

RK611

DISKLSS CONTROL PART 3
CZR6CC0

AH-9106C-MC

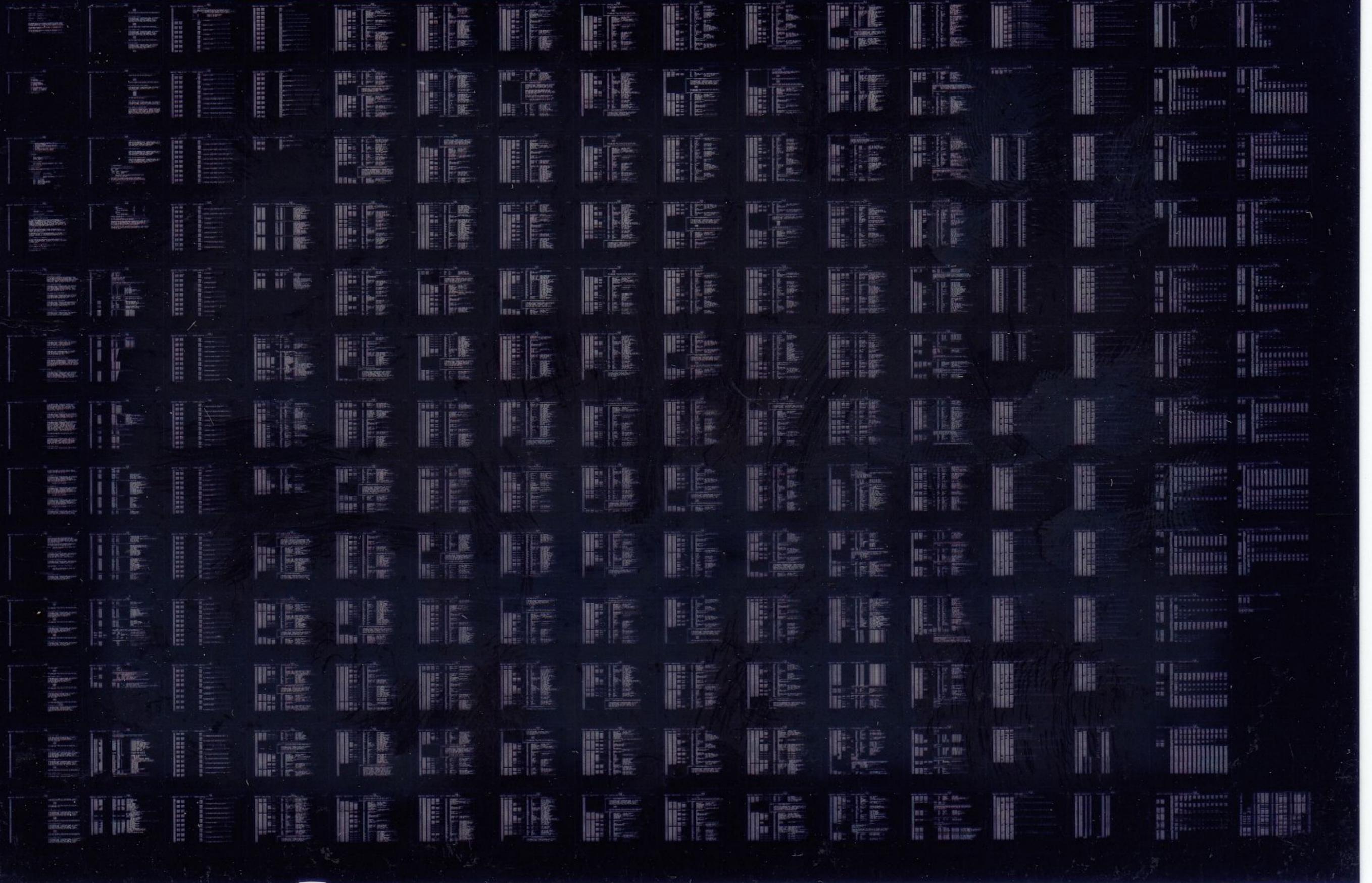
COPYRIGHT © 76-78

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA



56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111

1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3

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

NO RK06 DRIVE IS REQUIRED FOR PROGRAM EXECUTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

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

2.2 PRELIMINARY PROGRAMS

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

3.0 OPERATING PROCEDURES

3.1 LOADING PROCEDURE

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

3.2 STARTING PROCEDURE

- LOCATION 200 - START PROGRAM
- LOCATION 204 - RESTART PROGRAM
- LOCATION 214 - REQUEST BUS ADDRESS, VECTOR ADDRESS, AND PRIORITY MODIFICATION

3.3 OPTIONAL SWITCH SETTINGS

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

SWB - LOOP ON TEST IN SWITCHES 0-7

3.5 RUN TIME

FIRST PASS	30 SECONDS
SUBSEQUENT PASSES	8:40 MINUTES

4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPRIATE ADDRESS.

4.1 'SOFTWARE' SWITCH REGISTER

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

SWR = NNNNNN NEW ='

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

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

4.2 CONTROL C (↑C) OPERATION

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

4.3 CONTROL S (↑S) OPERATION

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

4.4 CONTROL Q (↑Q) OPERATION

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

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

5.0 PROGRAM DESCRIPTION

**DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

TEST 1 READ HEADER SEEK MESSAGE

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

TEST 2 WRITE HEADER SEEK MESSAGE

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

TEST 3 READ HEADER DRIVE CLEAR MESSAGE

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

TEST 4 WRITE HEADER DRIVE CLEAR MESSAGE

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

**INDEX AND SECTOR PULSE DETECT ON

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

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

168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223

MAKE SURE READ GATE DOES SET.

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

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

MAKE SURE READ GATE DOES NOT SET.

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

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

MAKE SURE READ GATE DOES NOT SET.

TEST 10 INDEX PULSE DETECTION IN WRITE HEADER

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

**NPR READING OF MEMORY

TEST 11 NPR OUTPUT DATA TRANSFER

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

TEST 12 PARTIAL SILO FILLING

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

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
276
277
278
279

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

TEST 13 SILO FILLING WITH NPR TRANSFERS

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

TEST 14 SILO CAPICITY WITH NPR TRANSFERS

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

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

TEST 15 BUS ADDRESS INHIBIT

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

TEST 16 NON-EXISTENT MEMORY

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06

280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335

IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS
(760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR
OCCURS IN THE RK611 CONTROLLER.

TEST 17 BUS ADDRESS BIT 16

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

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

TEST 20 BUS ADDRESS BIT 17

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

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

TEST 21 ADDRESSING GREATER THAN 96K

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

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

TEST 22 UNIBUS PARITY ERROR

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

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
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391

OCCURS.

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

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

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

TEST 23 SILO FILL IN 18 BIT MODE

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

TEST 24 BIT 16 AND 17 READING WITH NPR

CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY ONE WORD DATA TRANSFER FROM A LOCATION

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

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

**MFM READ LOOPBACK TESTS

TEST 25 READ LOOPBACK (PART 1)

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

392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447

FOLLOWING WORDS:

177777
000000
177777

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

TEST 26 READ LOOPBACK (PART 2)

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

000000
177777
000000

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

TEST 27 READ LOOPBACK (PART 3)

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

125252
052525
125252

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

TEST 30 READ LOOPBACK (PART 4)

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

448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503

504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
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

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

044444
022222
111111

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

TEST 31 READ LOOPBACK (PART 5)

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

052012
100520
052012

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

TEST 32 READ HEADER IN 18 BIT MODE

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

177777
000000
177777

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

TEST 33 SYNCH DETECT IN READ HEADER

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

TEST 34 ZERO SYNCH ON READ

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

177777
000000
177777

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

**MFM WRITE LOOPBACK TESTS

TEST 35 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER

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

TEST 36 WRITE LOOPBACK (PART 1)

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

177777
000000
177777

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

TEST 37 WRITE LOOPBACK (PART 2)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.

