD      CYLINDER
7132      ;*      .BYTE      ;SECTOR
7133      ;*      .BYTE      ;TRACK
7134      ;*      .WORD      ;BUFFER ADDRESS
7135      ;*      .WORD      ;WORD COUNT
7136      ;*      .WORD      ;COMMAND,FORMAT,DRIVE TYPE,&UPPER BUS ADDRESS BITS
7137      ;*
7138      ;*      RETURN:
7139      ;*      RTS      R4
7140
7141      034026 012437 002274      LOADRK: MOV      (R4)+,L.DCYL      ;LOAD CYLINDER
7142      034032 012437 002256      MOV      (R4)+,L.DA      ;TRACK AND SECTOR
7143      034036 012437 002264      MOV      (R4)+,L.BA      ;BUS ADDRESS
7144      034042 012437 002262      MOV      (R4)+,L.WC      ;WORD COUNT
7145      034046 012737 000040      MOV      #DMD,L.MR1      ;SET DIAGNOSTIC MODE
7146      034054 005037 002272      CLR      L.ASOF          ;CLEAR OFFSET
7147      034060 005037 002270      CLR      L.CS2          ;CLEAR CS2
7148      034064 012437 002260      MOV      (R4)+,L.CS1     ;LOAD CS1
7149      034070 000204      RTS      R4              ;RETURN
7150
7151      .SBTTL  START THE OPERATION
7152      ;*      THE INFORMATION IN THE "L" REGISTERS ARE LOADED INTO THE
7153      ;*      RK611 REGISTERS IN A STRAIGHT TRANSFER.
7154
7155      034072 013762 002262 000002  OPSTRT: MOV      L.WC,RKWC(R2) ;MOVE THE "L"REGISTERS INTO
7156      034100 013762 002264 000004  MOV      L.BA,RKBA(R2) ;CORRESPONDING RK611 REGISTERS.
7157      034106 013762 002274 000020  MOV      L.DCYL,RKDCYL(R2)
7158      034114 013762 002266 000006  MOV      L.DA,RKDA(R2)
7159      034122 013762 002270 000010  MOV      L.CS2,RKCS2(R2)
7160      034130 013762 002276 000026  MOV      L.MR1,RKMR1(R2)
7161      034136 013762 002272 000016  MOV      L.ASOF,RKASOF(R2)
7162      034144 013762 002260 000000  MOV      L.CS1,RKCS1(R2)
7163      034152 000204      RTS      R4
7164
7165      .SBTTL  GET THE RK611 REGISTERS
7166      ;*      ALL THE RK611 REGISTERS EXCEPT THE DATA BUFFER ARE
7167      ;*      STORED IN THE "T" REGISTERS.
7168
7169      034154      GETREG:
7170      034154 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
7171      034156 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7172      034160 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7173      034162 010200      MOV      R2,R0          ;GET RK BASE
7174      034164 012701 002160      MOV      #T.CS1,R1     ;GET START OF REGS
7175      034170 012703 000011      MOV      #11,R3        ;SET COUNT
7176      034174 012021      1$:      MOV      (R0)+,(R1)+   ;GET CS1 THRU DCYL
7177      034176 005303      DEC      R3
7178      034200 001375      BNE     1$            ;LOOP UNTIL DONE
7179
7180      034202 062700 000004      ADD     #4,R0          ;SKIP OVER DB AND SPARE
  
```

```

7181 034206 005021          CLR      (R1)+      ;CLEAR T.DB
7182 034210 012021          MOV      (R0)+,(R1)+ ;STORE MR1
7183 034212 012037 002212  MOV      (R0)+,T.ECPS ;      ECC POSITION
7184 034216 012037 002214  MOV      (R0)+,T.ECPT ;      ECC PATTERN
7185 034222 012037 002206  MOV      (R0)+,T.MR2  ;      MR2
7186 03-226 012037 002210  MOV      (R0)+,T.MR3  ;      MR3
7187 034232 012603          MOV      (SP)+,R3    ;:POP STACK INTO R3
7188 034234 012601          MOV      (SP)+,R1    ;:POP STACK INTO R1
7189 034236 012600          MOV      (SP)+,R0    ;:POP STACK INTO R0
7190 034240 000204          RTS      R4

7191
7192          .SBTTL  "FIRST SHALL BE LAST, LAST SHALL BE FIRST" SUBROUTINE
7193          :*      THE CONTENTS OF R3 IS SWAPPED END FOR END, I.E. BIT 15
7194          :*      BECOMES BIT 0 AND VICE VERSA,BIT 14 BECOMES BIT
7195          :*      1 AND VICE VERSA, ETC.
7196          :*
7197          :*      CALL:
7198          :*      JSR      R4,FSBLVV
7199          :*      WITH R3 LOADED WITH THE WORD TO BE SWAPPED
7200          :*
7201          :*      RETURN:
7202          :*      RTS      R4
7203          :*      WITH R3 SWAPPED.
7204          :*
7205          FSBLVV:
7206 034242 010046          MOV      R0,-(SP)    ;:PUSH R0 ON STACK
7207 034244 010146          MOV      R1,-(SP)    ;:PUSH R1 ON STACK
7208 034246 010446          MOV      R4,-(SP)    ;:PUSH R4 ON STACK
7209 034250 005004          CLR      R4          ;:CLEAR R4 FOR SWAPPED WORD
7210 034252 012700 000001  MOV      #BIT0,R0    ;:SET FOR BIT TEST
7211 034256 012701 100000  MOV      #BIT15,R1   ;:SET FOR BIT SET
7212 034262 030003          1$: BIT      R0,R3   ;:TEST IF BIT SET
7213 034264 001401          BEQ      2$         ;:NO - SKIP
7214 034266 050104          BIS      R1,R4     ;:SET CORRESPONDING BIT
7215 034270 006300          2$: ASL      R0     ;:SHIFT FOR NEXT BIT TEST
7216 034272 001403          BEQ      3$         ;:LAST BIT TESTED - YES - EXIT
7217 034274 000241          CLC          ;:CLEAR CARRY
7218 034276 006001          ROR      R1        ;:SHIFT FOR NEXT BIT SET
7219 034300 000770          BR       1$         ;:LOOP
7220 034302 010403          3$: MOV      R4,R3    ;:STORED SWAPPED WORD
7221 034304 012604          MOV      (SP)+,R4   ;:POP STACK INTO R4
7222 034306 012601          MOV      (SP)+,R1   ;:POP STACK INTO R1
7223 034310 012600          MOV      (SP)+,R0   ;:POP STACK INTO R0
7224 034312 000204          RTS      R4

7225
7226
7227          .SBTTL  CHECK FOR MEMORY CHECK ENABLE
7228
7229 034314 012737 034400 000004  PARCHK: MOV      #20$,ERRVEC ;SET VECTOR FOR MEMORY PARITY CHECK
7230 034322 012737 000340 000006  MOV      #PR7,ERRVEC+2
7231 034330 012737 000000 002334  MOV      #0,BADPAR   ;LOAD GOOD PARITY
7232 034336 005037 002332  CLR      MEMPAR     ;CLEAR FLAG
7233 034342 012703 172100  MOV      #MEMBAS,R3  ;LOAD REGISTER TO DETERMINE IF
7234          ;      MEMORY CHECK ENABLE AVAILIBLE
7235 034346 012704 000001  MOV      #1,R4       ;INITIALIZE MASK
7236 034352 012713 000001  16$: MOV      #PAR.EN,(R3) ;ENABLE MEMORY CHECK
    
```

J11

```

7237 034356 005713          TST      (R3)
7238 034360 050437 002332  BIS      R4,MEMPAR      ;SET FLAG
7239 034364 062703 000002  ADD      #2,R3
7240 034370 000241          CLC
7241 034372 006104          ROL      R4              ;CHECK IF FINISHED
7242 034374 001366          BNE     16$              ;NO, SET UP NEXT MEMORY PARITY MODULE
7243 034376 000406          BR       22$              ;RESTORE TRAP VECTOR
7244
7245 034400 022626          20$:    CMP      (SP)+,(SP)+  ;ADJUST STACK
7246 034402 062703 000002  ADD      #2,R3
7247 034406 000241          CLC
7248 034410 006104          ROL      R4              ;CHECK IF FINISHED
7249 034412 001357          BNE     16$              ;NO, GET NEXT LOCATION
7250 034414 012737 000006 000004 22$:    MOV      #ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
7251 034422 005037 000006  CLR      ERRVEC+2
7252 034426 005737 002332  TST      MEMPAR          ;CHECK IF MEMORY CHECK ENABLE AVAILIABLE
7253 034432 001005          BNE     25$              ;YES, RETURN
7254 034434 012737 000116 000114  MOV      #MEMVEC+2,MEMVEC ;RESTORE TRAP CATCHER
7255 034442 005037 000116  CLR      MEMVEC+2
7256 034446 000207          25$:    RTS      PC          ;RETURN
7257 .SBTTL ROUTINE TO SIZE MEMORY
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272
7273 034450 010046          $SIZE:  MOV      R0,-(SP)      ;;SAVE R0 ON THE STACK
7274 034452 010146          MOV      R1,-(SP)      ;;SAVE R1 ON THE STACK
7275 034454 010246          MOV      R2,-(SP)      ;;SAVE R2 ON THE STACK
7276 034456 010346          MOV      R3,-(SP)      ;;SAVE R3 ON THE STACK
7277 034460 013746 000004  MOV      @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
7278 034464 013746 000006  MOV      @#ERRVEC+2,-(SP)
7279 034470 010600          MOV      SP,R0          ;;SAVE THE STACK POINTER
7280 ;;SET THE ERRVEC PS TO THE PRESENT PS
7281 034472 104400          TRAP
7282 034474 012637 000006  MOV      (SP)+,@#ERRVEC+2 ;;PUSH OLD PSW AND PC ON STACK
7283 034500 012701 003776  MOV      #3776,R1      ;;SAVE THE PSW IN @#ERRVEC+2
7284 034504 105727          MOV      #3776,R1      ;;SETUP ADDRESS
7285 034506 000200          TSTB   (PC)+          ;;USE MEMORY MANAGEMENT?
7286 034510 100062          .WORD  200           ;;SET TO USE MEMORY MANAGEMENT
7287 034512 012737 034650 000004  BPL     SCORE         ;;BR IF NO
7288 034520 005737 177572  MOV      #SKTNEX,@#ERRVEC ;;SET FOR TIMEOUT
7289 034524 052737 100000 034506  TST     @#SRO         ;;KT11 ARE YOU THERE?
7290 034532 005046          BIS     #100000,$KT11 ;;YES--SET KT11 KEY
7291 034534 012702 172340  CLR     -(SP)         ;;INITIALIZE FOR "PAR" LOADING
7292 034540 012703 000010  MOV     #KIPARO,R2    ;;ADDRESS OF FIRST "PAR"
                          MOV     #+D8,R3      ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
    
```

```

*****
*CALL:
*   JSR   PC,$SIZE
*   RETURN
*$LSTAD WILL CONTAIN:
*   WITH KT11 OPTION      -- LAST VIRTUAL ADDRESS OF THE LAST BANK
*   WITHOUT KT11 OPTION  -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
*$KT11 IS THE MEMORY MANAGEMENT KEY
*$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
*   MUST BE SETUP BEFORE THE CALL
*$BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
*   DETERMINED BY ROUTINE
    
```

K11

```

7293 034544 012762 077406 177740 1$: MOV #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
7294 034552 011622 MOV (SP),(R2)+ ;;LOAD "PAR"
7295 034554 062716 000200 ADD #200,(SP) ;;UPDATE FOR NEXT "PAR"
7296 034560 077307 SOB R3,1$ ;;LOOP UNTIL ALL EIGHT ARE LOADED
7297 034562 012742 177600 MOV #177600,-(R2) ;;SETUP KIPAR7 FOR I/O
7298 034566 005042 CLR -(R2) ;;SETUP KIPAR6 FOR TESTING
7299 034570 012737 034606 000004 MOV #2$,@#ERRVEC ;;CATCH TIMEOUT IF NO SR3
7300 034576 012737 000020 172516 MOV #20,@#SR3 ;;ENABLE 22 BIT MODE
7301 034604 000401 BR 3$ ;;THIS PDP-11 HAS A SR3 REGISTER
7302 034606 022626 2$: CMP (SP)+,(SP)+ ;;CLEAN OFF THE STACK--NO SR3
7303 034610 005237 177572 3$: INC @#SR0 ;;TURN ON MEMORY MANAGEMENT
7304 034614 012737 034640 000004 MOV #SKTOUT,@#ERRVEC ;;SET FOR TIME OUT
7305 034622 005737 143776 4$: TST @#143776 ;;TRAP ON NON-EX-MEM
7306 034626 062712 000040 ADD #40,(R2) ;;MAKE A 1K STEP
7307 034632 023712 172356 CMP @#KIPAR7,(R2) ;;LAST ONE?
7309 034636 101371 BHI 4$ ;;NO--TRY IT
7309 034640 011202 SKTOUT: MOV (R2),R2 ;;GET LAST BANK+1
7310 034642 005037 177572 CLR @#SR0 ;;TURN OFF MEMORY MANAGEMENT
7311 034646 000421 BR $$SIZEX
7312 034650 042737 100000 034506 SKTNEX: BIC #100000,SKT11 ;;KT11 NON-EXISTENT
7313 034656 012737 034706 000004 SCORE: MOV #SCROUT,@#ERRVEC ;;SET FOR TIMEOUT
7314 034664 005002 CLR R2 ;;SET UP BANK
7315 034666 062701 004000 1$: ADD #4000,R1 ;;INCREMENT BY 1K
7316 034672 062702 000040 ADD #40,R2 ;;1K STEP
7317 034676 005711 TST (R1) ;;TRAP ON TIME OUT
7318 034700 022701 177776 CMP #177776,R1 ;;LAST ONE
7319 034704 001370 BNE 1$ ;;NO--TRY AGAIN
7320 034706 162701 004000 SCROUT: SUB #4000,R1
7321 034712 162702 000040 $SIZEX: SUB #40,R2 ;;DROP BACK
7322 034716 010006 MOV R0,SP ;;RESTORE THE STACK
7323 034720 012637 000006 MOV (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
7324 034724 012637 000004 MOV (SP)+,@#ERRVEC
7325 034730 010137 034752 MOV R1,$LSTAD ;;LAST ADDRESS
7326 034734 010237 034754 MOV R2,$LSTBK ;;LAST BANK
7327 034740 012603 MOV (SP)+,R3 ;;RESTORE R3
7328 034742 012602 MOV (SP)+,R2 ;;RESTORE R2
7329 034744 012601 MOV (SP)+,R1 ;;RESTORE R1
7330 034746 012600 MOV (SP)+,R0 ;;RESTORE R0
7331 034750 000207 RTS PC
7332 034752 000000 $LSTAD: .WORD 0 ;;CONTAINS THE LAST ADDRESS
7333 034754 000000 $LSTBK: .WORD 0 ;;CONTAINS THE LAST BANK
7334 .SBTTL SCOPE HANDLER ROUTINE
7335
7336 ;;*****
7337 ;;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7338 ;;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7339 ;;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7340 ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7341 ;;*SW14=1 LOOP ON TEST
7342 ;;*SW11=1 INHIBIT ITERATIONS
7343 ;;*SW09=1 LOOP ON ERROR
7344 ;;*SW08=1 LOOP ON TEST IN SWR<7:0>
7345 ;;*CALL
7346 ;;* SCOPE ;;SCOPE=IOT
7347
7348 034756 $SCOPE:

```