560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615

616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671

CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:

000000
177777
000000

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

TEST 40 WRITE LOOPBACK (PART 3)

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

125252
052525
125252

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

TEST 41 WRITE LOOPBACK (PART 4)

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

044444
022222
111111

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

TEST 42 WRITE LOOPBACK (PART 5)

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

672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727052012
100520
052012MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 43 WRITE LOOPBACK (PART 6)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:155555
066666
155555MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 44 WRITE LOOPBACK (PART 7)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:104210
104210
104210MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 45 WRITE TWO HEADERS

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTRLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
INDEX PULSE, SECTOR PULSE, THREE WORD HEADER
CONSISTING OF THE FOLLOWING DATA:177777
000000
177777

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

000000
177777
000000

SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP.
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMENSATION.

TEST 46 DATA FIELD FILLING ON WRITE HEADER

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND
SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE
FOLLOWING DATA:

125252
052525
125252
052525
125252
052525

MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
ECC FIELD ARE WRITTEN CORRECTLY.

TEST 47 WRITE HEADER FOR 26 SECTORS

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

TEST 50 WRITE HEADER IN 24 SECTOR FORMAT

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

**TYPE B INSTRUCTION ERRORS

728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783

784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
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

TEST 51 FORMAT ERROR (PART 1)

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

TEST 52 FORMAT ERROR (PART 2)

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

TEST 53 FAULT SETTING CONTROLLER ERROR

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

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION

TEST NUM	ERROR PC	OTHER PERTENANT INFORMATION	
XXXXXX	YYYYYY	EXPECT REG	ACTUAL REG
		ZZZZZZ	WWWWW
			AAAAA

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

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

THESE FIELDS (ALL VALUES GIVEN IN OCTAL) ARE:

FIELD	BITS	WORDS
HEADER PREAMBLE	400	20

840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869

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

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

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

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

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

%

870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925

167400
000001

001100

000011
000012
000015
000200
177776
177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007

```

; *** REV 003 ***
.NLIST  CND,MD,MC
.LIST   ME
.ENABL  ABS,AMA
$SWR=   167400
$TN=    1
.TITLE  CZR6CCO RK611 DSKLS CTRL PRT3
;*COPYRIGHT (C) 1976,1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY ROY SPITZER
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
.SBTTL  OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH          USE
;*      -----          -
;*      15             HALT ON ERROR
;*      14             LOOP ON TEST
;*      13             INHIBIT ERROR TYPEOUTS
;*      12             ABORT PROGRAM AFTER 20 ERRORS
;*      11             INHIBIT ITERATIONS
;*      10             BELL ON ERROR
;*      9              LOOP ON ERROR
;*      8              LOOP ON TEST IN SWR<7:0>
.SBTTL  BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK=  1100
.EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV  IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL
;*MISCELLANEOUS DEFINITIONS
HT=     11             ;;CODE FOR HORIZONTAL TAB
LF=     12             ;;CODE FOR LINE FEED
CR=     15             ;;CODE FOR CARRIAGE RETURN
CRLF=   200           ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS=     177776        ;;PROCESSOR STATUS WORD
.EQUIV  PS,PSW
STKLMT= 177774        ;;STACK LIMIT REGISTER
PIRQ=   177772        ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR=   177570        ;;HARDWARE SWITCH REGISTER
DDISP=  177570        ;;HARDWARE DISPLAY REGISTER
;*GENERAL PURPOSE REGISTER DEFINITIONS
R0=     %0            ;;GENERAL REGISTER
R1=     %1            ;;GENERAL REGISTER
R2=     %2            ;;GENERAL REGISTER
R3=     %3            ;;GENERAL REGISTER
R4=     %4            ;;GENERAL REGISTER
R5=     %5            ;;GENERAL REGISTER
R6=     %6            ;;GENERAL REGISTER
R7=     %7            ;;GENERAL REGISTER

```

```

926      000006      SP=      %6      ;;STACK POINTER
927      000007      PC=      %7      ;;PROGRAM COUNTER
928
929      ;*PRIORITY LEVEL DEFINITIONS
930      000000      PR0=      0      ;;PRIORITY LEVEL 0
931      000040      PR1=      40     ;;PRIORITY LEVEL 1
932      000100      PR2=      100    ;;PRIORITY LEVEL 2
933      000140      PR3=      140    ;;PRIORITY LEVEL 3
934      000200      PR4=      200    ;;PRIORITY LEVEL 4
935      000240      PR5=      240    ;;PRIORITY LEVEL 5
936      000300      PR6=      300    ;;PRIORITY LEVEL 6
937      000340      PR7=      340    ;;PRIORITY LEVEL 7
938
939      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
940      100000      SW15=     100000
941      040000      SW14=     40000
942      020000      SW13=     20000
943      010000      SW12=     10000
944      004000      SW11=     4000
945      002000      SW10=     2000
946      001000      SW09=     1000
947      000400      SW08=     400
948      000200      SW07=     200
949      000100      SW06=     100
950      000040      SW05=     40
951      000020      SW04=     20
952      000010      SW03=     10
953      000004      SW02=     4
954      000002      SW01=     2
955      000001      SW00=     1
956      .EQUIV     SW09,SW9
957      .EQUIV     SW08,SW8
958      .EQUIV     SW07,SW7
959      .EQUIV     SW06,SW6
960      .EQUIV     SW05,SW5
961      .EQUIV     SW04,SW4
962      .EQUIV     SW03,SW3
963      .EQUIV     SW02,SW2
964      .EQUIV     SW01,SW1
965      .EQUIV     SW00,SW0
966
967      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
968      100000      BIT15=    100000
969      040000      BIT14=    40000
970      020000      BIT13=    20000
971      010000      BIT12=    10000
972      004000      BIT11=    4000
973      002000      BIT10=    2000
974      001000      BIT09=    1000
975      000400      BIT08=    400
976      000200      BIT07=    200
977      000100      BIT06=    100
978      000040      BIT05=    40
979      000020      BIT04=    20
980      000010      BIT03=    10
981      000004      BIT02=    4

```

```

982          000002      BIT01= 2
983          000001      BIT00= 1
984          .EQUIV     BIT09,BIT9
985          .EQUIV     BIT08,BIT8
986          .EQUIV     BIT07,BIT7
987          .EQUIV     BIT06,BIT6
988          .EQUIV     BIT05,BIT5
989          .EQUIV     BIT04,BIT4
990          .EQUIV     BIT03,BIT3
991          .EQUIV     BIT02,BIT2
992          .EQUIV     BIT01,BIT1
993          .EQUIV     BIT00,BIT0
994
995          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
996          000004      ERRVEC= 4          ;: TIME OUT AND OTHER ERRORS
997          000010      RESVEC= 10         ;: RESERVED AND ILLEGAL INSTRUCTIONS
998          000014      TBITVEC=14        ;: "T" BIT
999          000014      TRTVEC= 14         ;: TRACE TRAP
1000         000014      BPTVEC= 14         ;: BREAKPOINT TRAP (BPT)
1001         000020      IOTVEC= 20         ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
1002         000024      PWRVEC= 24         ;: POWER FAIL
1003         000030      EMTVEC= 30         ;: EMULATOR TRAP (EMT) **ERROR**
1004         000034      TRAPVEC=34        ;: "TRAP" TRAP
1005         000060      TKVEC= 60          ;: TTY KEYBOARD VECTOR
1006         000064      TPVEC= 64          ;: TTY PRINTER VECTOR
1007         000240      PIRQVEC=240        ;: PROGRAM INTERRUPT REQUEST VECTOR
1008
1009         .SBTTL      MEMORY MANAGEMENT DEFINITIONS
1010
1011         ;*KT11 VECTOR ADDRESS
1012         000250      MMVEC= 250
1013
1014         ;*KT11 STATUS REGISTER ADDRESSES
1015
1016         177572      SR0= 177572
1017         177574      SR1= 177574
1018         177576      SR2= 177576
1019         172516      SR3= 172516
1020
1021         ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1022
1023         172300      KIPDR0= 172300
1024         172302      KIPDR1= 172302
1025         172304      KIPDR2= 172304
1026         172306      KIPDR3= 172306
1027         172310      KIPDR4= 172310
1028         172312      KIPDR5= 172312
1029         172314      KIPDR6= 172314
1030         172316      KIPDR7= 172316
1031
1032         ;*KERNEL "I" PAGE ADDRESS REGISTERS
1033
1034         172340      KIPAR0= 172340
1035         172342      KIPAR1= 172342
1036         172344      KIPAR2= 172344
1037         172346      KIPAR3= 172346

```

```

1038      172350      KIPAR4= 172350
1039      172352      KIPAR5= 172352
1040      172354      KIPAR6= 172354
1041      172356      KIPAR7= 172356
1042
1043      000114      MEMVEC= 114      ;MEMORY PARITY VECTOR
1044      172100      MEMBAS= 172100   ;MEM PARITY OPTION
1045      000004      WR.PAR= 4       ;WRITE BAD PARITY
1046      000001      PAR.EN= 1       ;ENABLE PARITY ENABLE
1047      120210      AVECT1= 120210  ;DEFINE RK611 VECTOR ADDRESS
1048      000005      APRIOR= 5       ;DEFINE RK611 PRIORITY
1049      177440      ABASE= 177440  ;DEFINE BASE OF RK611 REGISTERS
1050
1051      .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1052
1053      000000      RKCS1= 0        ;CONTROL AND STATUS REGISTER 1
1054      000002      RKWC= 2         ;WORD COUNT REGISTER
1055      000004      RKBA= 4         ;BUS ADDRESS REGISTER
1056      000006      RKDA= 6         ;DESIRED TRACK SECTOR REGISTER
1057      000010      RKCS2= 10       ;CONTROL AND STATUS REGISTER 2
1058      000012      RKDS= 12       ;DRIVE STATUS REGISTER
1059      000014      RKER= 14       ;ERROR REGISTER
1060      000016      RKASOF= 16      ;ATTENTION SUMMARY AND OFFSET REGISTER
1061      000020      RKDCYL= 20     ;DESIRED CYLINDER REGISTER
1062      000024      RKDB= 24       ;DATA BUFFER
1063      000026      RKMR1= 26     ;MAINTENANCE REGISTER 1
1064      000034      RKMR2= 34     ;MAINTENANCE REGISTER 2
1065      000036      RKMR3= 36     ;MAINTENANCE REGISTER 3
1066      000030      RKECPS= 30     ;ECC POSITION INFORMATION
1067      000032      RKECPT= 32     ;ECC PATTERN INFORMATION
1068      000022      RKSPAR= 22     ;SPARE REGISTER
1069
1070      .SBTTL  DRIVE COMMANDS
1071
1072      000001      SELDRV= 01     ;SELECT DRIVE
1073      000003      PACK= 03     ;PACK ACKNOWLEDGE
1074      000005      CLEAR= 05     ;DRIVE CLEAR
1075      000007      UNLOAD= 07    ;UNLOAD
1076      000011      SRTSPL= 11    ;START SPINDLE
1077      000013      RECAL= 13    ;RECALIBRATE
1078      000015      OFFSET= 15   ;OFFSET
1079      000017      SEEK= 17     ;SEEK
1080      000021      RDDATA= 21   ;READ DATA
1081      000023      WRDATA= 23   ;WRITE DATA
1082      000025      RDHEAD= 25   ;READ HEADER
1083      000027      WRHEAD= 27   ;WRITE HEADER AND DATA
1084      000031      WRTCHK= 31   ;WRITE CHECK
1085      000300      INTR= 300    ;GENERATE INTERRUPT TO CPU
1086
1087      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1088
1089      000001      GO= BIT0      ;GO BIT
1090      000100      IE= BIT6      ;INTERRUPT ENABLE
1091      000200      RDY= BIT7     ;CONTROLLER READY
1092      000400      BA16= BITE    ;BUS ADDRESS BIT 16
1093      001000      BA17= BITS    ;BUS ADDRESS BIT 17

```

1094	002000	CDT=	BIT10	; CONTROLLER DRIVE TYPE (0=RK06)
1095	004000	CTO=	BIT11	; CONTROLLER TIMED OUT WAITING FOR
1096				DRIVE RESPONSE
1097	010000	CFMT=	BIT12	; CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1098	020000	SPAR=	BIT13	; DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1099	040000	DI=	BIT14	; DRIVE INTERRUPT
1100	100000	CERR=	BIT15	; CONTROLLER ERROR
1101	100000	CCLR=	BIT15	; CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1105	000007	DRVMSK=	7	; MASK FOR DRIVE SELECTION CODE
1106	000010	RLS=	BIT3	; DESELECT OR RELEASE DRIVE IN BITS 0-2
1107	000020	BAI=	BIT4	; BUS ADDRESS INCREMENT INHIBIT
1108	000040	SCLR=	BIT5	; CLEAR CONTROLLER AND ALL DRIVES
1109	000100	IR=	BIT6	; INPUT READY
1110	000200	OR=	BIT7	; OUTPUT READY
1111	000400	UFE=	BIT8	; UNIT FIELD ERROR
1112	001000	MDS=	BIT9	; MULTIPLE DRIVE SELECT
1113	002000	PGE=	BIT10	; PROGRAMMING ERROR
1114	004000	NEM=	BIT11	; NON-EXISTENT MEMORY
1115	010000	NED=	BIT12	; NON-EXISTENT DRIVE
1116	020000	UPE=	BIT13	; UNIBUS PARITY ERROR
1117	040000	WCE=	BIT14	; WRITE CHECK ERROR
1118	100000	DLT=	BIT15	; DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1121	000001	ILF=	BIT0	; ILLEGAL FUNCTION CODE
1122	000002	SKI=	BIT1	; SEEK INCOMPLETE
1123	000004	NXF=	BIT2	; NON-EXECUTABLE DRIVE FUNCTION
1124	000010	DRPAR=	BIT3	; DRIVE DETECTED DRIVE BUS PARITY ERROR
1125	000020	FMTE=	BIT4	; FORMAT ERROR
1126	000040	DTYPE=	BIT5	; DRIVE TYPE ERROR
1127	000100	ECH=	BIT6	; ECC HARD
1128	000200	BSE=	BIT7	; BAD SECTOR ERROR
1129	000400	HVRC=	BIT8	; HEADER VRC ERROR
1130	001000	COE=	BIT9	; CYLINDER ADDRESS OVERFLOW ERROR
1131	002000	IDAE=	BIT10	; INVALID DISK ADDRESS ERROR
1132	004000	WLE=	BIT11	; WRITE LOCK ERROR
1133	010000	DTE=	BIT12	; DRIVE TIMING ERROR
1134	020000	OPI=	BIT13	; OPERATION (SEARCH) INCOMPLETE
1135	040000	UNS=	BIT14	; DRIVE UNSAFE
1136	100000	DCK=	BIT15	; DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1141	000001	DRA=	BIT0	; DRIVE AVAILABLE (CONTROLLER IS SET IF
1142				THIS BIT IS RESET)
1143	000004	OFST=	BIT2	; DRIVE OFFSET
1144	000010	ACLO=	BIT3	; AC LOW
1145	000020	SPDLSS=	BIT4	; SPEED LOSS
1146	000040	DROT=	BIT5	; DRIVE OFF TRACK
1147	000100	VV=	BIT6	; VOLUME VALID
1148	000200	DRDY=	BIT7	; DRIVE READY
1149	000400	DDT=	BIT8	; DRIVE TYPE (0=RK06)

```

1150          004000          WRL=      BIT11          ;WRITE LOCK
1151          020000          PIP=      BIT13          ;POSITIONING IN PROGRESS
1152          040000          DSC=      BIT14          ;DRIVE STATUS CHANGE
1153          100000          SVAL=     BIT15          ;STATUS VALID
1154
1155          .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1156
1157          000017          MESMSK= 17          ;MESSAGE MASK
1158
1159          000020          PAT=      BIT4           ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1160          000040          DMD=      BITS           ;DIAGNOSTIC MODE
1161          000100          MSP=      BIT6           ;MAINTENANCE SECTOR PULSE
1162          000200          MIND=     BIT7           ;MAINTENANCE INDEX
1163          000400          MCLK=     BIT8           ;MAINTENANCE CLOCK
1164          001000          MERD=     BIT9           ;MAINTENANCE ENCODED READ DATA
1165          002000          MEWD=     BIT10          ;MAINTENANCE ENCODED WRITE DATA
1166          004000          PCA=      BIT11          ;PRECOMPENSATION ADVANCE
1167          010000          PCD=      BIT12          ;PRECOMPENSATION DELAY
1168          020000          ECCW=     BIT13          ;ECC WORD IS BEING READ OR WRITTEN
1169          040000          WRTGAT=  BIT14          ;WRITE GATE
1170          100000          RDGATE=  BIT15          ;READ GATE
1171
1172          .SBTTL  TRANSMITTED MESSAGE A
1173
1174          000020          S. SEEK=  BIT4           ;SEEK COMMAND
1175          000040          S. RECL=  BITS           ;RECALIBRATE COMMAND
1176          000100          S. STSP=  BIT6           ;START SPINDLE COMMAND
1177          000200          S. RTC=   BIT7           ;DRIVE RETURN TO CENTERLINE COMMAND
1178          000400          S. CLR=   BIT8           ;CLEAR ERROR AND DSC
1179          001000          S. FMT=   BIT9           ;FORMAT
1180          002000          S. UNLD=  BIT10          ;UNLOAD
1181          004000          S. PACK=  BIT11          ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1182
1183          .SBTTL  TRAP CATCHER
1184
1185          000000          .=0
1186          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1187          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1188          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1189          000174          000000          DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
1190          000176          000000          SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
1191          .SBTTL  STARTING ADDRESS(ES)
1192          000200          000137          003662          JMP      @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1193          000204          000137          003652          JMP      RESTRT  ;;JUMP TO RESTART ROUTINE
1194          000214          000214          .=214
1195          000214          000137          003642          JMP      PARM    ;;JUMP TO OPERATOR ASSIGNED PARMETERS
1196          .SBTTL  ACT11 HOOKS
1197
1198          ;*****
1199          ;HOOKS REQUIRED BY ACT11
1200          000220          000220          $SVPC=.          ;SAVE PC
1201          000046          000046          .=46
1202          000046          044674          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1203          000052          000052          .=52
1204          000052          000000          .WORD 0          ;;2)SET LOC.52 TO ZERO
1205          000220          000220          .=$SVPC          ;; RESTORE PC