```

7349 034756 104407          CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
7350 034760 032777 040000 144152 1$: BIT      #BIT14,2SWR  ;; LOOP ON PRESENT TEST?
7351 034766 001131          BNE      $OVER    ;; YES IF SW14=1
7352          ;*****START OF CODE FOR THE XOR TESTER*****
7353 034770 000416          $XTSTR: BR     6$
7354          ;; IF RUNNING ON THE "XOR" TESTER CHANGE
7355 034772 013746 000004          MOV      2#ERRVEC, -(SP) ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
7356 034776 012737 035016 000004          MOV      #55,2#ERRVEC  ;; SAVE THE CONTENTS OF THE ERROR VECTOR
7357 035004 005737 177060          TST     2#177060      ;; SET FOR TIMEOUT
7358 035010 012637 000004          MOV      (SP)+,2#ERRVEC ;; TIME OUT ON XOR?
7359 035014 000500          BR      $SVLAD      ;; RESTORE THE ERROR VECTOR
7360 035016 022626          5$: CMP    (SP)+,(SP)+  ;; GO TO THE NEXT TEST
7361 035020 012637 000004          MOV      (SP)+,2#ERRVEC ;; CLEAR THE STACK AFTER A TIME OUT
7362 035024 000440          BR      7$          ;; RESTORE THE ERROR VECTOR
7363 035026          6$:;*****END OF CODE FOR THE XOR TESTER*****
7364 035026 032777 000400 144104          BIT      #BIT08,2SWR  ;; LOOP ON SPEC. TEST?
7365 035034 001421          BEQ     2$          ;; BR IF NO
7366 035036 005046          CLR     -(SP)      ;; CLEAR A TEMP. LOCATION
7367 035040 117716 144074          MOVB   2SWR,(SP)    ;; PICKUP THE DESIRED TEST NUMBER
7368 035044 001414          BEQ     8$          ;; BRANCH IF BAD TEST NUMBER IN SWR
7369 035046 022716 000042          CMP    #42,(SP)    ;; CHECK THE NUMBER IN THE SWR
7370 035052 002411          BLT    8$          ;; BRANCH IF TEST NUMBER IS OUT OF RANGE
7371 035054 011637 001102          MOV    (SP),$STNM  ;; UPDATE THE TEST NUMBER
7372 035060 005316          DEC    (SP)        ;; BACKUP BY ONE
7373 035062 006316          ASL    (SP)        ;; SCALE THE TEST NUMBER AS AN INDEX
7374 035064 062716 035270          ADD    #$$SW08TBL,(SP) ;; FORM THE ADDRESS OF TEST POINTER
7375 035070 013637 001106          MOV    2(SP)+,$LPADR ;; SET LOOP ADDRESS TO DESIRED TEST
7376 035074 000466          BR     $OVER      ;; GO LOOP ON THE TEST
7377 035076 005726          8$: TST   (SP)+      ;; CLEAN THE BAD TEST NUMBER OFF OF THE STACK
7378 035100 105737 001103          2$: TSTB  $ERFLG    ;; HAS AN ERROR OCCURRED?
7379 035104 001421          BEQ    3$          ;; BR IF NO
7380 035106 123737 001115 001103          CMPB  $ERMAX,$ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
7381 035114 101015          BHI    3$          ;; BR IF NO
7382 035116 032777 001000 144014          BIT   #BIT09,2SWR  ;; LOOP ON ERROR?
7383 035124 001404          BEQ    4$          ;; BR IF NO
7384 035126 013737 001110 001106          7$: MOV   $LPERR,$LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
7385 035134 000446          BR     $OVER
7386 035136 105037 001103          4$: CLRB  $ERFLG    ;; ZERO THE ERROR FLAG
7387 035142 005037 001234          CLR   $TIMES      ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
7388 035146 000415          BR     1$          ;; ESCAPE TO THE NEXT TEST
7389 035150 032777 004000 143762          3$: BIT   #BIT11,2SWR ;; INHIBIT ITERATIONS?
7390 035156 001011          BNE   1$          ;; BR IF YES
7391 035160 005737 001256          TST   $PASS      ;; IF FIRST PASS OF PROGRAM
7392 035164 001406          BEQ   1$          ;; INHIBIT ITERATIONS
7393 035166 005237 001104          INC   $ICNT      ;; INCREMENT ITERATION COUNT
7394 035172 023737 001234 001104          CMP   $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
7395 035200 002024          BGE   $OVER      ;; BR IF MORE ITERATION REQUIRED
7396 035202 012737 000001 001104          1$: MOV   #1,$ICNT  ;; REINITIALIZE THE ITERATION COUNTER
7397 035210 013737 035266 001234          MOV   $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
7398 035216 105237 001102          $SVLAD: INCB  $STNM  ;; COUNT TEST NUMBERS
7399 035222 113737 001102 001254          MOVB  $STNM,$STNM ;; SET TEST NUMBER IN APT MAILBOX
7400 035230 011637 001106          MOV   (SP),$LPADR ;; SAVE SCOPE LOOP ADDRESS
7401 035234 011637 001110          MOV   (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
7402 035240 005037 001236          CLR   $ESCAPE    ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
7403 035244 112737 000001 001115          MOVB  #1,$ERMAX  ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7404 035252 013777 001102 143662          $OVER: MOV   $STNM,2$DISPLAY ;; DISPLAY TEST NUMBER

```

```

7405 035260 013716 001106      MOV      $LPADR, (SP)      ;; FLDGE RETURN ADDRESS
7406 035264 000002      RTI                      ;; FIXES PS
7407 035266 003720      $MXCNT: 2000.           ;; MAX. NUMBER OF ITERATIONS
7408 035270      $SWC8TBL:
7409 035270 003426      .WORD   TST1+2          ;; STARTING ADDRESS OF TEST 1
7410 035272 004040      .WORD   TST2+2          ;; STARTING ADDRESS OF TEST 2
7411 035274 004452      .WORD   TST3+2          ;; STARTING ADDRESS OF TEST 3
7412 035276 005064      .WORD   TST4+2          ;; STARTING ADDRESS OF TEST 4
7413 035300 005476      .WORD   TST5+2          ;; STARTING ADDRESS OF TEST 5
7414 035302 006110      .WORD   TST6+2          ;; STARTING ADDRESS OF TEST 6
7415 035304 006522      .WORD   TST7+2          ;; STARTING ADDRESS OF TEST 7
7416 035306 007144      .WORD   TST10+2         ;; STARTING ADDRESS OF TEST 10
7417 035310 007414      .WORD   TST11+2         ;; STARTING ADDRESS OF TEST 11
7418 035312 007656      .WORD   TST12+2         ;; STARTING ADDRESS OF TEST 12
7419 035314 010126      .WORD   TST13+2         ;; STARTING ADDRESS OF TEST 13
7420 035316 010370      .WORD   TST14+2         ;; STARTING ADDRESS OF TEST 14
7421 035320 010704      .WORD   TST15+2         ;; STARTING ADDRESS OF TEST 15
7422 035322 011176      .WORD   TST16+2         ;; STARTING ADDRESS OF TEST 16
7423 035324 011724      .WORD   TST17+2         ;; STARTING ADDRESS OF TEST 17
7424 035326 012452      .WORD   TST20+2         ;; STARTING ADDRESS OF TEST 20
7425 035330 013200      .WORD   TST21+2         ;; STARTING ADDRESS OF TEST 21
7426 035332 013726      .WORD   TST22+2         ;; STARTING ADDRESS OF TEST 22
7427 035334 014462      .WORD   TST23+2         ;; STARTING ADDRESS OF TEST 23
7428 035336 015216      .WORD   TST24+2         ;; STARTING ADDRESS OF TEST 24
7429 035340 015744      .WORD   TST25+2         ;; STARTING ADDRESS OF TEST 25
7430 035342 016260      .WORD   TST26+2         ;; STARTING ADDRESS OF TEST 26
7431 035344 016552      .WORD   TST27+2         ;; STARTING ADDRESS OF TEST 27
7432 035346 017300      .WORD   TST30+2         ;; STARTING ADDRESS OF TEST 30
7433 035350 020034      .WORD   TST31+2         ;; STARTING ADDRESS OF TEST 31
7434 035352 020350      .WORD   TST32+2         ;; STARTING ADDRESS OF TEST 32
7435 035354 021032      .WORD   TST33+2         ;; STARTING ADDRESS OF TEST 33
7436 035356 021346      .WORD   TST34+2         ;; STARTING ADDRESS OF TEST 34
7437 035360 022116      .WORD   TST35+2         ;; STARTING ADDRESS OF TEST 35
7438 035362 022426      .WORD   TST36+2         ;; STARTING ADDRESS OF TEST 36
7439 035364 022764      .WORD   TST37+2         ;; STARTING ADDRESS OF TEST 37
7440 035366 023314      .WORD   TST40+2         ;; STARTING ADDRESS OF TEST 40
7441 035370 023650      .WORD   TST41+2         ;; STARTING ADDRESS OF TEST 41
7442 035372 024224      .WORD   TST42+2         ;; STARTING ADDRESS OF TEST 42
7443                                     ;;*****
7444      .SBTTL LOOP ON INTERNAL ERROR
7445
7446 035374 032777 001000 143536 SCOP1$: BIT      #SW9, @SWR      ;; CHECK IF LOOP ON ERROR
7447 035402 001405      BEQ      $$              ;; NO RETURN
7448 035404 105737 001103      TSTB     $ERFLG          ;; CHECK IF ERROR OCCURED
7449 035410 001402      BEQ      $$              ;; NO, RETURN
7450 035412 013716 001110      MOV      $LPERR, (SP)    ;; GO BACK TO BEGINNING OF LOOP
7451 035416 000002      $S:      RTI              ;; RETURN
7452      .SBTTL APT COMMUNICATIONS ROUTINE
7453
7454                                     ;;*****
7455 035420 112737 000001 035664 $ATY1: MOVB     #1, $FFLG      ;; TO REPORT FATAL ERROR
7456 035426 112737 000001 035662 $ATY3: MOVB     #1, $MFLG      ;; TO TYPE A MESSAGE
7457 035434 000403      BR       $ATYC
7458 035436 112737 000001 035664 $ATY4: MOVB     #1, $FFLG      ;; TO ONLY REPORT FATAL ERROR
7459 035444      $ATYC:
7460 035444 010046      MOV      RD, -(SP)       ;; PUSH RD ON STACK

```

```

7461 035446 010146      MOV      R1, -(SP)      ;; PUSH R1 ON STACK
7462 035450 105737 035662    TSTB    $MFLG          ;; SHOULD TYPE A MESSAGE?
7463 035454 001450      BEQ     5$             ;; IF NOT: BR
7464 035456 122737 000001 001270    CMPB    #APTENV, $ENV   ;; OPERATING UNDER APT?
7465 035464 001031      SNE     3$             ;; IF NOT: BR
7466 035466 132737 000100 001271    BITB    #APTPOOL, $ENVM ;; SHOULD SPOOL MESSAGES?
7467 035474 001425      BEQ     3$             ;; IF NOT: BR
7468 035476 017600 000004      MOV     @4(SP), R0      ;; GET MESSAGE ADDR.
7469 035502 062766 000002 000004    ADD     #2, 4(SP)       ;; BUMP RETURN ADDR.
7470 035510 005737 001250      1$:     TST     $MSGTYPE  ;; SEE IF DONE W/ LAST XMISSION?
7471 035514 001375      BNE     1$             ;; IF NOT: WAIT
7472 035516 010037 001264      MOV     R0, $MSGAD      ;; PUT ADDR IN MAILBOX
7473 035522 105720      2$:     TSTB    (R0)+    ;; FIND END OF MESSAGE
7474 035524 001376      BNE     2$
7475 035526 163700 001264      SUB     $MSGAD, R0      ;; SUB START OF MESSAGE
7476 035532 006200      ASF     R0              ;; GET MESSAGE LNGTH IN WORDS
7477 035534 010037 001266      MOV     R0, $MSGGLT     ;; PUT LENGTH IN MAILBOX
7478 035540 012737 000004 001250    MOV     #4, $MSGTYPE    ;; TELL APT TO TAKE MSG.
7479 035546 000413      BR      5$
7480 035550 017637 000004 035574 3$:     MOV     @4(SP), 4$      ;; PUT MSG ADDR IN JSR LINKAGE
7481 035556 062766 000002 000004    ADD     #2, 4(SP)       ;; BUMP RETURN ADDRESS
7482 035564 013746 177776      MOV     177776, -(SP)   ;; PUSH 177776 ON STACK
7483 035570 004737 036444      JSR     PC, $TYPE      ;; CALL TYPE MACRO
7484 035574 000000      4$:     .WORD    0
7485 035576      5$:
7486 035576 105737 035664      10$:    TSTB    $FFLG          ;; SHOULD REPORT FATAL ERROR?
7487 035602 001416      BEQ     12$           ;; IF NOT: BR
7488 035604 005737 001270      TST     $ENV           ;; RUNNING UNDER APT?
7489 035610 001413      BEQ     12$           ;; IF NOT: BR
7490 035612 005737 001250      11$:    TST     $MSGTYPE      ;; FINISHED LAST MESSAGE?
7491 035616 001375      BNE     11$           ;; IF NOT: WAIT
7492 035620 017637 000004 001252    MOV     @4(SP), $FATAL  ;; GET ERROR #
7493 035626 062766 000002 000004    ADD     #2, 4(SP)       ;; BUMP RETURN ADDR.
7494 035634 005237 001250      INC     $MSGTYPE       ;; TELL APT TO TAKE ERROR
7495 035640 105037 035664      12$:    CLRB    $FFLG          ;; CLEAR FATAL FLAG
7496 035644 105037 035663      CLRB    $LFLG          ;; CLEAR LOG FLAG
7497 035650 105037 035662      CLRB    $MFLG          ;; CLEAR MESSAGE FLAG
7498 035654 012601      MOV     (SP)+, R1       ;; POP STACK INTO R1
7499 035656 012600      MOV     (SP)+, R0       ;; POP STACK INTO R0
7500 035660 000207      RTS     PC              ;; RETURN
7501 035662 000      $MFLG: .BYTE 0          ;; MESSG. FLAG
7502 035663 000      $LFLG: .BYTE 0          ;; LOG FLAG
7503 035664 000      $FFLG: .BYTE 0          ;; FATAL FLAG
7504 035666      .EVEN
7505 000200      APTSIZE=200
7506 000001      APTENV=001
7507 000100      APTPOOL=100
7508 000040      APTCSUP=040
7509      .SBTTL  ERROR HANDLER ROUTINE
7510
7511      ;*****
7512      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7513      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7514      ;*AND GO TO TYPERR ON ERROR
7515      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7516      ;*SW15=1      HALT ON ERROR

```


7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572

035666 104407
035666 105237 001103
035670 001775
035674 013777 001102 143236
035704 032777 002000 143226
035712 001402
035714 104401 001240
035720 005237 001112
035724 011637 001116
035730 162737 000002 001116
035736 117737 143154 001114
035744 032777 020000 143166
035752 001004
035754 004737 036066
035760 104401 001245
035764
035764 122737 000001 001270
035772 001007
035774 113737 001114 036006
036002 004737 035436
036006 000
036007 000
036010 000777
036012 005777 143122
036016 100002
036020 000000
036022 104407
036024 032777 001000 143106
036032 001402
036034 013716 001110
036040 005737 001236
036044 001402
036046 013716 001236
036052
036052 022737 025042 000042
036060 001001
036062 000000
036064
036064 000002

```

;*SW13=1      INHIBIT ERROR TYPEOUTS
;*SW10=1      BELL ON ERROR
;*SW09=1      LOOP ON ERROR
;*CALL
;*           ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER

$ERROR:
00$:  CKSWR                ;;TEST FOR CHANGE IN SOFT-SWR
75:  INCB  SERFLG          ;;SET THE ERROR FLAG
    BEQ   75              ;;DON'T LET THE FLAG GO TO ZERO
    MOV  $STNM,$DISP     ;;DISPLAY TEST NUMBER AND ERROR FLAG
    BIT  #BIT10,$SWR     ;;BELL ON ERROR?
    BEQ  15              ;;NO - SKIP
    TYPE $BELL           ;;RING BELL
15:  INC  $ERTL           ;;COUNT THE NUMBER OF ERRORS
    MOV  (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
    SUB  #2,$ERRPC
    MOVB $ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
    BIT  #BIT13,$SWR     ;;SKIP TYPEOUT IF SET
    BNE  20$            ;;SKIP TYPEOUTS
    JSR  PC,TYPERR       ;;GO TO USER ERROR ROUTINE
    TYPE $SCRLF

20$:  CMPB #APTENV,$ENV   ;;RUNNING IN APT MODE
    BNE  25$            ;;NO SKIP APT ERROR REPORT
    MOVB $ITEMB,21$     ;;SET ITEM NUMBER AS ERROR NUMBER
    JSR  PC,$ATY4       ;;REPORT FATAL ERROR TO APT

21$:  .BYTE 0
    .BYTE 0
22$:  BR   22$          ;;APT ERROR LOOP
25:  TST  $SWR          ;;HALT ON ERROR
    BPL  35$          ;;SKIP IF CONTINUE
    HALT                ;;HALT ON ERROR!
35:  CKSWR                ;;TEST FOR CHANGE IN SOFT-SWR
    BIT  #BIT09,$SWR    ;;LOOP ON ERROR SWITCH SET?
    BEQ  45$          ;;BR IF NO
    MOV  $LPERR,(SP)    ;;FUDGE RETURN FOR LOOPING
    TST  $ESCAPE        ;;CHECK FOR AN ESCAPE ADDRESS
    BEQ  55$          ;;BR IF NONE
    MOV  $ESCAPE,(SP)  ;;FUDGE RETURN ADDRESS FOR ESCAPE
55:  CMP  #SENDAD,$#42  ;;ACT-11 AUTO-ACCEPT?
    BNE  65$          ;;BRANCH IF NO
    HALT                ;;YES
65:  RTI                ;;RETURN

```