```

1206		000114
1207	000114	046274
1208	000116	000340
1209		001000
1210		
1211		
1212		
1213		
1214		
1215		001000
1216		000024
1217	000024	000200
1218		000044
1219	000044	001000
1220		001000
1221		
1222		
1223		
1224		
1225	001000	
1226	001000	000000
1227	001002	001214
1228	001004	000000
1229	001006	000000
1230	001010	000000
1231	001012	000032

```

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

;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.SX=.      ;SAVE CURRENT LOCATION
.=24      ;SET POWER FAIL TO POINT TO START OF PROGRAM
200       ;FOR APT START UP
.=44      ;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR   ;POINT TO APT HEADER BLOCK
.=.SX     ;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

$APTHD:
$HI8TS: .WORD 0      ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL  ;ADDRESS OF APT MAILBOX (BITS 0-15)
$STSM:  .WORD       ;RUN TIM OF LONGEST TEST
$PASTM: .WORD       ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD       ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
          .WORD $ETEND-$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)

```

1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287

001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000
001162 000000
001164 000000
001166 000000
001170 000000
001172 000000
001174 000000
001176 000000
001200 000000
001202 000000
001204 177607 000377
001210 077
001211 015
001212 000012

.SBTTL COMMON TAGS

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

SCMTAG: .=1100

;; START OF COMMON TAGS

.WORD 0
\$STNM: .BYTE 00
SERFLG: .BYTE 00
\$ICNT: .WORD 00
\$LPADR: .WORD 00
\$LPERR: .WORD 00
\$ERTTL: .WORD 00
\$ITEMB: .BYTE 00
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 00
\$GDADR: .WORD 00
\$BDADR: .WORD 00
\$GDDAT: .WORD 00
\$BDDAT: .WORD 00
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$TMP0: .WORD 0
\$TMP1: .WORD 0
\$TMP2: .WORD 0
\$TMP3: .WORD 0
\$TMP4: .WORD 0
\$TMP5: .WORD 0
\$TMP6: .WORD 0
\$TMP7: .WORD 0
\$TIMES: 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

CONTAINS THE TEST NUMBER
CONTAINS ERROR FLAG
CONTAINS SUBTEST ITERATION COUNT
CONTAINS SCOPE LOOP ADDRESS
CONTAINS SCOPE RETURN FOR ERRORS
CONTAINS TOTAL ERRORS DETECTED
CONTAINS ITEM CONTROL BYTE
CONTAINS MAX. ERRORS PER TEST
CONTAINS PC OF LAST ERROR INSTRUCTION
CONTAINS ADDRESS OF 'GOOD' DATA
CONTAINS ADDRESS OF 'BAD' DATA
CONTAINS 'GOOD' DATA
CONTAINS 'BAD' DATA
RESERVED--NOT TO BE USED
AUTOMATIC MODE INDICATOR
INTERRUPT MODE INDICATOR
ADDRESS OF SWITCH REGISTER
ADDRESS OF DISPLAY REGISTER
TTY KBD STATUS
TTY KBD BUFFER
TTY PRINTER STATUS REG. ADDRESS
TTY PRINTER BUFFER REG. ADDRESS
CONTAINS NULL CHARACTER FOR FILLS
CONTAINS # OF FILLER CHARACTERS REQUIRED
INSERT FILL CHARS. AFTER A "LINE FEED"
"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
USER DEFINED
MAX. NUMBER OF ITERATIONS
ESCAPE ON ERROR ADDRESS
CODE FOR BELL
QUESTION MARK
CARRIAGE RETURN
LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN

1288	001214		\$MAIL:		:: APT MAILBOX
1289	001214	000000	\$MSGTY: .WORD	AMSGTY	:: MESSAGE TYPE CODE
1290	001216	000000	\$FATAL: .WORD	AFATAL	:: FATAL ERROR NUMBER
1291	001220	000000	\$TESTN: .WORD	ATESTN	:: TEST NUMBER
1292	001222	000000	\$PASS: .WORD	APASS	:: PASS COUNT
1293	001224	000000	\$DEVCT: .WORD	ADEVCT	:: DEVICE COUNT
1294	001226	000000	\$UNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
1295	001230	000000	\$MSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
1296	001232	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
1297	001234		\$ETABLE:		:: APT ENVIRONMENT TABLE
1298	001234	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
1299	001235	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
1300	001236	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
1301	001240	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
1302	001242	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
1303			.*		BITS 15-11=CPU TYPE
1304			.*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1305			.*		11/70=06, PDQ=07, Q=10
1306			.*		BIT 10=REAL TIME CLOCK
1307			.*		BIT 9=FLOATING POINT PROCESSOR
1308			.*		BIT 8=MEMORY MANAGEMENT
1309	001244	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
1310	001245	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
1311			.*		MEM. TYPE BYTE -- (HIGH BYTE)
1312			.*		900 NSEC CORE=001
1313			.*		300 NSEC BIPOLAR=002
1314			.*		500 NSEC MOS=003
1315	001246	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
1316			.*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1317	001250	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
1318	001251	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
1319	001252	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
1320	001254	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
1321	001255	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
1322	001256	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
1323	001260	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
1324	001261	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
1325	001262	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
1326	001264	120210	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
1327	001266	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
1328	001270	177440	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
1329	001272	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
1330	001274	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
1331	001276	000000	\$CDW2: .WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
1332	001300		\$ETEND:		
1333			.MEXIT		