```

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

```



```

*****
↑YPERR: SAVREG
7573 036066 104413
7574 036070 113737 001102 001254 MOV B $STNM,$STNM ;SAVE TEST NUMBER FOR REPORT
7575 036076 042737 177400 001254 BIC #177400,$STNM ;CLEAR UNUSED BITS
7576 036104 113700 001114 MOV B $ITEMB,R0 ;ENTER ERROR NUMBER
7577 036110 042700 177400 BIC #177400,R0 ;CLEAR UNUSED BITS
7578 036114 005300 DEC R0 ;FORM INDEX FOR ERROR TABLE
7579 036116 006300 ASL R0
7580 036120 006300 ASL R0
7581 036122 006300 ASL R0
7582 036124 062700 001330 15: ADD #ERRTB,R0 ;FORM ADDRESS OF ERROR ENTRY
7583 036130 012037 036144 MOV (R0)+,2$ ;GET EM POINTER
7584 036134 001404 BEQ 3$ ;BRANCH IF THERE ISN'T ONE
7585 036136 104401 001245 TYPE ,SCLF ;TYPE CARRIAGE RETURN LINE FEED
7586 036142 104401 TYPE ;TYPE ERROR MESSAGE (EM)
7587 036144 000000 25: .WORD 0 ;EM POINTER GOES HERE
7588 036146 012037 036162 35: MOV (R0)+,4$ ;GET DH POINTER
7589 036152 001404 BEQ 5$ ;BRANCH IF THERE ISN'T ONE
7590 036154 104401 001245 TYPE ,SCLF ;TYPE CR-LF
7591 036160 104401 TYPE ;TYPE DATA HEADER
7592 036162 000000 45: .WORD 0 ;DH POINTER GOES HERE
7593 036164 012001 55: MOV (R0)+,R1 ;GET DT POINTER
7594 036166 001445 BEQ 20$ ;BRANCH IF THERE ARE NONE
7595 036170 005004 CLR R4 ;RESET INDENT SWITCH
7596 036172 012000 MOV (R0)+,R0 ;GET DF POINTER
7597 036174 012002 MOV (R0)+,R2 ;STORE NUMBER OF DH'S
7598 036176 104401 001245 TYPE ,SCLF
7599 036202 112003 105: MOV B (R0)+,R3 ;GET & STORE NUMBER OF DATA WORDS
7600 036204 105720 TSTB (R0)+ ;BUMP PAST FORMAT WORD
7601 036206 005703 TST R3 ;TEST IF ANY DATA FOR THIS HEADER
7602 036210 001416 BEQ 14$ ;NO - SKIP DATA PRINT
7603 036212 005704 TST R4 ;CHECK IF INDENT WORDS
7604 036214 001004 BNE 12$ ;YES, GO INDENT
7605 036216 013146 115: MOV 2(R1)+,-(SP) ;PUT FIRST DATA WORD ON STACK
7606 036220 104402 TYPOC ;TYPE IT
7607 036222 005303 DEC R3 ;MORE DATA WORDS
7608 036224 001403 BEQ 13$ ;NO-BRANCH
7609 036226 104401 042724 125: TYPE ,SPACE2 ;TYPE SEPARATORS
7610 036232 000771 BR 11$ ;LOOP
7611 036234 104401 001245 135: TYPE ,SCLF ;TYPE <CR><LF>
7612 036240 005710 TST (R0) ;CHECK IF NEXT HEADER AVAILABLE
7613 036242 001401 BEQ 14$ ;NO, DO NOT CHANGE INDENT
7614 036244 005104 COM R4 ;CHANGE INDENT
7615 036246 005302 145: DEC R2 ;MORE DH'S?
7616 036250 003414 BLE 20$ ;NO-BRANCH
7617 036252 012037 036272 155: MOV (R0)+,18$ ;GET NEXT DH POINTER
7618 036256 001751 BEQ 10$ ;IF NO HEADER GET DATA
7619 036260 005704 TST R4 ;INDENT?
7620 036262 001402 BEQ 17$ ;NO-BRANCH
7621 036264 104401 042724 TYPE ,SPACE2 ;INDENT
7622 036270 104401 175: TYPE ;TYPE DH
7623 036272 000000 185: .WORD 0 ;DH POINTER GOES HERE
7624 036274 104401 001245 TYPE ,SCLF
7625 036300 000740 BR 10$ ;LOOP
7626 036302 104414 205: RESREG
7627 036304 005237 002314 INC ERRCNT ;INCREMENT ERROR COUNT

```

DZR6EA.F11 28-SEP-76 15:31

TYPE ERROR ROUTINE

```

7629 036310 032777 010000 142622 BIT      #SW12,2SWR      ;CHECK IF ABORT AFTER 20 ERRORS
7630 036316 001421 BEQ      25$      ;NO RETURN
7631 036320 022737 000024 002314 CMP      #20,ERRCNT ;CHECK IF EROR THRESHOLD EXCEEDED
7632 036326 103015 BHIS     25$      ;NO RETURN
7633 036330 104401 042727 TYPE     ,ABORT    ;TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
7634                                ;ERROR THRESHOLD EXCEEDED"
7635 036334 005737 000042 TST      42      ;CHECK IF IN CHAIN MODE
7636 036340 001407 BEQ      30$      ;NO HALT
7637 036342 012737 000001 024700 MOV      #1,$EOPCT ;FORCE END OF PASS COUNT TO ONE FOR ABCRT
7638 036350 012706 001100 MOV      #STACK,SP ;INITIALIZE STACK
7639 036354 000137 024652 JMP      $EOP     ;BRING IN NEXT PROGRAM IN CHAIN
7640 036360 000000 30$:     HALT
7641 036362 000207 25$:     RTS      PC

```

.SBTTL CONTROLLED PROGRAM HALT

```

7645 036364 013702 001324 CTRHLT: MOV      $BASE,R2      ;SET '611 BASE
7646 036370 012762 005000 000010 MOV      #CLR,AKCS2(R2) ;CLEAR SUBSYSTEM
7647 036376 005037 001236 CLR      $ESCAPE ;CLEAR ESCAPE
7648 036402 105037 001103 CLRFB   $ERFLG ;CLEAR ERROR FLAG
7649 036406 012706 001100 MOV      #STACK,SP ;CLEAR STACK
7650 036412 104401 042661 TYPE     ,OPROOF ;TYPE HALT MESSAGE
7651 036416 005737 000042 TST      42      ;TEST IF MONITOR PRESENT
7652 036422 001405 BEQ      5$      ;NO SKIP
7653 036424 012737 000001 024700 MOV      #1,$EOPCT ;FORCE END OF PROGRAM
7654 036432 000137 024652 JMP      $EOP     ;JUMP TO END OF PASS
7655
7656 036436 000000 5$:      HALT ;HALT PROGRAM
7657 036440 000137 002404 JMP      START1  ;RESTART IF CONTINUE

```

.SBTTL TYPE ROUTINE

```

7661 ;*****
7662 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7663 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7664 ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7665 ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7666 ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7667 ;*
7668 ;*CALL:
7669 ;*1) USING A TRAP INSTRUCTION
7670 ;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7671 ;*OR
7672 ;* TYPE
7673 ;* MESADR
7674 ;*
7675 ;*

```

```

7676 036444 105737 001157 $TYPE: TSTB   $TPFLG ;: IS THERE A TERMINAL?
7677 036450 100002 BPL      1$      ;: BR IF YES
7678 036452 000000 HALT     ;: HALT HERE IF NO TERMINAL
7679 036454 000430 BR      3$      ;: LEAVE
7680 036456 010046 1$:     MOV      RO,-(SP) ;: SAVE RO
7681 036460 017600 000002 MOV      @2(SP),RO ;: GET ADDRESS OF ASCIZ STRING
7682 036464 122737 000001 001270 CMPB   #APTENV,$ENV ;: RUNNING IN APT MODE
7683 036472 001011 BNE     62$     ;: NO GO CHECK FOR APT CONSOLE
7684 036474 132737 000100 001271 BITB   #APTPOOL,$ENVM ;: SPOOL MESSAGE TO APT

```

TYPE ROUTINE

```

7685 036502 001405          BEQ      62$          ;;NO GO CHECK FOR CONSOLE
7686 036504 010037 036514   MOV      RO,61$      ;;SETUP MESSAGE ADDRESS FOR APT
7687 036510 004737 035426   JSR     PC,$ATY3     ;;SPOOL MESSAGE TO APT
7688 036514 000000          .WORD    0           ;;MESSAGE ADDRESS
7689 036516 132737 000040 001271 61$:   BITB    #APTC SUP,$ENVM ;;APT CONSOLE SUPPRESSED
7690 036524 001003          BNE     60$          ;;YES, SKIP TYPE OUT
7691 036526 112046          MOVVB   (RO)+,-(SP)   ;;PUSH CHARACTER TO BE TYPED ONTO STACK
7692 036530 001005          BNE     4$           ;;BR IF IT ISN'T THE TERMINATOR
7693 036532 005726          TST    (SP)+         ;;IF TERMINATOR POP IT OFF THE STACK
7694 036534 012600          60$:   MOV      (SP)+,RO  ;;RESTORE RO
7695 036536 062716 000002       3$:   ADD      #2,(SP)    ;;ADJUST RETURN PC
7696 036542 000002          RTI                    ;;RETURN
7697 036544 122716 000011       4$:   CMPB    #HT,(SP)    ;;BRANCH IF <HT>
7698 036550 001430          BEQ     8$           ;;BRANCH IF NOT <CR>
7699 036552 122716 000200       5$:   CMPB    #CRLF,(SP)  ;;BRANCH IF NOT <CRLF>
7700 036556 001006          BNE     5$           ;;POP <CR><LF> EQUIV
7701 036560 005726          TST    (SP)+         ;;TYPE A CR AND LF
7702 036562 104401          TYPE                    ;;
7703 036564 001245          $CRLF                    ;;
7704 036566 105037 036722       CLRB   $CHARCNT      ;;CLEAR CHARACTER COUNT
7705 036572 000755          BR     2$           ;;GET NEXT CHARACTER
7706 036574 004737 036656       5$:   JSR     PC,$TYPEC    ;;GO TYPE THIS CHARACTER
7707 036600 123726 001156       6$:   CMPB    $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
7708 036604 001350          BNE     2$           ;;IF NO GO GET NEXT CHAR.
7709 036606 013746 001154       MOV     $NULL,-(SP)  ;;GET # OF FILLER CHARS. NEEDED
7710                                AND THE NULL CHAR.
7711 036612 105366 000001       7$:   DECB   1(SP)        ;;DOES A NULL NEED TO BE TYPED?
7712 036616 002770          BLT    6$           ;;BR IF NO--GO POP THE NULL OFF OF STACK
7713 036620 004737 036656       JSR     PC,$TYPEC    ;;GO TYPE A NULL
7714 036624 105337 036722       DECB   $CHARCNT     ;;DO NOT COUNT AS A COUNT
7715 036630 000770          BR     7$           ;;LOOP
7716
7717                                :HORIZONTAL TAB PROCESSOR
7718
7719 036632 112716 000040       8$:   MOVVB   #' ,(SP)    ;;REPLACE TAB WITH SPACE
7720 036636 004737 036656       9$:   JSR     PC,$TYPEC    ;;TYPE A SPACE
7721 036642 132737 000007 036722  BITB    #',$CHARCNT  ;;BRANCH IF NOT AT
7722 036650 001372          BNE     9$           ;;TAB STOP
7723 036652 005726          TST    (SP)+         ;;POP SPACE OFF STACK
7724 036654 000724          BR     2$           ;;GET NEXT CHARACTER
7725 036656 105777 142266       $TYPEC: TSTB   2$TPS      ;;WAIT UNTIL PRINTER IS READY
7726 036662 100375          BPL    $TYPEC        ;;
7727 036664 116677 000002 142260  MOVVB   2(SP),2$TPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7728 036672 122766 000015 000002  CMPB    #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
7729 036700 001003          BNE     1$           ;;BRANCH IF NO
7730 036702 105037 036722       CLRB   $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
7731 036706 000406          BR     $TYPEX        ;;EXIT
7732 036710 122766 000012 000002  1$:   CMPB    #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
7733 036716 001402          BEQ     $TYPEX        ;;BRANCH IF YES
7734 036720 105227          INCB   (PC)+        ;;COUNT THE CHARACTER
7735 036722 000000       $CHARCNT: .WORD    0 ;;CHARACTER COUNT STORAGE
7736 036724 000207       $TYPEX: RTS      PC
7737
7738                                .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
7739
7740                                ;:*****

```

F12

```

7741 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
7742 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
7743 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7744 ;*CALL:
7745 ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7746 ;*      TYPOS   N              ;;CALL FOR TYPEOUT
7747 ;*      .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
7748 ;*      .BYTE  M              ;;M=1 OR 0
7749 ;*                               ;;1=TYPE LEADING ZEROS
7750 ;*                               ;;0=SUPPRESS LEADING ZEROS
7751 ;*$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7752 ;*$TYPOS OR $TYPOC
7753 ;*CALL:
7754 ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7755 ;*      STYPON  N              ;;CALL FOR TYPEOUT
7756 ;*
7757 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7758 ;*CALL:
7759 ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7760 ;*      TYPOC  N              ;;CALL FOR TYPEOUT
7761
7762
7763 036726 017646 000000 $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
7764 036732 116637 000001 037151 MOVVB   1(SP),SOFILL  ;;LOAD ZERO FILL SWITCH
7765 036740 112637 037153 MOVVB   (SP)+,SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
7766 036744 062716 000002 ADD     #2,(SP)      ;;ADJUST RETURN ADDRESS
7767 036750 000406 BR      $TYPON
7768 036752 112737 000001 037151 $TYPOC: MOVVB  #1,SOFILL      ;;SET THE ZERO FILL SWITCH
7769 036760 112737 000006 037153 MOVVB   #6,SOMODE+1  ;;SET FOR SIX(6) DIGITS
7770 036766 112737 000005 037150 $TYPON: MOVVB  #5,SOCNT      ;;SET THE ITERATION COUNT
7771 036774 010346 MOV     R3,-(SP)      ;;SAVE R3
7772 036776 010446 MOV     R4,-(SP)      ;;SAVE R4
7773 037000 010546 MOV     R5,-(SP)      ;;SAVE R5
7774 037 02 113704 037153 MOVVB   SOMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
7775 037006 005404 NEG     R4
7776 037010 062704 000006 ADD     #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
7777 037014 110437 037152 MOVVB   R4,SOMODE  ;;SAVE IT FOR USE
7778 037020 113704 037151 MOVVB   SOFILL,R4  ;;GET THE ZERO FILL SWITCH
7779 037024 016605 000012 MOV     12(SP),R5  ;;PICKUP THE INPUT NUMBER
7780 037030 005003 CLR     R3      ;;CLEAR THE OUTPUT WORD
7781 037032 006105 1$: ROL    R5      ;;ROTATE MSB INTO "C"
7782 037034 000404 BR      3$      ;;GO DO MSB
7783 037036 006105 2$: ROL    R5      ;;FORM THIS DIGIT
7784 037040 006105 ROL    R5
7785 037042 006105 ROL    R5
7786 037044 010503 MOV     R5,R3
7787 037046 006103 3$: ROL    R3      ;;GET LSB OF THIS DIGIT
7788 037050 105337 037152 DECB   SOMODE      ;;TYPE THIS DIGIT?
7789 037054 100016 BPL    7$      ;;BR IF NO
7790 037056 042703 177770 BIC    #177770,R3 ;;GET RID OF JUNK
7791 037062 001002 BNE    4$      ;;TEST FOR 0
7792 037064 005704 TST    R4      ;;SUPPRESS THIS 0?
7793 037066 001403 BEQ    5$      ;;BR IF YES
7794 037070 005204 4$: INC    R4      ;;DON'T SUPPRESS ANYMORE 0'S
7795 037072 052703 000060 BIS    #'0,R3    ;;MAKE THIS DIGIT ASCII
7796 037076 052703 000040 5$: BIS    #' ,R3  ;;MAKE ASCII IF NOT ALREADY
  
```

G12

```

7797 037102 110337 037146          MOVB   R3,B5          ;;SAVE FOR TYPING
7798 037106 104401 037146          TYPE   B5           ;;GO TYPE THIS DIGIT
7799 037112 105337 037150          7$:   DECB   $OCNT    ;;COUNT BY 1
7800 037116 003347          BGT    2$          ;;BR IF MORE TO DO
7801 037120 002402          BLT    6$          ;;BR IF DONE
7802 037122 005204          INC    R4           ;;INSURE LAST DIGIT ISN'T A BLANK
7803 037124 000744          BR     2$          ;;GO DO THE LAST DIGIT
7804 037126 012605          6$:   MOV    (SP)+,R5  ;;RESTORE R5
7805 037130 012604          MOV    (SP)+,R4  ;;RESTORE R4
7806 037132 012603          MOV    (SP)+,R3  ;;RESTORE R3
7807 037134 016666 000002 000004  MOV    2(SP),4(SP) ;;SET THE STACK FOR RETURNING
7808 037142 012616          MOV    (SP)+,(SP)
7809 037144 000002          RTI                    ;;RETURN
7810 037146          000          8$:   .BYTE   0          ;;STORAGE FOR ASCII DIGIT
7811 037147          000          ;;TERMINATOR FOR TYPE ROUTINE
7812 037150          000          $OCNT: .BYTE   0          ;;OCTAL DIGIT COUNTER
7813 037151          000          $OFILL: .BYTE   0          ;;ZERO FILL SWITCH
7814 037152 000000          $OMODE: .WORD   0          ;;NUMBER OF DIGITS TO TYPE
7815          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7816
7817          ;*****
7818          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7819          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7820          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7821          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7822          ;*REPLACED WITH SPACES.
7823          ;*CALL:
7824          ;*   MOV    NUM,-(SP)          ;;PUT THE BINARY NUMBER ON THE STACK
7825          ;*   TYPDS          ;;GO TO THE ROUTINE
7826
7827          $TYPDS:
7828          037154 010046          MOV    R0,-(SP)      ;;PUSH R0 ON STACK
7829          037156 010146          MOV    R1,-(SP)      ;;PUSH R1 ON STACK
7830          037160 010246          MOV    R2,-(SP)      ;;PUSH R2 ON STACK
7831          037162 010346          MOV    R3,-(SP)      ;;PUSH R3 ON STACK
7832          037164 010546          MOV    R5,-(SP)      ;;PUSH R5 ON STACK
7833          037166 012746 020200          MOV    #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
7834          037172 016605 000020          MOV    20(SP),R5     ;;GET THE INPUT NUMBER
7835          037176 100004          BPL    1$           ;;BR IF INPUT IS POS.
7836          037200 005405          NEG    R5           ;;MAKE THE BINARY NUMBER POS.
7837          037202 112766 000055 000001  MOVB   #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
7838          037210 005000          1$:   CLR    R0           ;;ZERO THE CONSTANTS INDEX
7839          037212 012703 037370          MOV    #5DBLK,R3     ;;SETUP THE OUTPUT POINTER
7840          037216 112723 000040          MOVB   #'',(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
7841          037222 005002          2$:   CLR    R2           ;;CLEAR THE BCD NUMBER
7842          037224 016001 037360          MOV    $DTBL(R0),R1  ;;GET THE CONSTANT
7843          037230 160105          3$:   SUB    R1,R5       ;;FORM THIS BCD DIGIT
7844          037232 002402          BLT    4$           ;;BR IF DONE
7845          037234 005202          INC    R2           ;;INCREASE THE BCD DIGIT BY 1
7846          037236 000774          BR     3$
7847          037240 060105          4$:   ADD    R1,R5       ;;ADD BACK THE CONSTANT
7848          037242 005702          TST    R2           ;;CHECK IF BCD DIGIT=0
7849          037244 001002          BNE    5$           ;;FALL THROUGH IF 0
7850          037246 105716          TSTB  (SP)          ;;STILL DOING LEADING 0'S?
7851          037250 100407          BMI    7$           ;;BR IF YES
7852          037252 106316          5$:   ASLB   (SP)       ;;MSD?
    
```



```

7909
7910      ;*TK SERVICE ROUTINE
7911      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
7912      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
7913      ;*IT IN THE QUEUE.
7914      ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
7915      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
7916
7917 037460 117746 141462 $TKSRV: MOVB    @STKB,-(SP)      ;; PICKUP THE CHARACTER
7918 037464 042716 177600      BIC    #↑C177,(SP)      ;; STRIP THE JUNK
7919 037470 021627 000003      CMP    (SP),#3         ;; IS IT A CONTROL C?
7920 037474 001007      BNE    1$              ;; BRANCH IF NO
7921 037476 104401 040574      TYPE   ,SCNTLC        ;; TYPE A CONTROL-C (↑C)
7922 037502 004737 037410      JSR    PC,$TKINT      ;; INIT THE KEYBOARD
7923 037506 005726      TST    (SP)+          ;; CLEAN UP STACK
7924 037510 000137 036364      JMP    CTRHLT         ;; CONTROL C RESTART
7925 037514 021627 000007      1$:   CMP    (SP),#7     ;; IS IT A CONTROL G?
7926 037520 001004      BNE    2$              ;; BRANCH IF NO
7927 037522 022737 000176 001140  CMP    #SWREG,SWR     ;; IS SOFT-SWR SELECTED?
7928 037530 001500      BLO    6$              ;; GO TO SWR CHANGE
7929
7930      2$:
7931 037532 022737 000001 037400      CMP    #1,$TKCNT     ;; IS THE QUEUE FULL?
7932 037540 001004      BNE    3$              ;; BRANCH IF NO
7933 037542 104401 001240      TYPE   ,SBELL         ;; RING THE TTY BELL
7934 037546 005726      TST    (SP)+          ;; CLEAN CHARACTER OFF OF STACK
7935 037550 000451      BR     5$              ;; EXIT
7936 037552 021627 000023      3$:   CMP    (SP),#23     ;; IS IT A CONTROL-S?
7937 037556 001021      BNE    32$            ;; BRANCH IF NO
7938 037560 005077 141360      CLR    @STKS         ;; DISABLE TTY KEYBOARD INTERRUPTS
7939 037564 005726      TST    (SP)+          ;; CLEAN CHAR OFF STACK
7940 037566 105777 141352      31$:  TSTB   @STKS         ;; WAIT FOR A CHAR
7941 037572 100375      BPL    31$            ;; LOOP UNTIL ITS THERE
7942 037574 117746 141346      MOVB   @STKB,-(SP)   ;; GET THE CHARACTER
7943 037600 042716 177600      BIC    #↑C177,(SP)   ;; MAKE IT 7-BIT ASCII
7944 037604 022627 000021      CMP    (SP)+,#21     ;; IS IT A CONTROL-Q?
7945 037610 001366      BNE    31$            ;; BRANCH IF NO
7946 037612 012777 000100 141324      MOV    #100,@STKS    ;; REENABLE TTY KEYBOARD INTERRUPTS
7947 037620 000002      RTI                               ;; RETURN
7948 037622 005237 037400      32$:  INC    $TKCNT        ;; COUNT THIS CHARACTER
7949 037626 021627 000140      CMP    (SP),#140     ;; IS IT UPPER CASE?
7950 037632 002405      BLT    4$              ;; BRANCH IF YES
7951 037634 021627 000175      CMP    (SP),#175     ;; IS IT A SPECIAL CHAR?
7952 037640 003002      BGT    4$              ;; BRANCH IF YES
7953 037642 042716 000040      BIC    #40,(SP)      ;; MAKE IT UPPER CASE
7954 037646 112677 177530      4$:   MOVB   (SP)+,@STKQIN ;; AND PUT IT IN QUEUE
7955 037652 005237 037402      INC    $TKQIN        ;; UPDATE THE POINTER
7956 037656 023727 037402 037407      CMP    $TKQIN,$STKQEND ;; GO OFF THE END?
7957 037664 001003      BNE    5$              ;; BRANCH IF NO
7958 037666 012737 037406 037402      MOV    #STKQSRST,$TKQIN ;; RESET THE POINTER
7959 037674 000002      5$:   RTI                               ;; RETURN
7960
7961      ;*****
7962      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7963      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7964      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP

```



```

7965      : *CALL WHEN OPERATING IN TTY INTERRUPT MODE.
7966 037676 022737 000176 001140 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED
7967 037704 001124          BNE      15$          ;; EXIT IF NOT
7968 037706 105777 141232          TSTB     @STKS          ;; IS A CHAR WAITING?
7969 037712 100121          BPL      15$          ;; IF NOT, EXIT
7970 037714 117746 141226          MOVB     @STKB,-(SP)      ;; YES
7971 037720 042716 177600          BIC     #↑C17↑,(SP)    ;; MAKE IT 7-BIT ASCII
7972 037724 021627 000007          CMP     (SP),#7       ;; IS IT A CONTROL-G?
7973 037730 001300          BNE     2$           ;; IF NOT, PUT IT IN THE TTY QUEUE
7974          ;; AND EXIT
7975
7976          ;:*****
7977          ;: *CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7978          ;: *ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7979          ;: *CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
7980 037732 123727 001134 000001 6$:  CMPB     $AUTOB,#1      ;; ARE WE RUNNING IN AUTO-MODE?
7981 037740 001674          BEQ     2$           ;; BRANCH IF YES
7982 037742 005726          TST     (SP)+         ;; CLEAR CONTROL-G OFF STACK
7983 037744 004737 037410          JSR     PC,$TKINT     ;; FLUSH THE TTY INPUT QUEUE
7984 037750 005077 141170          CLR     @STKS        ;; DISABLE TTY KEYBOARD INTERRUPTS
7985 037754 112737 000001 001135          MOVB     #1,$INTAG    ;; SET INTERRUPT MODE INDICATOR
7986
7987 037762 104401 040606          TYPE    ,$CNTLG      ;; ECHO THE CONTROL-G (↑G)
7988 037766 104401 040613          $GTSWR: TYPE    ,$MSWR      ;; TYPE CURRENT CONTENTS
7989 037772 013746 000176          MOV     $WREG,-(SP)  ;; SAVE SWREG FOR TYPEOUT
7990 037776 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
7991 040000 104401 040624          TYPE    ,$MNEW      ;; PROMPT FOR NEW SWR
7992 040004 005046          19$:  CLR     -(SP)      ;; CLEAR COUNTER
7993 040006 005046          CLR     -(SP)      ;; THE NEW SWR
7994 040010 105777 141130          7$:  TSTB     @STKS        ;; CHAR THERE?
7995 040014 100375          BPL     7$         ;; IF NOT TRY AGAIN
7996
7997 040016 117746 141124          MOVB     @STKB,-(SP)  ;; PICK UP CHAR
7998 040022 042716 177600          BIC     #↑C17↑,(SP)  ;; MAKE IT 7-BIT ASCII
7999
8000 040026 021627 000003          CMP     (SP),#3      ;; IS IT A CONTROL-C?
8001 040032 001015          BNE     9$         ;; BRANCH IF NOT
8002 040034 104401 040574          TYPE    ,$CNTLC     ;; YES, ECHO CONTROL-C (↑C)
8003 040040 062706 000006          ADD     #6,SP        ;; CLEAN UP STACK
8004 040044 123727 001135 000001          CMPB     $INTAG,#1   ;; REENABLE TTY KEYBOARD INTERRUPTS?
8005 040052 001003          BNE     8$         ;; BRANCH IF NO
8006 040054 012777 000100 141062          MOV     #100,@STKS  ;; ALLOW TTY KEYBOARD INTERRUPTS
8007 040062 000137 036364          8$:  JMP     CTRHLT     ;; CONTROL-C RESTART
8008
8009
8010 040066 021627 000025          9$:  CMP     (SP),#25   ;; IS IT A CONTROL-U?
8011 040072 001005          BNE     10$        ;; BRANCH IF NOT
8012 040074 104401 040601          TYPE    ,$CNTLU     ;; YES, ECHO CONTROL-U (↑U)
8013 040100 062706 000006          20$:  ADD     #6,SP        ;; IGNORE PREVIOUS INPUT
8014 040104 000737          BR     19$        ;; LET'S TRY IT AGAIN
8015
8016
8017 040106 021627 000015          10$:  CMP     (SP),#15   ;; IS IT A <CR>?
8018 040112 001022          BNE     16$        ;; BRANCH IF NO
8019 040114 005766 000004          TST     4(SP)       ;; YES, IS IT THE FIRST CHAR?
8020 040120 001403          BEQ     11$        ;; BRANCH IF YES
    
```



```

8021 040122 016677 000002 141010      MOV      2(SP),@SWR      ;;SAVE NEW SWR
8022 040130 062706 000006      11$:    ADD      #6,SP      ;;CLEAR UP STACK
8023 040134 104401 001245      14$:    TYPE     $CRLF      ;;ECHO <CR> AND <LF>
8024 040140 123727 001135 000001      CMPB    $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
8025 040146 001003      BNE     15$      ;;BRANCH IF NOT
8026 040150 012777 000100 140766      MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
8027 040156 000002      15$:    RTI      ;;RETURN
8028 040160 004737 036656      16$:    JSR     PC,$TYPEC      ;;ECHO CHAR
8029 040164 021627 000060      CMP     (SP),#60      ;;CHAR < 0?
8030 040170 002420      BLT     18$      ;;BRANCH IF YES
8031 040172 021627 000067      CMP     (SP),#67      ;;CHAR > ??
8032 040176 003015      BGT     18$      ;;BRANCH IF YES
8033 040200 042726 000060      BIC     #60,(SP)+      ;;STRIP-OFF ASCII
8034 040204 005766 000002      TST     2(SP)      ;;IS THIS THE FIRST CHAR
8035 040210 001403      BEQ     17$      ;;BRANCH IF YES
8036 040212 006316      ASL     (SP)      ;;NO, SHIFT PRESENT
8037 040214 006316      ASL     (SP)      ;;CHAR OVER TO MAKE
8038 040216 006316      ASL     (SP)      ;;ROOM FOR NEW ONE.
8039 040220 005266 000002      17$:    INC     2(SP)      ;;KEEP COUNT OF CHAR
8040 040224 056616 177776      BIS     -2(SP),(SP)      ;;SET IN NEW CHAR
8041 040230 000667      BR      7$      ;;GET THE NEXT ONE
8042 040232 104401 001244      18$:    TYPE     $QUES      ;;TYPE ?<CR><LF>
8043 040236 000720      BR      20$      ;;SIMULATE CONTROL-U
8044      .DSABL  LSB

```

```

8045
8046
8047      ;;*****
8048      ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8049      ;;CALL:
8050      ;;
8051      ;;      RDCHR      ;;GET A CHARACTER FROM THE QUEUE
8052      ;;      RETURN HERE      ;;CHARACTER IS ON THE STACK
8053      ;;
8054      ;;      WITH PARITY BIT STRIPPED OFF

```

```

8055 040240 011646 000004 000002 $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
8056 040242 016666 000004      MOV     4(SP),2(SP)      ;;THE PS
8057 040250 005066 000004      CLR     4(SP)      ;;GET READY FOR A CHARACTER
8058 040254 005046      CLR     -(SP)      ;;PUT NEW PS ON STACK
8059 040256 012746 040264      MOV     #64$,-(SP)      ;;PUT NEW PC ON STACK
8060 040262 000002      RTI      ;;POP NEW PC AND PS
8061 040264      64$:
8062 040264 005737 037400      1$:    TST     $TKCNT      ;;WAIT ON A CHARACTER
8063 040270 001775      BEQ     1$
8064 040272 005337 037400      DEC     $TKCNT      ;;DECREMENT THE COUNTER
8065 040276 117766 177102 000004      MOVB   @$TKQOUT,4(SP)      ;;GET ONE CHARACTER
8066 040304 005237 037404      INC     $TKQOUT      ;;UPDATE THE POINTER
8067 040310 023727 037404 037407      CMP     $TKQOUT,$$TKQEND      ;;DID IT GO OFF OF THE END?
8068 040316 001003      BNE     2$      ;;BRANCH IF NO
8069 040320 012737 037406 037404      MOV     $$TKQSRT,$TKQOUT      ;;RESET THE POINTER
8070 040326 000002      2$:    RTI      ;;RETURN

```