.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

1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348 001300
1349
1350
1351 001300 057755
1352 001302 064134
1353 001304 053214
1354 001306 053656
1355
1356
1357 001310 057755
1358 001312 064177
1359 001314 053214
1360 001316 053656
1361
1362
1363 001320 057755
1364 001322 064223
1365 001324 053214
1366 001326 053656
1367
1368
1369 001330 060037
1370 001332 064134
1371 001334 053214
1372 001336 053656
1373
1374
1375 001340 060037
1376 001342 064177
1377 001344 053214
1378 001346 053656
1379
1380
1381 001350 060037
1382 001352 064223
1383 001354 053214
1384 001356 053656
1385
1386
1387 001360 060122
1388 001362 064134
1389 001364 053214

\$ERRTB:
: ERROR 1: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
: CS1 INCORRECT.
: EM200
: EM3000
: DT001
: DF001
: ERROR 2: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
: MESSAGE A INCORRECT.
: EM200
: EM3001
: DT001
: DF001
: ERROR 3: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
: MESSAGE B INCORRECT.
: EM200
: EM3002
: DT001
: DF001
: ERROR 4: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
: CS1 INCORRECT.
: EM201
: EM3000
: DT001
: DF001
: ERROR 5: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
: MESSAGE A INCORRECT.
: EM201
: EM3001
: DT001
: DF001
: ERROR 6: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
: MESSAGE IS INCORRECT.
: EM201
: EM3002
: DT001
: DF001
: ERROR 7: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
: CS1 INCORRECT
: EM202
: EM3000
: DT001

1390	001366	053656	DF001	
1391			ERROR 10:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1392				MESSAGE A INCORRECT.
1393	001370	060122	EM202	
1394	001372	064177	EM3001	
1395	001374	053214	DT001	
1396	001376	053656	DF001	
1397			ERROR 11:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1398				MESSAGE B INCORRECT.
1399	001400	060122	EM202	
1400	001402	064223	EM3002	
1401	001404	053214	DT001	
1402	001406	053656	DF001	
1403			ERROR 12:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1404				CS1 INCORRECT.
1405	001410	060213	EM203	
1406	001412	064134	EM3000	
1407	001414	053214	DT001	
1408	001416	053656	DF001	
1409			ERROR 13:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1410				MESSAGE A INCORRECT.
1411	001420	060213	EM203	
1412	001422	064177	EM3001	
1413	001424	053214	DT001	
1414	001426	053656	DF001	
1415			ERROR 14:	ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1416				MESSAGE B INCORRECT.
1417	001430	060213	EM203	
1418	001432	064223	EM3002	
1419	001434	053214	DT001	
1420	001436	053656	DF001	
1421			ERROR 15:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1422				CS1 INCORRECT AFTER SENDING DRIVE CLEAR.
1423	001440	060305	EM204	
1424	001442	064247	EM3003	
1425	001444	053234	DT015	
1426	001446	053702	DF015	
1427			ERROR 16:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1428				CS1 INCORRECT AFTER DATE SIMULATION.
1429	001450	060305	EM204	
1430	001452	064317	EM3004	
1431	001454	053234	DT015	
1432	001456	053702	DF015	
1433			ERROR 17:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1434				MAINT REG. 1 INCORRECT DURING DATA SIMULATION (SECTOR PULSE)
1435	001460	060305	EM204	
1436	001462	064371	EM3005	
1437	001464	053244	DT017	
1438	001466	053726	DF017	
1439			ERROR 20:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1440				MAINT REG. 1 INCORRECT DURING DATA SIMULATION (INDEX PULSE)
1441	001470	060305	EM204	
1442	001472	064473	EM3006	
1443	001474	053244	DT017	
1444	001476	053726	DF017	
1445			ERROR 21:	ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT

1446					MAINT REG. 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)
1447	001500	060305	EM204		
1448	001502	064574	EM3007		
1449	001504	053244	DT017		
1450	001506	053726	DF017		
1451			ERROR 22:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1452				CSI INCORRECT AFTER SENDING DRIVE CLEAR	
1453	001510	060373	EM205		
1454	001512	064247	EM3003		
1455	001514	053262	DT022		
1456	001516	053752	DF022		
1457			ERROR 23:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1458				BUS ADD INCORRECT AFTER SENDING DRIVE CLEAR.	
1459	001520	060373	EM205		
1460	001522	064715	EM3008		
1461	001524	053262	DT022		
1462	001526	053752	DF022		
1463			ERROR 24:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1464				WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR	
1465	001530	060373	EM205		
1466	001532	064775	EM3009		
1467	001534	053262	DT022		
1468	001536	053752	DF022		
1469			ERROR 25:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1470				MAINT REG 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)	
1471	001540	060373	EM205		
1472	001542	065273	EM3014		
1473	001544	053302	DT025		
1474	001546	053776	DF025		
1475			ERROR 26:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1476				CSI CHANGED DURING COMMAND EXECUTION	
1477	001550	060373	EM205		
1478	001552	065054	EM3010		
1479	001554	053262	DT022		
1480	001556	053752	DF022		
1481			ERROR 27:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1482				BUS ADDRESS CHANGED BEFORE INDEX PULSE	
1483	001560	060373	EM205		
1484	001562	065121	EM3011		
1485	001564	053262	DT022		
1486	001566	053752	DF022		
1487			ERROR 30:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1488				WORD COUNT CHANGED BEFORE INDEX PULSE	
1489	001570	060373	EM205		
1490	001572	065170	EM3012		
1491	001574	053262	DT022		
1492	001576	053752	DF022		
1493			ERROR 31:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1494				CSI CHANGED AFTER INDEX PULSE	
1495	001600	060373	EM205		
1496	001602	065236	EM3013		
1497	001604	053314	DT031		
1498	001606	054022	DF031		
1499			ERROR 32:	ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1500				BUS ADDRESS CHANGED AFTER INDEX PULSE.	
1501	001610	060373	EM205		

1502	001612	065344	EM3015
1503	001614	053314	DT031
1504	001616	054022	DF031
1505			ERROR 33: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1506			WORD COUNT CHANGED AFTER INDEX PULSE
1507	001620	060373	EM205
1508	001622	065412	EM3016
1509	001624	053314	DT031
1510	001626	054022	DF031
1511			ERROR 34: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1512			MAINT REG 1 INCORRECT AFTER SECTOR PULSE
1513	001630	060373	EM205
1514	001632	065772	EM3025
1515	001634	053314	DT031
1516	001636	054022	DF031
1517			ERROR 35: ATTEMPTING AN NPR READ OF ONE WORD
1518			CS1 INCORRECT
1519	001640	060461	EM206
1520	001642	064134	EM3000
1521	001644	053340	DT035
1522	001646	054046	DF035
1523			ERROR 36: ATTEMPTING AN NPR READ OF ONE WORD
1524			CS2 INCORRECT
1525	001650	060461	EM206
1526	001652	065457	EM3018
1527	001654	053340	DT035
1528	001656	054046	DF035
1529			ERROR 37: ATTEMPTING AN NPR READ OF ONE WORD
1530			BUS ADDRESS INCORRECT
1531	001660	060461	EM206
1532	001662	065523	EM3019
1533	001664	053340	DT035
1534	001666	054046	DF035
1535			ERROR 40: ATTEMPTING AN NPR READ OF ONE WORD
1536			WORD COUNT REG INCORRECT
1537	001670	060461	EM206
1538	001672	065551	EM3020
1539	001674	053340	DT035
1540	001676	054046	DF035
1541			ERROR 41: ATTEMPTING AN NPR READ OF ONE WORD
1542			WORD READ INCORRECT
1543	001700	060461	EM206
1544	001702	065602	EM3021
1545	001704	053364	DT041
1546	001706	054072	DF041
1547			ERROR 42: ATTEMPTING AN NPR READ OF ONE WORD
1548			CS1 INCORRECT AFTER READING DATA BUFFER
1549	001710	060461	EM206
1550	001712	065626	EM3022
1551	001714	053374	DT042
1552	001716	054116	DF042
1553			ERROR 43: ATTEMPTING AN NPR READ OF ONE WORD
1554			CS2 INCORRECT AFTER READING DATA BUFFER
1555	001720	060461	EM206
1556	001722	065676	EM3023
1557	001724	053374	DT042