```

8071      ;;*****
8072      ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
8073      ;;CALL:
8074      ;;
8075      ;;      RDLIN      ;;INPUT A STRING FROM THE TTY
8076      ;;      RETURN HERE      ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8077      ;;      TERMINATOR WILL BE A BYTE OF ALL 0'S

```

8077											
8078	040330	010346			\$RDLIN:	MOV	R3, -(SP)	;	SAVE R3		
8079	040332	005046				CLR	-(SP)	;	CLEAR THE RUBOUT KEY		
8080	040334	012703	040564		1\$:	MOV	#STTYIN, R3	;	GET ADDRESS		
8081	040340	022703	040574		2\$:	CMP	#STTYIN+8., R3	;	BUFFER FULL?		
8082	040344	101456				BLOS	4\$;	BR IF YES		
8083	040346	104410				RDCHR		;	GO READ ONE CHARACTER FROM THE TTY		
8084	040350	112613				MOV	(SP)+, (R3)	;	GET CHARACTER		
8085	040352	122713	000177		10\$:	CMP	#177, (R3)	;	IS IT A RUBOUT		
8086	040356	001022				BNE	5\$;	BR IF NO		
8087	040360	005716				TST	(SP)	;	IS THIS THE FIRST RUBOUT?		
8088	040362	001007				BNE	6\$;	BR IF NO		
8089	040364	112737	000134	040562		MOV	#'\, 9\$;	TYPE A BACK SLASH		
8090	040372	104401	040562			TYPE	9\$				
8091	040376	012716	177777			MOV	#-1, (SP)	;	SET THE RUBOUT KEY		
8092	040402	005303			6\$:	DEC	R3	;	BACKUP BY ONE		
8093	040404	020327	040564			CMP	R3, #STTYIN	;	STACK EMPTY?		
8094	040410	103434				BLO	4\$;	BR IF YES		
8095	040412	111337	040562			MOV	(R3), 9\$;	SETUP TO TYPEOUT THE DELETED CHAR.		
8096	040416	104401	040562			TYPE	9\$;	GO TYPE		
8097	040422	000746				BR	2\$;	GO READ ANOTHER CHAR.		
8098	040424	005716			5\$:	TST	(SP)	;	RUBOUT KEY SET?		
8099	040426	001406				BEQ	7\$;	BR IF NO		
8100	040430	112737	000134	040562		MOV	#'\, 9\$;	TYPE A BACK SLASH		
8101	040436	104401	040562			TYPE	9\$				
8102	040442	005016				CLR	(SP)	;	CLEAR THE RUBOUT KEY		
8103	040444	122713	000025		7\$:	CMP	#25, (R3)	;	IS CHARACTER A CTRL U?		
8104	040450	001003				BNE	9\$;	BR IF NO		
8105	040452	104401	040601			TYPE	\$CNTLU	;	TYPE A CONTROL "U"		
8106	040456	000726				BR	1\$;	GO START OVER		
8107	040460	122713	000022		8\$:	CMP	#22, (R3)	;	IS CHARACTER A "↑R"?		
8108	040464	001011				BNE	3\$;	BRANCH IF NO		
8109	040466	105013				CLRB	(R3)	;	CLEAR THE CHARACTER		
8110	040470	104401	001245			TYPE	\$CRLF	;	TYPE A "CR" & "LF"		
8111	040474	104401	040564			TYPE	\$STTYIN	;	TYPE THE INPUT STRING		
8112	040500	000717				BR	2\$;	GO PICKUP ANOTHER CHARACTER		
8113	040502	104401	001244		4\$:	TYPE	\$QUES	;	TYPE A "?"		
8114	040506	000712				BR	1\$;	CLEAR THE BUFFER AND LOOP		
8115	040510	111337	040562		3\$:	MOV	(R3), 9\$;	ECHO THE CHARACTER		
8116	040514	104401	040562			TYPE	9\$				
8117	040520	122723	000015			CMP	#15, (R3)+	;	CHECK FOR RETURN		
8118	040524	001305				BNE	2\$;	LOOP IF NOT RETURN		
8119	040526	105063	177777			CLRB	-1(R3)	;	CLEAR RETURN (THE 15)		
8120	040532	104401	001246			TYPE	\$LF	;	TYPE A LINE FEED		
8121	040536	005726				TST	(SP)+	;	CLEAR RUBOUT KEY FROM THE STACK		
8122	040540	012603				MOV	(SP)+, R3	;	RESTORE R3		
8123	040542	011646				MOV	(SP), -(SP)	;	ADJUST THE STACK AND PUT ADDRESS OF THE		
8124	040544	016666	000004	000002		MOV	4(SP), 2(SP)	;	FIRST ASCII CHARACTER ON IT		
8125	040552	012766	040564	000004		MOV	#STTYIN, 4(SP)				
8126	040560	000002				RTI		;	RETURN		
8127	040562	000			9\$:	.BYTE	0	;	STORAGE FOR ASCII CHAR. TO TYPE		
8128	040563	000				.BYTE	0	;	TERMINATOR		
8129	040564	000010			\$TTYIN:	.BLKB	8.	;	RESERVE 8 BYTES FOR TTY INPUT		
8130	040574	041536	005015	000	\$CNTLC:	.ASCIZ	/↑C/<15><12>	;	CONTROL "C"		
8131	040601	136	006525	000012	\$CNTLU:	.ASCIZ	/↑U/<15><12>	;	CONTROL "U"		
8132	040606	043536	005015	000	\$CNTLG:	.ASCIZ	/↑G/<15><12>	;	CONTROL "G"		

M12

```

8133 040613 015 051412 051127 $MSWR: .ASCIZ <15><12>/SWR = /
8134 040620 036440 000040
8135 040624 020040 042516 020127 $MNEW: .ASCIZ / NEW = /
8136 040632 020075 000
8137 040636
8138 .EVEN
8139 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
8140
8141 ;*****
8142 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
8143 ;*CHANGE IT TO BINARY.
8144 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
8145 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
8146 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
8147 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
8148 ;*CALL:
8149 ;*      RDOCT          ;; READ AN OCTAL NUMBER
8150 ;*      RETURN HERE   ;; LOW ORDER BITS ARE ON TOP OF THE STACK
8151 ;*                   ;; HIGH ORDER BITS ARE IN $HIOCT
8152 040636 011646 $RDOCT: MOV      (SP),-(SP)      ;; PROVIDE SPACE FOR THE
8153 040640 016666 000004 000002 MOV      4(SP),2(SP)      ;; INPUT NUMBER
8154 040646 010046 MOV      RO,-(SP)       ;; PUSH RO ON STACK
8155 040650 010146 MOV      R1,-(SP)       ;; PUSH R1 ON STACK
8156 040652 010246 MOV      R2,-(SP)       ;; PUSH R2 ON STACK
8157 040654 104411 1$:      RDLIN          ;; READ AN ASCIZ LINE
8158 040656 012600 MOV      (SP)+,RO        ;; GET ADDRESS OF 1ST CHARACTER
8159 040660 010037 040764 MOV      RO,$5         ;; AND SAVE IT
8160 040664 005001 CLR      R1             ;; CLEAR DATA WORD
8161 040666 005002 CLR      R2
8162 040670 112046 2$:      MOVVB     (RO)+,-(SP)      ;; PICKUP THIS CHARACTER
8163 040672 001420 BEQ      3$            ;; IF ZERO GET OUT
8164 040674 122716 000060 CMPB     #'0,(SP)      ;; MAKE SURE THIS CHARACTER
8165 040700 003026 BGT      4$            ;; IS AN OCTAL DIGIT
8166 040702 122716 000067 CMPB     #'7,(SP)
8167 040706 002423 BLT      4$
8168 040710 006301 ASL     R1              ;; *2
8169 040712 006102 ROL     R2
8170 040714 006301 ASL     R1              ;; *4
8171 040716 006102 ROL     R2
8172 040720 006301 ASL     R1              ;; *8
8173 040722 006102 ROL     R2
8174 040724 042716 177770 BIC     #'C7,(SP)      ;; STRIP THE ASCII JUNK
8175 040730 062601 ADD     (SP)+,R1       ;; ADD IN THIS DIGIT
8176 040732 000756 BR      2$            ;; LOOP
8177 040734 005726 3$:      TST     (SP)+          ;; CLEAN TERMINATOR FROM STACK
8178 040736 010166 000012 MOV     R1,12(SP)      ;; SAVE THE RESULT
8179 040742 010237 040774 MOV     R2,$HIOCT
8180 040746 012602 MOV     (SP)+,R2      ;; POP STACK INTO R2
8181 040750 012601 MOV     (SP)+,R1      ;; POP STACK INTO R1
8182 040752 012600 MOV     (SP)+,RO      ;; POP STACK INTO RO
8183 040754 000002 RTI
8184 040756 005726 4$:      TST     (SP)+          ;; CLEAN PARTIAL FROM STACK
8185 040760 105010 CLRB   (RO)           ;; SET A TERMINATOR
8186 040762 104401 TYPE
8187 040764 000000 5$:      .WORD    0          ;; TYPE UP THRU THE BAD CHAR.
8188 040766 104401 001244 TYPE    ,SQUES      ;; "?" "CR" & "LF"
    
```

8189 040772 000730
8190 040774 000000
8191
8192
8193
8194
8195
8196
8197
8198
8199
8200
8201
8202
8203
8204
8205
8206
8207
8208 040776
8209 040776 010046
8210 041000 010146
8211 041002 010246
8212 041004 010346
8213 041006 010446
8214 041010 010546
8215 041012 016646 000022
8216 041016 016646 000022
8217 041022 016646 000022
8218 041026 016646 000022
8219 041032 000002
8220
8221
8222
8223
8224 041034
8225 041034 012666 000022
8226 041040 012666 000022
8227 041044 012666 000022
8228 041050 012666 000022
8229 041054 012605
8230 041056 012604
8231 041060 012603
8232 041062 012602
8233 041064 012601
8234 041066 012600
8235 041070 000002
8236
8237
8238
8239
8240
8241
8242 041072 017737 140042 002356
8243 041100 012737 041120 000024
8244 041106 012737 000340 0C0026

```
BR 15 ;;TRY AGAIN
$HI OCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

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

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

.SBTTL POWER DOWN AND UP ROUTINE

;*****
:POWER DOWN ROUTINE
$PWRDN: MOV $SWR,SAVSWR ;;SAVE SWITCH REGISTER
MOV $PWRUP,PWRVEC ;;SET UP VECTOR
MOV #PR7,PWRVEC+2
```

```

0245 041114 000000          HALT
0246 041116 000776          BR      -2      ;HANG UP
0247
0248      ;:*****
0249
0250      ;POWER UP ROUTINE
0251 041120 005037 041214  $PWRUP: CLR      $PWRCT      ;LOAD WAIT COUNT
0252 041122 012737 000144 041216  MOV      #100,$PWRCT+2
0253 041130 005237 041214  IS:      INC      $PWRCT      ;WAIT FOR TELETYPE
0254 041136 001375          BNE      IS
0255 041140 005337 041216  DEC      $PWRCT+2
0256 041144 001372          BNE      IS
0257 041146 012737 041072 000024  MOV      $PWRDN,PWRVEC ;SET UP FOR POWER DOWN VECTOR
0258 041154 012737 000340 000026  MOV      #PR7,PWRVEC+2
0259 041162 012706 001100  MOV      $STACK,SP      ;FORCE STACK
0260 041166 104401 041220  TYPE      $POWER      ;TYPE POWER
0261 041172 004737 074314  JSR      PC,PARCHK      ;REINITIALIZE MEMORY CHECK ENABLE
0262 041176 013777 002356 137734  MOV      SAVSWR,JSWR      ;RESTORE SWITCH REGISTER
0263 041204 013702 001324  MOV      $BASE,R2      ;REINITIALISE R2 FOR '611 BASE
0264 041210 000177 137672  JMP      $SLPADR ;GO BACK TO LAST TEST
0265
0266 041214 000000 000000  $PWRCT: .WORD 0,0      ;TELETYPE TIME OUT
0267 041220 047520 042527 000122  $POWER: .ASCIZ 'POWER'
0268          EVEN
0269      .SBTTL TRAP DECODER
0270
0271      ;:*****
0272      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
0273      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
0274      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
0275      ;*GO TO THAT ROUTINE.
0276
0277 041226 010046          STRAP: MOV      R0,-(SP)      ;;SAVE R0
0278 041230 016600 000002  MOV      2(SP),R0      ;;GET TRAP ADDRESS
0279 041234 005740          TST      -(R0)      ;;BACKUP BY 2
0280 041236 111000          MOVB     (R0),R0      ;;GET RIGHT BYTE OF TRAP
0281 041240 006300          ASL     R0      ;;POSITION FOR INDEXING
0282 041242 016000 041262  MOV      $TRPAD(R0),R0 ;;INDEX TO TABLE
0283 041246 000200          RTS     R0      ;;GO TO ROUTINE
0284
0285
0286      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
0287
0288 041250 011646          STRAP2: MOV     (SP),-(SP) ;;MOVE THE PC DOWN
0289 041252 016666 000004 000002  MOV     4(SP),2(SP) ;;MOVE THE PSW DOWN
0290 041260 000002          RTI     ;;RESTORE THE PSW
0291
0292      .SBTTL TRAP TABLE
0293
0294      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
0295      ;*BY THE "TRAP" INSTRUCTION.
0296
0297      :
0298      ROUTINE
0299      -----
0300 041262 041250          $TRPAD: .WORD  $STRAP2
0300 041264 036444          $TYPE  ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE

```

8301	041266	036752	\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8302	041270	036726	\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
8303	041272	036766	\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
8304	041274	037154	\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
8305						
8306	041276	037766	\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
8307						
8308	041300	037676	\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
8309	041302	040240	\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
8310	041304	040330	\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
8311	041306	040636	\$RDOCT	::CALL=RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
8312	041310	040776	\$SAVREG	::CALL=SAVREG	TRAP+13(104413)	SAVE R0-R5 ROUTINE
8313	041312	041034	\$RESREG	::CALL=RESREG	TRAP+14(104414)	RESTORE R0-R5 ROUTINE
8314	041314	035374	\$SCOPI\$::CALL=SCOPI	TRAP+15(104415)	INTERNAL LOOP ON ERROR

.SBTTL DATA TABLE FOR PRINT OUT				
8315				
8316				
8317	041316	001254	002310	DT000: .WORD \$TESTN,TRAPPC
8318	041322	001254	001116 002220	DT0004: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,T.CS2,T.ER,T.DS
8319	041330	002160	002170 002174	
8320	041336	002172		
8321	041340	001254	001116 002244	DT0005: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1
8322	041346	002204		
8323	041350	001254	001116 002246	DT0006: .WORD \$TESTN,\$ERRPC,E.MR2,T.MR2
8324	041356	002206		
8325	041360	001254	001116 002250	DT0007: .WORD \$TESTN,\$ERRPC,E.MR3,T.MR3
8326	041366	002210		
8327	041370	001254	001116 002244	DT0010: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
8328	041376	002204	002320 002322	
8329	041404	002326		
8330	041406	001254	001116 002220	DT0011: .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,PR.BIT,M1.BIT,BITCNT,T.CS2,T.ER,T.DS
8331	041414	002160	002320 002322	
8332	041422	002326	002170 002174	
8333	041430	002172		
8334	041432	001254	001116 002240	DT0014: .WORD \$TESTN,\$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
8335	041440	002200	002226 002166	
8336	041446	001254	001116 002244	DT0017: .WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT
8337	041454	002204	002316 002320	
8338	041462	002322	002324 002326	
8339	041470	001254	001116 002254	DT0021: .WORD \$TESTN,\$ERRPC,E.ECPT,T.ECPT,BITCNT
8340	041476	002214	002326	
8341	041502	001254	001116 002160	DT0026: .WORD \$TESTN,\$ERRPC,T.CS1,T.CS2,T.ER,T.DS,T.WC,T.DCYL,T.DA
8342	041510	002170	002174 002172	
8343	041516	002162	002200 002166	
8344	041524	001254	001116	DT0037: .WORD \$TESTN,\$ERRPC
8345	041530	001162	001164 001166	DT0040: .WORD \$REG0,\$REG1,\$REG2
8346	041536	001254	001116 002254	DT0041: .WORD \$TESTN,\$ERRPC,E.ECPT,T.ECPT,E.ECPS,T.ECPS,BITCNT
8347	041544	002214	002252 002212	
8348	041552	002326		
8349	041554	001254	001116 002252	DT0043: .WORD \$TESTN,\$ERRPC,E.ECPS,T.ECPS,BITCNT
8350	041562	002212	002326	
8351	041566	001254	001116 002234	DT0045: .WORD \$TESTN,\$ERRPC,E.ER,T.ER
8352	041574	002174		
8353	041576	001254	001116 002224	DT0046: .WORD \$TESTN,\$ERRPC,E.BA,T.BA
8354	041604	002164		
8355	041606	001254	001116 002222	DT0062: .WORD \$TESTN,\$ERRPC,E.WC,T.WC
8356	041614	002162		
8357	041616	001254	001116 001162	DT0063: .WORD \$TESTN,\$ERRPC,\$REG0,\$REG3,\$REG1,\$REG4,\$REG2
8358	041624	001170	001164 001172	
8359	041632	001166		

.SBTTL DATA FORMAT FOR PRINT OUT			
8360			
8361			
8362	041634	000001	
8363	041636	002	000
8364	041640	000006	
8365	041642	000	000
8366	041644	043040	
8367	041646	000	000
8368	041650	043056	
8369	041652	002	000
8370	041654	043122	
8371	041656	000	000
8372	041660	043141	
8373	041662	002	000
8374	041664	043231	
8375	041666	003	000
8376	041670	000005	
8377	041672	000	000
8378	041674	043040	
8379	041676	000	000
8380	041700	043056	
8381	041702	002	000
8382	041704	043122	
8383	041706	000	000
8384	041710	043157	
8385	041712	002	000
8386	041714	000005	
8387	041716	000	000
8388	041720	043040	
8389	041722	000	000
8390	041724	043056	
8391	041726	002	000
8392	041730	043122	
8393	041732	000	000
8394	041734	043175	
8395	041736	002	000
8396	041740	000005	
8397	041742	000	000
8398	041744	043040	
8399	041746	000	000
8400	041750	043056	
8401	041752	002	000
8402	041754	043122	
8403	041756	000	000
8404	041760	043213	
8405	041762	002	000
8406	041764	000005	
8407	041766	000	000
8408	041770	043040	
8409	041772	000	000
8410	041774	043056	
8411	041776	002	000
8412	042000	043325	
8413	042002	000	000
8414	042004	043371	
8415	042006	005	000