1558	001726	054116	DF042	
1559			ERROR 44:	ATTEMPTING AN NPR READ OF ONE WORD
1560				CS1 INCORRECT
1561	001730	060524	EM207	
1562	001732	064134	EM3000	
1563	001734	053340	DT035	
1564	001736	054046	DF035	
1565			ERROR 45:	ATTEMPTING AN NPR READ
1566				CS2 INCORRECT
1567	001740	060524	EM207	
1568	001742	065457	EM3018	
1569	001744	053340	DT035	
1570	001746	054046	DF035	
1571			ERROR 46:	ATTEMPTING AN NPR READ
1572				BUS ADDRESS INCORRECT
1573	001750	060524	EM207	
1574	001752	065523	EM3019	
1575	001754	053340	DT035	
1576	001756	054046	DF035	
1577			ERROR 47:	ATTEMPTING AN NPR READ
1578				WORD COUNT INCORRECT
1579	001760	060524	EM207	
1580	001762	065551	EM3020	
1581	001764	053340	DT035	
1582	001766	054046	DF035	
1583			ERROR 50:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1584				CS1 INCORRECT
1585	001770	060553	EM208	
1586	001772	064134	EM3000	
1587	001774	053340	DT035	
1588	001776	054046	DF035	
1589			ERROR 51:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1590				CS2 INCORRECT
1591	002000	060553	EM208	
1592	002002	065457	EM3018	
1593	002004	053340	DT035	
1594	002006	054046	DF035	
1595			ERROR 52:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1596				BUS ADDRESS INCORRECT
1597	002010	060553	EM208	
1598	002012	065523	EM3019	
1599	002014	053340	DT035	
1600	002016	054046	DF035	
1601			ERROR 53:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1602				WORD COUNT INCORRECT
1603	002020	060553	EM208	
1604	002022	065551	EM3020	
1605	002024	053340	DT035	
1606	002026	054046	DF035	
1607			ERROR 54:	ATTEMPTING NPR READ
1608				DATA BUFFER INCORRECT
1609	002030	060524	EM207	
1610	002032	065602	EM3021	
1611	002034	053410	DT054	
1612	002036	054142	DF054	
1613			ERROR 55:	ATTEMPTING NPR READ

1614			:		CS1 INCORRECT AFTER READING DATA BUFFER
1615	002040	060524	:	EM207	
1616	002042	065626	:	EM3022	
1617	002044	053422	:	DT055	
1618	002046	054166	:	DF055	
1619			:	ERROR 56:	ATTEMPTING NPR READ
1620			:		CS2 INCORRECT AFTER READING DATA BUFFER
1621	002050	060524	:	EM207	
1622	002052	065676	:	EM3023	
1623	002054	053422	:	DT055	
1624	002056	054166	:	DF055	
1625			:	ERROR 57:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1626			:		INHIBIT CS1 INCORRECT
1627	002060	060624	:	EM209	
1628	002062	064134	:	EM3000	
1629	002064	053340	:	DT035	
1630	002066	054046	:	DF035	
1631			:	ERROR 60:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1632			:		INHIBIT CS2 INCORRECT
1633	002070	060624	:	EM209	
1634	002072	065457	:	EM3018	
1635	002074	053340	:	DT035	
1636	002076	054046	:	DF035	
1637			:	ERROR 61:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1638			:		INHIBIT BUS ADDRESS INCORRECT
1639	002100	060624	:	EM209	
1640	002102	065523	:	EM3019	
1641	002104	053340	:	DT035	
1642	002106	054046	:	DF035	
1643			:	ERROR 62:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1644			:		INHIBIT WORD COUNT INCORRECT
1645	002110	060624	:	EM209	
1646	002112	065551	:	EM3020	
1647	002114	053340	:	DT035	
1648	002116	054046	:	DF035	
1649			:	ERROR 63:	ATTEMPTING NPR READ WITH IBA TO CHECK ZERO
1650			:		DETECT-CS1 INCORRECT
1651	002120	060713	:	EM210	
1652	002122	064134	:	EM3000	
1653	002124	053340	:	DT035	
1654	002126	054046	:	DF035	
1655			:	ERROR 64:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1656			:		DETECT-CS2 INCORRECT
1657	002130	060713	:	EM210	
1658	002132	065457	:	EM3018	
1659	002134	053340	:	DT035	
1660	002136	054046	:	DF035	
1661			:	ERROR 65:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1662			:		DETECT-BUS ADDRESS INCORRECT
1663	002140	060713	:	EM210	
1664	002142	065523	:	EM3019	
1665	002144	053340	:	DT035	
1666	002146	054046	:	DF035	
1667			:	ERROR 66:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1668			:		DETECT-WORD COUNT INCORRECT
1669	002150	060713	:	EM210	

1670	002152	065551	EM3020
1671	002154	053340	DT035
1672	002156	054046	DF035
1673			ERROR 67: ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1674			INCREMENT-DATA BUFFER INCORRECT
1675	002160	060624	EM209
1676	002162	065602	EM3021
1677	002164	053410	DT054
1678	002166	054142	DF054
1679			ERROR 70: ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1680			INCREMENT-CS1 INCORRECT AFTER READING DATA BUFFER
1681	002170	060624	EM209
1682	002172	065626	EM3022
1683	002174	053422	DT055
1684	002176	054166	DF055
1685			ERROR 71: ATTEMPTING NPR READ WITH ADDRESS BUFFER INHIBIT
1686			INCREMENT-CS2 INCORRECT AFTER READING DATA BUFFER
1687	002200	060624	EM209
1688	002202	065676	EM3023
1689	002204	053422	DT055
1690	002206	054166	DF055
1691			ERROR 72: ATTEMPTING TO FORCE NON-EXISTANT MEMORY
1692			CS1 INCORRECT
1693	002210	061027	EM211
1694	002212	064134	EM3000
1695	002214	053440	DT072
1696	002216	054212	DF072
1697			ERROR 73: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1698			CS2 INCORRECT
1699	002220	061027	EM211
1700	002222	065457	EM3018
1701	002224	053440	DT072
1702	002226	054212	DF072
1703			ERROR 74: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1704			ERROR REG INCORRECT
1705	002230	061027	EM211
1706	002232	065746	EM3024
1707	002234	053440	DT072
1708	002236	054212	DF072
1709			ERROR 75: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1710			BUS ADDRESS INCORRECT
1711	002240	061027	EM211
1712	002242	065523	EM3019
1713	002244	053440	DT072
1714	002246	054212	DF072
1715			ERROR 76: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1716			WORD COUNT INCORRECT
1717	002250	061027	EM211
1718	002252	065551	EM3020
1719	002254	053440	DT072
1720	002256	054212	DF072
1721			ERROR 77: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1722			CS1 INCORRECT
1723	002260	061077	EM212
1724	002262	064134	EM3000
1725	002264	053470	DT077

1726	002266	054246	DF077
1727			ERROR 100: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1728			CS2 INCORRECT
1729	002270	061077	EM212
1730	002272	065457	EM3018
1731	002274	053470	DT077
1732	002276	054246	DF077
1733			ERROR 101: TESTING EXTENDED MEMORY ADDRESSING BITS
1734			CS1 INCORRECT
1735	002300	061147	EM213
1736	002302	064134	EM3000
1737	002304	053340	DT035
1738	002306	054046	DF035
1739			ERROR 102: TESTING EXTENDED MEMORY ADDRESSING BITS
1740			CS2 INCORRECT
1741	002310	061147	EM213
1742	002312	065457	EM3018
1743	002314	053340	DT035
1744	002316	054046	DF035
1745			ERROR 103: TESTING EXTENDED MEMORY ADDRESSING BITS
1746			BUS ADDRESS INCORRECT
1747	002320	061147	EM213
1748	002322	065523	EM3019
1749	002324	053340	DT035
1750	002326	054046	DF035
1751			ERROR 104: TESTING EXTEND MEMORY ADDRESSING BITS
1752			WORD COUNT INCORRECT
1753	002330	061147	EM213
1754	002332	065551	EM3020
1755	002334	053340	DT035
1756	002336	054046	DF035
1757			ERROR 105: TESTING EXTENDED MEMORY ADDRESSING BITS
1758			DATA BUFFER INCORRECT
1759	002340	061147	EM213
1760	002342	065602	EM3021
1761	002344	053364	DT041
1762	002346	054072	DF041
1763			ERROR 106: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1764			CS1 INCORRECT
1765	002350	061217	EM214
1766	002352	064134	EM3000
1767	002354	053440	DT072
1768	002356	054212	DF072
1769			ERROR 107: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1770			CS2 INCORRECT
1771	002360	061217	EM214
1772	002362	065457	EM3018
1773	002364	053440	DT072
1774	002366	054212	DF072
1775			ERROR 110: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1776			BUS ADDRESS INCORRECT
1777	002370	061217	EM214
1778	002372	065523	EM3019
1779	002374	053440	DT072
1780	002376	054212	DF072
1781			ERROR 111: ATEMPTING TO FORCE UNIBUS PARITY ERROR

1782			:	WORD COUNT INCORRECT
1783	002400	061217	:	EM214
1784	002402	065551	:	EM3020
1785	002404	053440	:	DT072
1786	002406	054212	:	DF072
1787			;	ERROR 112: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1788			;	CS1 INCORRECT
1789	002410	061267	:	EM215
1790	002412	064134	:	EM3000
1791	002414	053340	:	DT035
1792	002416	054046	:	DF035
1793			;	ERROR 113: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1794			;	CS2 INCORRECT
1795	002420	061267	:	EM215
1796	002422	065457	:	EM3018
1797	002424	053340	:	DT035
1798	002426	054046	:	DF035
1799			;	ERROR 114: ATTEMPTING NPR READ OR LOCATION PRIOR TO BAD PARITY
1800			;	BUS ADDRESS INCORRECT
1801	002430	061267	:	EM215
1802	002432	065523	:	EM3019
1803	002434	053340	:	DT035
1804	002436	054046	:	DF035
1805			;	ERROR 115: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1806			;	WORD COUNT INCORRECT
1807	002440	061267	:	EM215
1808	002442	065457	:	EM3018
1809	002444	053340	:	DT035
1810	002446	054046	:	DF035
1811			;	ERROR 116: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1812			;	ERROR REG INCORRECT
1813	002450	061217	:	EM214
1814	002452	065746	:	EM3024
1815	002454	053440	:	DT072
1816	002456	054212	:	DF072
1817			;	ERROR 117: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1818			;	CS1 INCORRECT
1819	002460	061353	:	EM216
1820	002462	064134	:	EM3000
1821	002464	053470	:	DT077
1822	002466	054246	:	DF077
1823			;	ERROR 120: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1824			;	CS2 INCORRECT
1825	002470	061353	:	EM216
1826	002472	065457	:	EM3018
1827	002474	053470	:	DT077
1828	002476	054246	:	DF077
1829			;	ERROR 121: ATTEMPTING 18 BIT NPR READ
1830			;	CS1 INCORRECT
1831	002500	061423	:	EM217
1832	002502	064134	:	EM3000
1833	002504	053340	:	DT035
1834	002506	054046	:	DF035
1835			;	ERROR 122: ATTEMPTING 18 BIT NPR READ
1836			;	CS2 INCORRECT
1837	002510	061423	:	EM217

1838	002512	065457	EM3018
1839	002514	053340	DT035
1840	002516	054046	DF035
1841			ERROR 123: ATTEMPTING 18 BIT NPR READ
1842			BUS ADDRESS INCORRECT
1843	002520	061423	EM217
1844	002522	065523	EM3019
1845	002524	053340	DT035
1846	002526	054046	DF035
1847			ERROR 124: ATTEMPTING 18 BIT NPR READ
1848			WORD COUNT INCORRECT
1849	002530	061423	EM217
1850	002532	065551	EM3020
1851	002534	053340	DT035
1852	002536	054046	DF035
1853			ERROR 125: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1854			CS1 INCORRECT
1855	002540	061456	EM218
1856	002542	064134	EM3000
1857	002544	053340	DT035
1858	002546	054046	DF035
1859			ERROR 126: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1860			CS2 INCORRECT
1861	002550	061456	EM218
1862	002552	065457	EM3018
1863	002554	053340	DT035
1864	002556	054046	DF035
1865			ERROR 127: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1866			BUS ADDRESS INCORRECT
1867	002560	061456	EM218
1868	002562	065523	EM3019
1869	002564	053340	DT035
1870	002566	054046	DF035
1871			ERROR 130: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1872			WORD COUNT INCORRECT
1873	002570	061456	EM218
1874	002572	065551	EM3020
1875	002574	053340	DT035
1876	002576	054046	DF035
1877			ERROR 131: ATTEMPTING 18 BIT NPR READ
1878			DATA BUFFER INCORRECT
1879	002600	061423	EM217
1880	002602	065602	EM3021
1881	002604	053374	DT042
1882	002606	054116	DF042
1883			ERROR 132: ATTEMPTING 18 BIT NPR READ
1884			CS1 INCORRECT AFTER READING DATA BUFFER
1885	002610	061423	EM217
1886	002612	065626	EM3022
1887	002614	053422	DT055
1888	002616	054166	DF055
1889			ERROR 133: ATTEMPTING 18 BIT NPR READ
1890			CS2 INCORRECT AFTER READING DATA BUFFER
1891	002620	061423	EM217
1892	002622	065676	EM3023
1893	002624	053422	DT055

1894	002626	054166	DF055	
1895			ERROR 134: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1896			CS1 INCORRECT	
1897	002630	061536	EM219	
1898	002632	064134	EM3000	
1899	002634	053340	DT035	
1900	002636	054046	DF035	
1901			ERROR 135: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1902			CS2 INCORRECT	
1903	002640	061536	EM219	
1904	002642	065457	EM3018	
1905	002644	053340	DT035	
1906	002646	054046	DF035	
1907			ERROR 136: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1908			BUS ADDRESS INCORRECT	
1909	002650	061536	EM219	
1910	002652	065523	EM3019	
1911	002654	053340	DT035	
1912	002656	054046	DF035	
1913			ERROR 137: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1914			WORD COUNT INCORRECT	
1915	002660	061536	EM219	
1916	002662	065551	EM3020	
1917	002664	053340	DT035	
1918	002666	054046	DF035	
1919			ERROR 140: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1920			CHECKING ZERO DETECT	
1921			CS1 INCORRECT	
1922	002670	061616	EM220	
1923	002672	064134	EM3000	
1924	002674	053340	DT035	
1925	002676	054046	DF035	
1926			ERROR 141: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1927			CHECKING ZERO DETECT	
1928			CS2 INCORRECT	
1929	002700	061616	EM220	
1930	002702	065457	EM3018	
1931	002704	053340	DT035	
1932	002706	054046	DF035	
1933			ERROR 142: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1934			CHECKING ZERO DETECT	
1935			BUS ADDRESS INCORRECT	
1936	002710	061616	EM220	
1937	002712	065523	EM3019	
1938	002714	053340	DT035	
1939	002716	054046	DF035	
1940			ERROR 143: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1941			CHECKING ZERO DETECT	
1942			WORD COUNT INCORRECT	
1943	002720	061616	EM220	
1944	002722	065551	EM3020	
1945	002724	053340	DT035	
1946	002726	054046	DF035	
1947			ERROR 144: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET	
1948			DATA BUFFER INCORRECT	
1949	002730	061536	EM219	