```

DF000: .WORD 1
        .BYTE 2,0
DF0004: .WORD 6,0
        .BYTE 0,0
        .WORD 04000A
        .BYTE 0,0
        .WORD 04000B
        .BYTE 2,0
        .WORD 04004
        .BYTE 0,0
        .WORD 04004A
        .BYTE 2,0
        .WORD 04004E
        .BYTE 3,0
DF0005: .WORD 5,0
        .BYTE 0,0
        .WORD 04000A
        .BYTE 0,0
        .WORD 04000B
        .BYTE 2,0
        .WORD 04004
        .BYTE 0,0
        .WORD 04004B
        .BYTE 2,0
DF0006: .WORD 5,0
        .BYTE 0,0
        .WORD 04000A
        .BYTE 0,0
        .WORD 04000B
        .BYTE 2,0
        .WORD 04004
        .BYTE 0,0
        .WORD 04004C
        .BYTE 2,0
DF0007: .WORD 5,0
        .BYTE 0,0
        .WORD 04000A
        .BYTE 0,0
        .WORD 04000B
        .BYTE 2,0
        .WORD 04004
        .BYTE 0,0
        .WORD 04004D
        .BYTE 2,0
DF0010: .WORD 5,0
        .BYTE 0,0
        .WORD 04000A
        .BYTE 0,0
        .WORD 04000B
        .BYTE 2,0
        .WORD 04010
        .BYTE 0,0
        .WORD 04010A
        .BYTE 5,0

```


042010	000006	000
042012	000	000
042014	043040	000
042016	000	000
042020	043056	000
042022	002	000
042024	043325	000
042026	000	000
042028	043437	000
042030	005	000
042034	043231	000
042036	003	000
042040	000005	000
042042	000	000
042044	043040	000
042046	000	000
042048	043056	000
042050	002	000
042054	043505	000
042056	000	000
042060	043544	000
042062	004	000
042064	000005	000
042066	000	000
042070	043040	000
042072	000	000
042074	043056	000
042076	002	000
042100	043601	000
042102	000	000
042104	043665	000
042106	007	000
042110	000005	000
042112	000	000
042114	043040	000
042116	000	000
042120	043056	000
042122	002	000
042124	043753	000
042126	000	000
042130	043777	000
042132	003	000
042134	000004	000
042136	000	000
042140	043040	000
042142	000	000
042144	043056	000
042146	002	000
042150	044053	000
042152	007	000
042154	000003	000
042156	000	000
042160	043056	000
042162	002	000
042164	044053	000
042166	007	000

DF0011:	.WORD	6
	.BYTE	0,0
	.WORD	DH000A
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH010
	.BYTE	0,0
	.WORD	DH010B
	.BYTE	5,0
	.WORD	DH004E
	.BYTE	3,0
DF0014:	.WORD	5
	.BYTE	0,0
	.WORD	DH000A
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH014
	.BYTE	0,0
	.WORD	DH014A
	.BYTE	4,0
DF0017:	.WORD	5
	.BYTE	0,0
	.WORD	DH000A
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH0017
	.BYTE	0,0
	.WORD	DH017A
	.BYTE	7,0
DF0021:	.WORD	5
	.BYTE	0,0
	.WORD	DH000A
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH0021
	.BYTE	0,0
	.WORD	DH021A
	.BYTE	3,0
DF0026:	.WORD	4
	.BYTE	0,0
	.WORD	DH000A
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH0026
	.BYTE	7,0
DF0027:	.WORD	3
	.BYTE	0,0
	.WORD	DH0008
	.BYTE	2,0
	.WORD	DH0026
	.BYTE	7,0

8472	042170	000004		DF0037: .WORD	4
8473	042172	000	000	.BYTE	0,0
8474	042174	043056		.WORD	DH0008
8475	042176	002	000	.BYTE	2,0
8476	042200	044140		.WORD	DH037A
8477	042202	000	000	.BYTE	0,0
8478	042204	044165		.WORD	DH037B
8479	042206	003	000	.BYTE	3,0
8480	042210	000000		DF0040: .WORD	0
8481	042212	003	000	.BYTE	3,0
8482	042214	000004		DF0041: .WORD	4
8483	042216	000	000	.BYTE	0,0
8484	042220	043056		.WORD	DH0008
8485	042222	002	000	.BYTE	2,0
8486	042224	044214		.WORD	DH0041
8487	042226	000	000	.BYTE	0,0
8488	042230	044260		.WORD	DH041A
8489	042232	005	000	.BYTE	5,0
8490	042234	000004		DF0042: .WORD	4
8491	042236	000	000	.BYTE	0,0
8492	042240	043056		.WORD	DH0008
8493	042242	002	000	.BYTE	2,0
8494	042244	043753		.WORD	DH0021
8495	042246	000	000	.BYTE	0,0
8496	042250	043777		.WORD	DH021A
8497	042252	003	000	.BYTE	3,0
8498	042254	000004		DF0043: .WORD	4
8499	042256	000	000	.BYTE	0,0
8500	042260	043056		.WORD	DH0008
8501	042262	002	000	.BYTE	2,0
8502	042264	043753		.WORD	DH0021
8503	042266	000	000	.BYTE	0,0
8504	042270	044025		.WORD	DH021B
8505	042272	003	000	.BYTE	3,0
8506	042274	000005		DF0045: .WORD	5
8507	042276	000	000	.BYTE	0,0
8508	042300	043040		.WORD	DH000A
8509	042302	000	000	.BYTE	0,0
8510	042304	043056		.WORD	DH0008
8511	042306	002	000	.BYTE	2,0
8512	042310	043122		.WORD	DH004
8513	042312	000	000	.BYTE	0,0
8514	042314	043256		.WORD	DH004F
8515	042316	002	000	.BYTE	2,0
8516	042320	000004		DF0046: .WORD	4
8517	042322	000	000	.BYTE	0,0
8518	042324	043056		.WORD	DH0008
8519	042326	002	000	.BYTE	2,0
8520	042330	043122		.WORD	DH004
8521	042332	000	000	.BYTE	0,0
8522	042334	043273		.WORD	DH004G
8523	042336	002	000	.BYTE	2,0
8524	042340	000005		DF0050: .WORD	5
8525	042342	000	000	.BYTE	0,0
8526	042344	043040		.WORD	DH000A
8527	042346	000	000	.BYTE	0,0

8528	042350	043056		.WORD	DH0008
8529	042352	002	000	.BYTE	2,0
8530	042354	043122		.WORD	DH004
8531	042356	000	000	.BYTE	0,0
8532	042360	043141		.WORD	DH004A
8533	042362	002	000	.BYTE	2,0
8534	042364	000004	DF0060:	.WORD	4
8535	042366	000	000	.BYTE	0,0
8536	042370	043056		.WORD	DH0008
8537	042372	002	000	.BYTE	2,0
8538	042374	044326		.WORD	DH060A
8539	042376	000	000	.BYTE	0,0
8540	042400	044353		.WORD	DH0608
8541	042402	003	000	.BYTE	3,0
8542	042404	000005	DF0061:	.WORD	5
8543	042406	000	000	.BYTE	0,0
8544	042410	043040		.WORD	DH000A
8545	042412	000	000	.BYTE	0,0
8546	042414	043056		.WORD	DH0008
8547	042416	002	000	.BYTE	2,0
8548	042420	043122		.WORD	DH004
8549	042422	000	000	.BYTE	0,0
8550	042424	043273		.WORD	DH004G
8551	042426	002	000	.BYTE	2,0
8552	042430	000005	DF0062:	.WORD	5
8553	042432	000	000	.BYTE	0,0
8554	042434	043040		.WORD	DH000A
8555	042436	000	000	.BYTE	0,0
8556	042440	043056		.WORD	DH0008
8557	042442	002	000	.BYTE	2,0
8558	042444	043122		.WORD	DH004
8559	042446	000	000	.BYTE	0,0
8560	042450	043310		.WORD	DH004H
8561	042452	002	000	.BYTE	2,0
8562	042454	000005	DF0063:	.WORD	5
8563	042456	000	000	.BYTE	0,0
8564	042460	043040		.WORD	DH000A
8565	042462	000	000	.BYTE	0,0
8566	042464	043056		.WORD	DH0008
8567	042466	002	000	.BYTE	2,0
8568	042470	044400		.WORD	DH063A
8569	042472	000	000	.BYTE	0,0
8570	042474	044445		.WORD	DH063B
8571	042476	005	000	.BYTE	5,0

```

8572                                     .SBTTL  ASCII MESSAGES
8573
8574 042500 005015 045522 030466 OPRO01: .ASCIZ <15><12>/RK611 VECTOR ADDRESS ( /
8575 042506 020061 042526 052103
8576 042514 051117 040440 042104
8577 042522 042522 051523 024040
8578 042530 000040
8579 042532 024440 036440 000040 OPRO02: .ASCIZ / ) = /
8580 042540 045522 030466 020061 OPRO03: .ASCIZ /RK611 VECTOR ADDRESS ( /
8581 042546 042526 052103 051117
8582 042554 040440 042104 042522
8583 042562 051523 024040 000040
8584 042570 045522 030466 020061 OPRO04: .ASCIZ /RK611 PRIORITY ( /
8585 042576 051120 047511 044522
8586 042604 054524 024040 000040
8587 042612 005015 047062 020104 OPRO06: .ASCIZ <15><12>/2ND PASS RUN TIME APPROX 4 MINUTES/<15><12>
8588 042620 040520 051523 051040
8589 042626 047125 052040 046511
8590 042634 020105 050101 051120
8591 042642 054117 032040 046440
8592 042650 047111 052125 051505
8593 042656 005015 000
8594 042661 015 025012 025052 OPRO07: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
8595 042666 025052 020040 050040
8596 042674 047522 051107 046501
8597 042702 044040 046101 042524
8598 042710 020104 020040 025052
8599 042716 025052 006452 000012
8600 042724 020040 000
8601 042727 015 050012 047522 SPACE2: .ASCIZ / /
8602 042734 051107 046501 040440 ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
8603 042742 047502 052122 042105
8604 042750 041040 041505 052501
8605 042756 042523 042440 051122
8606 042764 051117 052040 051110
8607 042772 051505 047510 042114
8608 043000 042440 041530 042505
8609 043006 042504 006504 000012
8610 043014 005015 042524 052123 TSTBY1: .ASCIZ <15><12>/TEST /
8611 043022 000040
8612 043024 041040 050131 051501 TSTBY2: .ASCIZ / BYPASSED/<15><12>
8613 043032 042523 006504 000012

```

					.SBTTL DATA HEADERS		
8614							
8615							
8616	043040	042524	052123	020040	DH000A:	.ASCIZ	/TEST ERRCR/
8617	043046	020040	051105	047522			
8618	043054	000122					
8619	043056	052516	020115	020040	DH000B:	.ASCIZ	/NUM PC/
8620	043064	020040	041520	000			
8621	043071	124	051505	020124	DH000C:	.ASCII	/TEST TRAP/<15><12>
8622	043076	020040	052040	040522			
8623	043104	006520	012				
8624	043107	116	046525	020040		.ASCIZ	/NUM PC/
8625	043114	020040	050040	000103			
8626	043122	054105	042520	052103	DH004:	.ASCIZ	/EXPECT ACTUAL/
8627	043130	020040	041501	052524			
8628	043136	046101	000				
8629	043141	122	041513	030523	DH004A:	.ASCIZ	/RKCS1 RKCS1/
8630	043146	020040	051040	041513			
8631	043154	030523	000				
8632	043157	122	046513	030522	DH004B:	.ASCIZ	/RKMR1 RKMR1/
8633	043164	020040	051040	046513			
8634	043172	030522	000				
8635	043175	122	046513	031122	DH004C:	.ASCIZ	/RKMR2 RKMR2/
8636	043202	020040	051040	046513			
8637	043210	031122	000				
8638	043213	122	046513	031522	DH004D:	.ASCIZ	/RKMR3 RKMR3/
8639	043220	020040	051040	046513			
8640	043226	031522	000				
8641	043231	122	041513	031123	DH004E:	.ASCIZ	/RKCS2 RKER RKDS/
8642	043236	020040	051040	042513			
8643	043244	020122	020040	051040			
8644	043252	042113	000123				
8645	043256	045522	051105	020040	DH004F:	.ASCIZ	/RKER RKER/
8646	043264	020040	045522	051105			
8647	043272	000					
8648	043273	122	041113	020101	DH004G:	.ASCIZ	/RKBA RKBA/
8649	043300	020040	051040	041113			
8650	043306	000101					
8651	043310	045522	041527	020040	DH004H:	.ASCIZ	/RKWC RKWC/
8652	043316	020040	045522	041527			
8653	043324	000					
8654	043325	105	050130	041505	DH010:	.ASCIZ	/EXPECT ACTUAL PRESENT PREVIOUS BIT/
8655	043332	020124	040440	052103			
8656	043340	040525	020114	050040			
8657	043346	042522	042523	052116			
8658	043354	050040	042522	047526			
8659	043362	051525	041040	052111			
8660	043370	000					
8661	043371	122	046513	030522	DH010A:	.ASCIZ	/RKMR1 RKMR1 BIT BIT COUNT/
8662	043376	020040	051040	046513			
8663	043404	030522	020040	041040			
8664	043412	052111	020040	020040			
8665	043420	041040	052111	020040			
8666	043426	020040	041440	052517			
8667	043434	052116	000				
8668	043437	122	041513	030523	DH010B:	.ASCIZ	/RKCS1 RKCS1 BIT BIT COUNT/
8669	043444	020040	051040	041513			


```

8769      .SBTTL  ERROR MESSAGES
8770
8771 044512 047125 054105 042520 EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
8772 044520 052103 042105 046440
8773 044526 046505 051117 020131
8774 044534 040520 044522 054524
8775 044542 042440 040516 046102
8776 044550 020105 051124 050101
8777 044556      000
8778 044557      105 051122 051117 EMS004: .ASCIZ /ERROR AFTER ATTEMPTING IMPLIED SEEK/
8779 044564 040440 052106 051105
8780 044572 040440 052124 046505
8781 044600 052120 047111 020107
8782 044606 046511 046120 042511
8783 044614 020104 042523 045505
8784 044622      000
8785 044623      122 041513 030523 E5004A: .ASCIZ /RKCS1 IN ERROR/
8786 044630 044440 020116 051105
8787 044636 047522 000122
8788 044642 045522 051115 020061 E5004B: .ASCIZ /RKMR1 IN ERROR/
8789 044650 047111 042440 051122
8790 044656 051117      000
8791 044661      122 046513 031122 E5004C: .ASCIZ /RKMR2 IN ERROR/
8792 044666 044440 020116 051105
8793 044674 047522 000122
8794 044700 045522 051115 020063 E5004D: .ASCIZ /RKMR3 IN ERROR/
8795 044706 047111 042440 051122
8796 044714 051117      000
8797 044717      122 042513 020122 E5004E: .ASCIZ /RKER IN ERROR/
8798 044724 047111 042440 051122
8799 044732 051117      000
8800 044735      122 041113 020101 E5004F: .ASCIZ /RKBA IN ERROR/
8801 044742 047111 042440 051122
8802 044750 051117      000
8803 044753      122 053513 020103 E5004G: .ASCIZ /RKWC IN ERROR/
8804 044760 047111 042440 051122
8805 044766 051117      000
8806 044771      105 051122 051117 EMS010: .ASCIZ /ERROR ATTEMPTING TO READ HEADER SYNC (PREAMBLE)/
8807 044776 040440 052124 046505
8808 045004 052120 047111 020107
8809 045012 047524 051040 040505
8810 045020 020104 042510 042101
8811 045026 051105 051440 047131
8812 045034 020103 050050 042522
8813 045042 046501 046102 024505
8814 045050      000
8815 045051      105 051122 051117 EMS011: .ASCIZ &ERROR ATTEMPTING TO DO HEADER READ/COMPARE&
8816 045056 040440 052124 046505
8817 045064 052120 047111 020107
8818 045072 047524 042040 020117
8819 045100 042510 042101 051105
8820 045106 051040 040505 027504
8821 045114 047503 050115 051101
8822 045122 000105
8823 045124 045522 041504 046131 E5011A: .ASCIZ /RKDCYL OR RKDA IN ERROR/
8824 045132 047440 020122 045522

```


N13

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A
 DZR6EA.P11 28-SEP-76 15:31

MACY11 27(1006) 05-OCT-76 09:12 PAGE 169

SEQ 0169

ERROR MESSAGES

8825	045140	040504	044440	020116	
8826	045146	051105	047522	000122	
8827	045154	051105	047522	020122	EM5015: .ASCIZ /ERROR ATTEMPTING TO READ GAP/
8828	045162	052101	042524	050115	
8829	045170	044524	043516	052040	
8830	045176	020117	042522	042101	
8831	045204	043440	050101	000	
8832	045211	105	051122	051117	EM5020: .ASCIZ /ERROR ATTEMPTING TO WRITE DATA SYNC (PREAMBLE)/
8833	045216	040440	052124	046505	
8834	045224	052120	047111	020107	
8835	045232	047524	053440	044522	
8836	045240	042524	042040	052101	
8837	045246	020101	054523	041516	
8838	045254	024040	051120	040505	
8839	045262	041115	042514	000051	
8840	045270	051105	047522	020122	EM5021: .ASCIZ /ERROR ATTEMPTING TO WRITE DATA OR ECC/
8841	045276	052101	042524	050115	
8842	045304	044524	043516	052040	
8843	045312	020117	051127	052111	
8844	045320	020105	040504	040524	
8845	045326	047440	020122	041505	
8846	045334	000103			
8847	045336	051105	047522	020122	E5021A: .ASCIZ /ERROR IN ECC GENERATION/
8848	045344	047111	042440	041503	
8849	045352	043440	047105	051105	
8850	045360	052101	047511	000116	
8851	045366	051105	047522	020122	EM5023: .ASCIZ /ERROR BETWEEN SECTORS ATTEMPTING MULTI-SECTOR WRITE/
8852	045374	042502	053524	042505	
8853	045402	020116	042523	052103	
8854	045410	051117	020123	052101	
8855	045416	042524	050115	044524	
8856	045424	043516	046440	046125	
8857	045432	044524	051455	041505	
8858	045440	047524	020122	051127	
8859	045446	052111	000105		
8860	045452	051115	020061	047111	EMW1: .ASCIZ /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
8861	045460	047503	051122	041505	
8862	045466	020124	047117	030440	
8863	045474	052123	052440	053520	
8864	045502	051101	020104	051124	
8865	045510	047101	044523	044524	
8866	045516	047117	047440	020106	
8867	045524	040515	047111	020124	
8868	045532	046103	041517	000113	
8869	045540	051115	020061	047111	EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
8870	045546	047503	051122	041505	
8871	045554	020124	047117	030440	
8872	045562	052123	042040	053517	
8873	045570	053516	051101	020104	
8874	045576	051124	047101	044523	
8875	045604	044524	047117	047440	
8876	045612	020106	040515	047111	
8877	045620	020124	046103	041517	
8878	045626	000113			
8879	045630	051115	020061	047111	EMW3: .ASCIZ /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/
8880	045636	047503	051122	041505	

8889	045644	020124	047117	031040
8890	045652	042116	052440	053520
8891	045660	051101	020104	051124
8892	045666	047101	044523	044524
8893	045674	047117	047440	020106
8894	045702	040515	047111	020124
8895	045710	046103	041517	000113
8896	045716	051115	020061	047111
8897	045724	047503	051122	041505
8898	045732	020124	047117	031040
8899	045740	042116	042040	053517
8900	045746	053516	051101	020104
8901	045754	051124	047101	044523
8902	045762	044524	047117	047440
8903	045770	020106	040515	047111
8904	045776	020124	046103	041517
8905	046004	000113		
8906	046006	054105	042520	052103
8907	046014	042105	041440	046131
8908	046022	047111	042504	020122
8909	046030	053117	051105	046106
8910	046036	053517	042440	051122
8911	046044	042040	042111	047040
8912	046052	052117	051440	052105
8913	046060	000		
8914	046061	127	044522	042524
8915	046066	052040	020117	054503
8916	046074	020114	031466	026062
8917	046102	052040	040522	045503
8918	046110	031040	020054	042523
8919	046116	052103	051117	031040
8920	046124	026065	053440	051117
8921	046132	020104	047503	047125
8922	046140	020124	030064	000061
8923	046146	047125	054105	042520
8924	046154	052103	042105	042440
8925	046162	051122	051117	053440
8926	046170	044510	042514	052040
8927	046176	054522	047111	020107
8928	046204	047524	043040	051117
8929	046212	042503	041440	046131
8930	046220	047440	042526	043122
8931	046226	047514	000127	
8932	046232	051127	052111	020105
8933	046240	047524	041440	046131
8934	046246	030440	032464	026062
8935	046254	052040	040522	045503
8936	046262	031040	020054	042523
8937	046270	052103	051117	031040
8938	046276	026065	053440	051117
8939	046304	020104	047503	047125
8940	046312	020124	030064	000061
8941	046320	047125	054105	042520
8942	046326	052103	042105	041440
8943	046334	046131	047111	042504
8944	046342	020122	053117	051105

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

EM5026: .ASCIZ EXPECTED CYLINDER OVERFLOW ERR DID NOT SET/

E5026A: .ASCIZ /WRITE TO CYL 632, TRACK 2, SECTOR 25, WORD COUNT 401/

EM5027: .ASCIZ /UNEXPECTED ERROR WHILE TRYING TO FORCE CYL OVERFLOW/

E5026B: .ASCIZ /WRITE TO CYL 1452, TRACK 2, SECTOR 25, WORD COUNT 401/

EM5030: .ASCIZ /UNEXPECTED CYLINDER OVERFLOW WRITING LAST PHYSICAL ADDRESS/

ERROR MESSAGES

0933	046350	046106	053517	053440
0938	046356	044522	044524	043516
0939	046364	046040	051501	020124
0940	046372	044120	051531	041511
0941	046400	046101	040440	042104
0942	046406	042522	051523	000
0943	046413	125	042516	050130
0944	046420	041505	042524	020104
0945	046426	051105	047522	020122
0946	046434	044127	046111	020105
0947	046442	051127	052111	047111
0948	046450	020107	040514	052123
0949	046456	050040	054510	044523
0950	046464	040503	020114	042101
0951	046472	051104	051505	000123
0952	046500	051105	047522	020122
0953	046506	052101	042524	050115
0954	046514	044524	043516	052040
0955	046522	020117	042522	042101
0956	046530	042040	052101	020101
0957	046536	054523	041516	024040
0958	046544	051120	040505	041115
0959	046552	042514	000051	
0960	046556	051105	047522	020122
0961	046564	052101	042524	050115
0962	046572	044524	043516	052040
0963	046600	020117	042522	042101
0964	046606	042040	052101	020101
0965	046614	051117	042440	041503
0966	046622	000		
0967	046623	105	041503	042440
0968	046630	051122	051117	044440
0969	046636	020116	042522	042101
0970	046644	042040	052101	020101
0971	046652	050117	051105	052101
0972	046660	047511	020116	
0973	046664	040504	040524	041440
0974	046672	046517	040520	042522
0975	046700	042440	051122	051117
0976	046706	040440	052106	051105
0977	046714	044440	050116	052125
0978	046722	047040	051120	052040
0979	046730	040522	051516	042506
0980	046736	000122		
0981	046740	041505	020103	042522
0982	046746	044507	052123	051105
0983	046754	047040	052117	041440
0984	046762	051117	042522	052103
0985	046770	040440	052106	051105
0986	046776	042440	041503	051040
0987	047004	040505	020104	047111
0988	047012	051040	040505	020104
0989	047020	040504	040524	000
0990	047025	105	051122	051117
0991	047032	044440	020116	041505
0992	047040	020103	040520	052124

EMS031: .ASCIZ /UNEXPECTED ERROR WHILE WRITING LAST PHYSICAL ADDRESS/

EMS032: .ASCIZ /ERROR ATTEMPTING TO READ DATA SYNC (PREAMBLE)/

EMS034: .ASCIZ /ERROR ATTEMPTING TO READ DATA OR ECC/

EMS035: .ASCIZ /ECC ERROR IN READ DATA OPERATION/

EMS037: .ASCIZ /DATA COMPARE ERROR AFTER INPUT NPR TRANSFER/

EMS041: .ASCIZ /ECC REGISTER NOT CORRECT AFTER ECC READ IN READ DATA/

EMS042: .ASCIZ /ERROR IN ECC PATTERN CALCULATION AFTER ECC ERROR/

8993	047046	051105	020116	040503
8994	047054	041514	046125	052101
8995	047062	047511	020116	043101
8996	047070	042524	020122	041505
8997	047076	020103	051105	047522
8998	047104	000122		
8999	047106	051105	047522	020122
9000	047114	047111	042440	041503
9001	047122	050040	051517	052111
9002	047130	047511	020116	047503
9003	047136	047125	044524	043516
9004	047144	040440	052106	051105
9005	047152	042440	041503	042440
9006	047160	051122	051117	000
9007	047165	105	051122	051117
9008	047172	040440	052106	051105
9009	047200	050040	047522	042503
9010	047206	051523	047111	020107
9011	047214	040504	040524	041440
9012	047222	042510	045503	042440
9013	047230	051122	051117	000
9014	047235	124	040522	051516
9015	047242	042506	020122	042514
9016	047250	043516	044124	042440
9017	047256	051122	051117	040440
9018	047264	052124	046505	052120
9019	047272	047111	020107	040520
9020	047300	052122	040511	020114
9021	047306	042523	052103	051117
9022	047314	047440	042520	040522
9023	047322	044524	047117	000
9024	047327	102	042101	051440
9025	047334	041505	047524	020122
9026	047342	051105	047522	020122
9027	047350	044504	020104	047516
9028	047356	020124	051120	053105
9029	047364	047105	020124	042504
9030	047372	044523	042522	020104
9031	047400	042101	051104	051505
9032	047406	020123	047111	051103
9033	047414	046505	047105	000124
9034	047422	040502	020104	042523
9035	047430	052103	051117	042440
9036	047436	051122	051117	042040
9037	047444	042111	047040	052117
9038	047452	041440	052501	042523
9039	047460	041440	051105	020122
9040	047466	047524	051440	052105
9041	047474	000		
9042	047475	102	042101	051440
9043	047502	041505	047524	020122
9044	047510	051105	047522	020122
9045	047516	040506	046111	042105
9046	047524	052040	020117	042523
9047	047532	020124	044127	047105
9048	047540	042440	050130	041505

EMS043: .ASCIZ /ERROR IN ECC POSITION COUNTING AFTER ECC ERROR/

EMS044: .ASCIZ /ERROR AFTER PROCESSING DATA CHECK ERROR/

EMS046: .ASCIZ /TRANSFER LENGTH ERROR ATTEMPTING PARTIAL SECTOR OPERATION/

EMS047: .ASCIZ /BAD SECTOR ERROR DID NOT PREVENT DESIRED ADDRESS INCREMENT/

EMS050: .ASCIZ /BAD SECTOR ERROR DID NOT CAUSE CERR TO SET/

EMS051: .ASCIZ /BAD SECTOR ERROR FAILED TO SET WHEN EXPECTED/

E14

RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A
DZR6EA.P11 28-SEP-76 15:31

MACY11 27(1006) 05-OCT-76 09:12 PAGE 173

SEQ 0173

ERROR MESSAGES

9049	047546	042524	000104		
9050	047552	040504	040524	052040	EM5052: .ASCII /DATA TRANSFER DID NOT CONTINUE TO END OF SECTOR/
9051	047560	040522	051516	042506	
9052	047566	020122	044504	020104	
9053	047574	047516	020124	047503	
9054	047602	052116	047111	042525	
9055	047610	052040	020117	047105	
9056	047616	020104	043117	051440	
9057	047624	041505	047524	122	
9058	047631	015	053412	042510	.ASCIZ <15><12>/WHEN BAD SECTOR ERROR OCCURRED/
9059	047636	020116	040502	020104	
9060	047644	042523	052103	051117	
9061	047652	042440	051122	051117	
9062	047660	047440	041503	051125	
9063	047666	042522	000104		
9064	047672	051105	047522	020122	EM5053: .ASCIZ /ERROR ATTEMPTING WRITE CHECK OR ECC READ AFTER WRT CHK/
9065	047700	052101	042524	050115	
9066	047706	044524	043516	053440	
9067	047714	044522	042524	041440	
9068	047722	042510	045503	047440	
9069	047730	020122	041505	020103	
9070	047736	042522	042101	040440	
9071	047744	052106	051105	053440	
9072	047752	052122	041440	045510	
9073	047760	000			
9074	047761	105	041503	042440	EM5054: .ASCIZ /ECC ERROR IN WRITE CHECK OPERATION/
9075	047766	051122	051117	044440	
9076	047774	020116	051127	052111	
9077	050002	020105	044103	041505	
9078	050010	020113	050117	051105	
9079	050016	052101	047511	000116	
9080	050024	051127	052111	020105	EM5060: .ASCIZ /WRITE CHECK ERROR FAILED TO SET IN 16 BIT MODE/
9081	050032	044103	041505	020113	
9082	050040	051105	047522	020122	
9083	050046	040506	046111	042105	
9084	050054	052040	020117	042523	
9085	050062	020124	047111	030440	
9086	050070	020066	044502	020124	
9087	050076	047515	042504	000	
9088	050103	124	040522	051516	EM5061: .ASCIZ /TRANSFER LENGTH ERROR WITH WRITE CHECK ERROR/
9089	050110	042506	020122	042514	
9090	050116	043516	044124	042440	
9091	050124	051122	051117	053440	
9092	050132	052111	020110	051127	
9093	050140	052111	020105	044103	
9094	050146	041505	020113	051105	
9095	050154	047522	000122		
9096	050160	051127	052111	020105	EM5063: .ASCIZ /WRITE CHECK ERROR FAILED TO SET IN 18 BIT MODE/
9097	050166	044103	041505	020113	
9098	050174	051105	047522	020122	
9099	050202	040506	046111	042105	
9100	050210	052040	020117	042523	
9101	050216	020124	047111	030440	
9102	050224	020070	044502	020124	
9103	050232	047515	042504	000	
9104					

9105	050240		.EVEN
9106			.SBTTL DATA PATTERNS AND BUFFERS
9107			:* PATTERN 1 - ALL ZEROS
9108			:* PATTERN 2 - ROTATING CELL PULSE PRECOMP
9109			
9110	050240		PAT2:
9111	050240	121105	121105
9112	050242	150442	150442
9113	050244	064221	064221
9114	050246	132110	132110
9115	050250	055044	055044
9116	050252	026422	026422
9117	050254	013211	013211
9118	050256	105504	105504
9119	050260	042642	042642
9120	050262	021321	021321
9121	050264	110550	110550
9122	050266	044264	044264
9123	050270	022132	022132
9124	050272	011055	011055
9125	050274	104426	104426
9126	050276	042213	042213
9127			
9128			:* PATTERN 3 - MAX PRECOMP PHASE MIX
9129			
9130	050300		PAT3:
9131	050300	133333	133333
9132	050302	066666	066666
9133	050304	155555	155555
9134	050306	155555	155555
9135	050310	133333	133333
9136	050312	066666	066666
9137	050314	066666	066666
9138	050316	155555	155555
9139	050320	155555	155555
9140	050322	133333	133333
9141	050324	133333	133333
9142	050326	133333	133333
9143	050330	133333	133333
9144	050332	133333	133333
9145	050334	133333	133333
9146	050336	133333	133333
9147			
9148			:* PATTERN 4 - HI-LO FREQ. MIX
9149			
9150	050340		PAT4:
9151	050340	177777	177777
9152	050342	177777	177777
9153	050344	177777	177777
9154	050346	052525	052525
9155	050350	052525	052525
9156	050352	052525	052525
9157	050354	177777	177777
9158	050356	177777	177777
9159	050360	052525	052525
9160	050362	052525	052525

9161	050364	177777	177777
9162	050366	052525	052525
9163	050370	177252	177252
9164	050372	177252	177252
9165	050374	172765	172765
9166	050376	172765	172765

;* PATTERN 5 - ALTERNATING 1'S AND 0'S

;* PATTERN 6 - COMPOSIT ROTATING

FAT6:

9172	050400		
9173	050400	072307	072307
9174	050402	135143	135143
9175	050404	156461	156461
9176	050406	167230	167230
9177	050410	073514	073514
9178	050412	035646	035646
9179	050414	016723	016723
9180	050416	107351	107351
9181	050420	143564	143564
9182	050422	061672	061672
9183	050424	030735	030735
9184	050426	114356	114356
9185	050430	046167	046167
9186	050432	123073	123073
9187	050434	151453	151453
9188	050436	164616	164616

;* PATTERN 7 - ALL ONES

9192	051440	000402	IBUFF: .BLKW 402
9193	051444	000402	OBUFF: .BLKW 402
9194		000001	.END

ABASE = 177440	1098#	1351	1392
ABORT = 042727	7633	8601#	
ACDW1 = 000000	1351		
ACDW2 = 000000	1351		
ACLO = 000010	1193#		
ACPUOP = 000000	1351	1366	
ADDW0 = 000000	1351		
ADDW1 = 000000	1351		
ADDW10 = 000000	1351		
ADDW11 = 000000	1351		
ADDW12 = 000000	1351		
ADDW13 = 000000	1351		
ADDW14 = 000000	1351		
ADDW15 = 000000	1351		
ADDW2 = 000000	1351		
ADDW3 = 000000	1351		
ADDW4 = 000000	1351		
ADDW5 = 000000	1351		
ADDW6 = 000000	1351		
ADDW7 = 000000	1351		
ADDW8 = 000000	1351		
ADDW9 = 000000	1351		
ADEVCT = 000000	1351	1357	
ADEVN = 000000	1351	1393	
AENV = 000000	1351	1362	
AENVN = 000000	1351	1363	
AFATAL = 000000	1351	1354	
AMADR1 = 000000	1351	1378	
AMADR2 = 000000	1351	1383	
AMADR3 = 000000	1351	1386	
AMADR4 = 000000	1351	1389	
AMAMS1 = 000000	1351	1373	
AMAMS2 = 000000	1351	1381	
AMAMS3 = 000000	1351	1384	
AMAMS4 = 000000	1351	1387	
AMSGAD = 000000	1351	1359	
AMSGLG = 000000	1351	1360	
AMSGTY = 000000	1351	1353	
AMTYP1 = 000000	1351	1374	
AMTYP2 = 000000	1351	1382	
AMTYP3 = 000000	1351	1385	
AMTYP4 = 000000	1351	1388	
APASS = 000000	1351	1356	
APRIOR = 000005	1097#	1351	
APTCSU = 000040	7508#	7689	
APTENV = 000001	7464	7506#	7540 7682
APTSIZ = 000200	1850	7505#	
APTSP0 = 000100	7466	7507#	7684
ASWREG = 000000	1351	1364	
ATESTN = 000000	1351	1355	
AUNIT = 000000	1351	1358	
AUSWR = 000000	1351	1365	
AVECT1 = 120210	1096#	1351	1390
AVECT2 = 000000	1351	1391	
BADPAR = 002334	1784#	7231*	
BAI = 000020	1156#		

BA16 = 000400	1141*													
BA17 = 001000	1142*													
BITCNT 002326	1781*	3372*	3389*	3502*	3519*	3635*	3652*	3766*	3783*	3897*	3914*	4028*	4045*	
	4160*	4177*	4483*	4500*	4614*	4631*	5076*	5093*	5900*	5925*	5938*	5953*	5996*	
	6055*	6188*	6193*	6195*	6208*	6232*	6242*	6253*	6290*	6322*	6365*	6391*	6466*	
	6471*	6473*	6486*	6510*	6521*	6533*	6545*	6632*	6661	6669*	6688	6714	6722*	
	6723	6726	6739	6744	8327	8330	8336	8339	8346	8349				
BIT0 = 000001	1042*	1138	1171	1190	3728	3858	3989	4575	5351	5448	6027	6032	6304	
	6353	6379	6646	6702	6772	7210								
BIT00 = 000001	1032*	1042												
BIT01 = 000002	1031*	1041												
BIT02 = 000004	1030*	1040												
BIT03 = 000010	1029*	1039												
BIT04 = 000020	1028*	1038												
BIT05 = 000040	1027*	1037												
BIT06 = 000100	1026*	1036												
BIT07 = 000200	1025*	1035												
BIT08 = 000400	1024*	1034	7364											
BIT09 = 001000	1023*	1033	7382	7551										
BIT1 = 000002	1041*	1172	3859	4576	5802	6843								
BIT10 = 002000	1022*	1143	1162	1181	1214	1229	6095	6867	6869	7528				
BIT11 = 004000	1021*	1144	1163	1182	1199	1215	1230	3725	3855	3986	4572	7389		
BIT12 = 010000	1020*	1146	1164	1183	1216	3726	3856	3987	3990	4573	6106			
BIT13 = 020000	1019*	1147	1165	1184	1200	1217	6863	6865	7535					
BIT14 = 040000	1018*	1148	1166	1185	1201	1218	3726	3856	3987	4573	6006	6343	6679	
	7350													
BIT15 = 100000	1017*	1149	1150	1167	1186	1202	1219	4837	6006	6076	7211			
BIT2 = 000004	1040*	1173	1192											
BIT3 = 000010	1039*	1155	1174	1193										
BIT4 = 000020	1038*	1156	1175	1194	1208	1223	6863	6865						
BIT5 = 000040	1037*	1157	1176	1195	1209	1224								
BIT6 = 000100	1036*	1139	1158	1177	1196	1210	1225	3329	3331	3333	3459	3461	3463	
	3592	3594	3596	3726	3856	3987	4117	4119	4121	4440	4442	4444	4573	
	5032	5034	5036	5652	5654	5656	6100							
BIT7 = 000200	1035*	1140	1159	1178	1197	1211	1226							
BIT8 = 000400	1034*	1141	1160	1179	1198	1212	1227	6104	6867	6869				
BIT9 = 001000	1033*	1142	1161	1180	1213	1228	2023	2141	2258	2375	2492	2609	2729	
	3725	3855	3986	4572	5830	5842	5851	5859	5867	5872	5915	5962	6010	
	6041	6097	6152	6214	6263	6417	6427	6492	6555	6804	6821			
BLDDAT 033612	1969	2087	2204	2321	2438	2555	2675	2790	2873	2957	3040	3127	3234	
	3321	3451	3584	3717	3847	3978	4109	4240	4345	4432	4564	4696	4791	
	4925	5018	5154	5245	5638	7068*								
BPTVEC= 000014	1049*													
BSE = 000200	1178*	4859	4861											
BSEDES 002352	1791*	1959*	2077*	2194*	2311*	2428*	2545*	2662*	2779*	2862*	2947*	3029*	3112*	
	3219*	3307*	3437*	3570*	3702*	3833*	3964*	4095*	4225*	4330*	4418*	4550*	4681*	
	4778*	4837*	4853*	4911*	5006*	5141*	5232*	5337*	5434*	5527*	5627*	6016	6018	
	6019													
CCLR = 100000	1150*	1957	2075	2192	2309	2426	2543	2660	2780	2863	2945	3030	3113	
	3220	3308	3438	3571	3704	3834	3965	4096	4226	4331	4419	4551	4682	
	4776	4935	5039	5164	5255	5356	5453	5541	5659					
CDT = 002000	1143*	2436	2553	2955	3038	6093	6131							
CERR = 100000	1149*	2833	2919	3000	3086	3409	3539	3672	3803	3934	4065	4197	4520	
	4651	4881	5113											
CFMT = 010000	1146*	2021	2139	2256	2373	2490	2607	2673	2727	4236	4341	4428	4560	
	5150	5535	5808	5828	6008	6090	6123	6328	6339	6636	6675			

T.WC	002162	1716#	5719	8341	8355													
LFE	= 000400	1160#																
UNLOAD	= 000007	1124#																
UNS	= 040000	1185#																
LPE	= 020000	1165#																
VV	= 000100	1196#																
WCE	= 040000	1166#	5399	5496	5589	5703												
WLE	= 004000	1182#																
WRDATA	= 000023	1130#	1967	2085	2202	2319	2436	2553	2673	2788	2871	2955	3038					
WRDCNT	002330	1782#																
WRHEAD	= 000027	1132#																
WRL	= 004000	1199#																
WRTBIT	032450	6240	6251	6314	6362	6389	6888#											
WRTCHK	= 000031	1133#	4921	5014	5150	5241	5346	5443	5535	5634								
WRTGAT	= 040000	1218#	6237	6903														
WR.PAR	= 000004	1094#																
\$APTHD	001000	1268	1274#															
\$ASTAT	= ***** U	7486	7501															
\$ATYC	035444	7457	7459#															
\$ATY1	035420	7455#																
\$ATY3	035426	7456#	7687															
\$ATY4	035436	7458#	7543															
\$AUTOB	001134	1305#	1871*	7980	8137													
\$BASE	001324	1392#	1882	1890*	1931	7645	8263											
\$BOADR	001122	1300#																
\$BODAT	001126	1302#																
\$BELL	001240	1343#	7530	7563	7933	8130												
\$CHARC	036722	7704*	7714*	7721	7730*	7735#												
\$CKSWR	037676	7966#	8308															
\$CNTAG	001100	1288#	1812	1813	1821	1827	1828	1829										
\$CM1	= 000010	1320#	1321#	1322#	1323#	1324#	1325#	1326#	1327#	1328#								
\$CM2	= 000020	1320#	1321#	1322#	1323#	1324#	1325#	1326#	1327#	1328#								
\$CM3	= 000010	1318#	1320															
\$CM4	= 000015	1328#	1329#	1330#	1331#	1332#	1333#	1334#	1335#	1336#	1337#	1338#	1339#	1340#				
		1341#																
\$CNTLC	040574	7921	8002	8130#														
\$CNTLG	040606	7987	8132#															
\$CNTLU	040601	8012	8105	8131#														
\$SCORE	034656	7286	7313#															
\$CPUOP	001276	1366#																
\$CRLF	001245	1345#	5760	7538	7563	7586	7591	7599	7612	7625	7703	7738	8023	8110				
		8130	8191															
\$CROUT	034706	7313	7320#															
\$DBLK	037370	7839	7873	7881#														
\$DEVCT	001260	1357#																
\$DEVH	001326	1393#																
\$DOAGN	025052	5742	5763	5769#														
\$DTBL	037360	7842	7877#															
\$ENDAD	025042	1251	1859	5765#	7558													
\$ENDCT	024706	1827	5744#															
\$ENULL	025056	5772#																
\$ENV	001270	1362#	1865	7464	7488	7540	7682											
\$ENVH	001271	1363#	1850	7466	7684	7689												
\$EOP	024652	5734#	7639	7654														
\$EOPCT	024700	1827*	5741#	5745	7637*	7653*												
\$ERFLG	001103	1291#	1804*	7339	7378	7380	7386*	7408	7448	7525*	7563	7648*						

\$MTYP4	001315	1388#													
\$MXCNT	035266	7397	7407#												
\$NULL	001154	1314#	7709	7738											
\$NWTST=	000001	1939#	1941	2057#	2059	2175#	2177	2292#	2294	2409#	2411	2526#	2528	2643#	
		2645	2763#	2765	2847#	2849	2929#	2931	3014#	3016	3097#	3099	3204#	3206	
		3289#	3291	3421#	3423	3552#	3554	3685#	3687	3816#	3818	3947#	3949	4078#	
		4080	4210#	4212	4315#	4317	4400#	4402	4533#	4535	4666#	4668	4759#	4761	
		4896#	4898	4988#	4990	5126#	5128	5216#	5218	5315#	5317	5412#	5414	5510#	
		5512	5610#	5612											
\$OCNT	037150	7770#	7799#	7812#											
\$OMODE	037152	7765#	7769#	7774	7777#	7788#	7814#								
\$OVER	035252	7351	7376	7385	7395	7404#									
\$PASS	001256	1356#	1849#	5738#	5739#	5750	5772	7391	7408						
\$PASTM	001006	1278#													
\$POWER	041220	8260	8267#												
\$PWACT	041214	8251#	8252#	8253#	8255#	8266#									
\$PWON	041072	1825	8242#	8257											
\$PWUP	041120	8243	8251#												
\$QUES	001244	1344#	7563	7738	8042	8113	8130	8188	8191						
\$ROCHR	040240	8055#	8309												
\$RODEC=	***** U	8312													
\$ROLIN	040330	8078#	8310												
\$RODOCT	040636	8152#	8311												
\$RDSZ =	000010	8071#													
\$REGAD	001160	1318#													
\$REG0	001162	1320#	3187#	4300#	5401#	5498#	5591#	5705#	8345	8357					
\$REG1	001164	1321#	3188#	4301#	5402#	5499#	5592#	5706#	8345	8357					
\$REG2	001166	1322#	3189#	4302#	5403#	5500#	5593#	5707#	8345	8357					
\$REG3	001170	1323#	5594#	8357											
\$REG4	001172	1324#	5595#	5597#	8357										
\$REG5	001174	1325#													
\$REG6	001176	1326#													
\$REG7	001200	1327#													
\$RESRE	041034	8224#	8313												
\$RTNAD	025054	5771#													
\$R2A =	***** U	8314													
\$SAVRE	040776	8208#	8312												
\$SCOPE	034756	1819	7348#												
\$SETUP=	000137	1796#	1818	1819	1821	1823	1825	1827	1828	1829	1831	1859	1862	5736	
		7349	7524	7550	7558	7925	7930	7931	7961	8137					
\$SIZE	034450	1929	7273#												
\$SIZEX	034712	7311	7321#												
\$STUP =	177777	1796#													
\$SVLAD	035216	7359	7398#												
\$SVPC =	000220	1249#	1254												
\$SWR =	167400	923#	935	939	940	941	942	943	944	945	946	1341	1342	1343	
		1828	1829	1831	1832	1955	2073	2190	2307	2424	2541	2658	2777	2860	
		2943	3027	3110	3217	3305	3436	3568	3700	3831	3962	4093	4223	4328	
		4416	4548	4679	4774	4909	5004	5139	5230	5335	5432	5525	5624	5731	
		5737	5764	5770	5772	7340	7341	7342	7343	7344	7350	7362	7364	7365	
		7378	7379	7380	7387	7388	7389	7401	7404	7407	7515	7516	7517	7518	
		7519	7528	7535	7547	7551	7563								
\$SWREG	001272	1364#	1852												
\$SWRMK=	000000	946	947	7344	7345	7368									
\$SWOBT	035270	5919	6045	6218	6431	6496	6826	7374	7408#						
\$TESTN	001254	1355#	5917	6043	6216	6429	6494	6824	7399#	7575#	7576#	8317	8318	8321	

M15

GETSWR	1057*	1862*													
LDLPER	1259*														
MSG	1939*	1941	2057*	2059	2175*	2177	2292*	2294	2409*	2411	2526*	2528	2643*	2645	2763*
	2765	2847*	2849	2929*	2921	3014*	3016	3097*	3099	3204*	3206	3289*	3291	3421*	3423
	3552*	3554	3685*	3687	3816*	3818	3947*	3949	4078*	4080	4210*	4212	4315*	4317	4400*
	4402	4533*	4535	4666*	4668	4759*	4761	4896*	4898	4988*	4990	5126*	5128	5216*	5218
	5315*	5317	5412*	5414	5510*	5512	5610*	5612							
MULT	1057*														
NEWST	1057*	1939	2057	2175	2292	2409	2526	2643	2763	2847	2929	3014	3097	3204	3289
	3421	3552	3685	3816	3947	4078	4210	4315	4400	4533	4666	4759	4896	4988	5126
	5216	5315	5412	5510	5610										
JSECRD	1765*	3115	3222	3310	3440	3573	3706	3836	3967	4098	4228	4333	4421	4553	4684
	4779														
OSECWC	1766*	4914	5008	5143	5234	5339	5436	5628							
POP	1057*	5876	5921	6048	6220	6433	6498	6810	7118	7187	7221	7498	7499	7868	8180
	8229														
PUSH	1057*	5799	5898	5991	6180	6285	6458	6614	7068	7169	7205	7459	7461	7482	7827
	8154	8209													
REPORT	1057*														
SCOPE	952*	1954	2072	2189	2306	2423	2540	2657	2776	2859	2942	3026	3109	3216	3304
	3435	3567	3699	3830	3961	4092	4222	4327	4415	4547	4678	4773	4908	5003	5138
	5229	5334	5431	5524	5623	5735									
SETPRI	1057*	8058													
SETTRA	8292*	8301	8302	8303	8304	8306	8308	8309	8310	8311	8312	8313	8314		
SETUP	1057*	1811													
SKIP	1057*	2034	2039	2044	2152	2157	2162	2269	2274	2279	2386	2391	2396	2503	2508
	2513	2620	2625	2630	2740	2745	2750	4863	4870	4885					
SLASH	1057*														
SPACE	1057*														
SSECHT	1765*	2782	2865	2949	3032										
STARS	1057*	1247	1261	1263	1270	1283	1347	1350	1939	1953	2057	2071	2175	2188	2292
	2305	2409	2422	2526	2539	2643	2656	2763	2775	2847	2858	2929	2941	3014	3025
	3097	3108	3204	3215	3289	3303	3421	3434	3552	3566	3685	3698	3816	3829	3947
	3960	4078	4091	4210	4221	4315	4326	4400	4414	4533	4546	4666	4677	4759	4772
	4896	4907	4988	5002	5126	5137	5216	5228	5315	5333	5412	5430	5510	5523	5610
	5622	5727	7259	7336	7443	7454	7511	7564	7573	7661	7740	7817	7884	7961	7976
	8047	9071	8140	8193	8239	8248	8271								
SWRSU	1057*	.833*													
TRMTRP	8292*														
TYPBIN	1057*														
TYPDEC	1057*	5750	5757												
TYPNAM	1057*	1855													
TYPNUM	1057*														
TYPOCS	1057*														
TYPOCT	1057*	1882	7989												
TYPTXT	1057*	5746	5753												
\$\$CMRE	1281*	1320	1321	1322	1323	1324	1325	1326	1327						
\$\$CHTM	1281*	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	
\$\$ESCA	1057*														
\$\$NEWT	1057*	1939	2057	2175	2292	2409	2526	2643	2763	2847	2929	3014	3097	3204	3289
	3421	3552	3685	3816	3947	4078	4210	4315	4400	4533	4666	4759	4896	4988	5126
	5216	5315	5412	5510	5610										
\$\$SET	8292*	8301	8302	8303	8304	8306	8308	8309	8310	8311	8312	8313	8314		
\$\$SETM	1849*														
\$\$SKIP	1057*	2034	2039	2044	2152	2157	2162	2269	2274	2279	2386	2391	2396	2503	2508
	2513	2620	2625	2630	2740	2745	2750	4863	4870	4885					

.EQUAT	923#	947
.HEADE	923#	925
.KT11	923#	1057
.SETUP	923#	1796
.SWRHI	923#	935
.SWRLO	923#	947#
.SACT1	923#	1245
.SAPT8	1348#	
.SAPTH	923#	1259
.SAPTY	923#	7452
.SCATC	923#	1231
.SCMTA	923#	1281
.SEOP	923#	5725
.SERRO	923#	7509
.SERRT	923#	
.SPOWE	923#	
.SRDOC	923#	8138
.SREAD	923#	7882
.SSAVE	923#	8191
.SSCOP	923#	7334
.SSIZE	923#	7257
.STRAP	923#	8269
.STYPD	923#	7815
.STYPE	923#	7659
.STYPO	923#	7738

. ABS. 052450 000

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

DSKZ:DZR6EA, DSKZ:DZR6EA.SEQ/SOL/DOC/CRF=DZR6EA.P11
 RUN-TIME: 86 81 8 SECONDS
 RUN-TIME RATIO: 246/177=1.3
 CORE USED: 31K (61 PAGES)

DOCUMENT PAGES: 195