ERROR POINTER TABLE

1950	002732	065602	EM3021
1951	002734	053364	DT041
1952	002736	054072	DF041
1953			ERROR 145: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1954			CS1 INCORRECT AFTER READING DATA BUFFER
1955	002740	061536	EM219
1956	002742	065626	EM3022
1957	002744	053470	DT077
1958	002746	054246	DF077
1959			ERROR 146: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1960			CS2 INCORRECT AFTER READING DATA BUFFER
1961	002750	061536	EM219
1962	002752	065676	EM3023
1963	002754	053470	DT077
1964	002756	054246	DF077
1965			ERROR 147: UNEXPECTED MEMORY PARITY ENABLE TRAP
1966	002760	057710	EM000
1967	002762	055070	DH000C
1968	002764	053210	DT000
1969	002766	053652	DF000
1970			ERROR 150: ATTEMPTING SIMULATION OF DATA IN READ HEADER
1971			CS1 INCORRECT
1972	002770	061675	EM221
1973	002772	064134	EM3000
1974	002774	053504	DT150
1975	002776	054272	DF150
1976			ERROR 151: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1977			CS1 INCORRECT AFTER COMPLETION OF COMMAND
1978	003000	061752	EM222
1979	003002	066042	EM3026
1980	003004	053374	DT042
1981	003006	054116	DF042
1982			ERROR 152: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1983			CS2 INCORRECT AFTER COMPLETION OF COMMAND
1984	003010	061752	EM222
1985	003012	066111	EM3027
1986	003014	053374	DT042
1987	003016	054116	DF042
1988			ERROR 153: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1989			CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1990	003020	062025	EM223
1991	003022	065626	EM3022
1992	003024	053422	DT055
1993	003026	054166	DF055
1994			ERROR 154: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1995			CS2 INCORRECT AFTER UNLOADING DATA
1996	003030	062025	EM223
1997	003032	065676	EM3023
1998	003034	053422	DT055
1999	003036	054166	DF055
2000			ERROR 155: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
2001			DATA READ INCORRECT
2002	003040	062025	EM223
2003	003042	065602	EM3021
2004	003044	053410	DT054
2005	003046	054142	DF054

2006			:	ERROR 156: ATTEMPTING SIMULATION OF DATA IN READ HEADER (20 BIT FORMAT)
2007			:	CS1 INCORRECT
2008	003050	062103	:	EM224
2009	003052	064134	:	EM3000
2010	003054	053504	:	DT150
2011	003056	054272	:	DF150
2012			:	ERROR 157: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE/
2013			:	CS1 INCORRECT AFTER COMMAND COMPLETION
2014	003060	062200	:	EM225
2015	003062	066042	:	EM3026
2016	003064	053374	:	DT042
2017	003066	054116	:	DF042
2018			:	ERROR 160: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE
2019			:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
2020	003070	062200	:	EM225
2021	003072	066111	:	EM3027
2022	003074	053374	:	DT042
2023	003076	054116	:	DF042
2024			:	ERROR 161: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2025			:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
2026	003100	062265	:	EM226
2027	003102	065626	:	EM3022
2028	003104	053422	:	DT055
2029	003106	054166	:	DF055
2030			:	ERROR 162: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2031			:	CS2 INCORRECT AFTER UNLOADING DATA BUFFER
2032	003110	062265	:	EM226
2033	003112	065676	:	EM3023
2034	003114	053422	:	DT055
2035	003116	054166	:	DF055
2036			:	ERROR 163: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
2037			:	DATA BUFFER INCORRECT
2038	003120	062265	:	EM226
2039	003122	065602	:	EM3021
2040	003124	053410	:	DT054
2041	003126	054142	:	DF054
2042			:	ERROR 164: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2043			:	CS1 INCORRECT
2044	003130	062365	:	EM227
2045	003132	064134	:	EM3000
2046	003134	053520	:	DT164
2047	003136	054316	:	DF164
2048			:	ERROR 165: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2049			:	CS2 INCORRECT
2050	003140	062365	:	EM227
2051	003142	065457	:	EM3018
2052	003144	053520	:	DT164
2053	003146	054316	:	DF164
2054			:	ERROR 166: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2055			:	CS1 INCORRECT READING EMPTY SILO
2056	003150	062365	:	EM227
2057	003152	066160	:	EM3028
2058	003154	053520	:	DT164
2059	003156	054316	:	DF164
2060			:	ERROR 167: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2061			:	CS1 INCORRECT READING EMPTY SILO

2062	003160	062365	EM227
2063	003162	066227	EM3029
2064	003164	053520	DT164
2065	003166	054316	DF164
2066			ERROR 170: WRITE BIT ERRORS
2067	003170	000000	0
2068	003172	000000	0
2069	003174	053536	DT170
2070	003176	054342	DF170
2071			ERROR 171: WRITE GATE NOT RESET WITH SECTOR PULSE
2072	003200	062601	EM231
2073	003202	066276	EM3030
2074	003204	053562	DT171
2075	003206	054366	DF171
2076			ERROR 172: WRITE GATE NOT SET WITH SECTOR PULSE RESET
2077	003210	062723	EM232
2078	003212	066276	EM3030
2079	003214	053562	DT171
2080	003216	054366	DF171
2081			ERROR 173: WRITE GATE NOT RESET WITH SECOND INDEX PULSE
2082	003220	063151	EM235
2083	003222	066276	EM3030
2084	003224	053562	DT171
2085	003226	054366	DF171
2086			ERROR 174: CS1 INCORRECT AT END OF WRITE HEADER
2087	003230	063261	EM236
2088	003232	064134	EM3000
2089	003234	053234	DT015
2090	003236	053702	DF015
2091			ERROR 175: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2092			CS1 INCORRECT
2093	003240	063631	EM241
2094	003242	064134	EM3000
2095	003244	053572	DT175
2096	003246	054412	DF175
2097			ERROR 176: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2098			CS2 INCORRECT
2099	003250	063631	EM241
2100	003252	065457	EM3018
2101	003254	053572	DT175
2102	003256	054412	DF175
2103			ERROR 177: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2104			DRIVE STATUS REG INCORRECT
2105	003260	063631	EM241
2106	003262	066324	EM3031
2107	003264	053572	DT175
2108	003266	054412	DF175
2109			ERROR 200: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2110			ERROR REG INCORRECT
2111	003270	063631	EM241
2112	003272	066357	EM3032
2113	003274	053572	DT175
2114	003276	054412	DF175
2115			ERROR 201: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2116			CS1 INCORRECT
2117	003300	063715	EM242

2118	003302	064134	EM3000
2119	003304	053572	DT175
2120	003306	054412	DF175
2121			ERROR 202: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2122			CS2 INCORRECT
2123	003310	063715	EM242
2124	003312	065457	EM3018
2125	003314	053572	DT175
2126	003316	054412	DF175
2127			ERROR 203: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2128			DRIVE STATUS REG INCORRECT
2129	003320	063715	EM242
2130	003322	066324	EM3031
2131	003324	053572	DT175
2132	003326	054412	DF175
2133			ERROR 204: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2134			ERROR REGISTER INCORRECT
2135	003330	063715	EM242
2136	003332	066357	EM3032
2137	003334	053572	DT175
2138	003336	054412	DF175
2139			ERROR 205: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2140			CS1 INCORRECT
2141	003340	064001	EM243
2142	003342	064134	EM3000
2143	003344	053572	DT175
2144	003346	054412	DF175
2145			ERROR 206: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2146			CS2 INCORRECT
2147	003350	064001	EM243
2148	003352	065457	EM3018
2149	003354	053572	DT175
2150	003356	054412	DF175
2151			ERROR 207: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2152			DRIVE STATUS REG INCORRECT
2153	003360	064001	EM243
2154	003362	066324	EM3031
2155	003364	053572	DT175
2156	003366	054412	DF175
2157			ERROR 210: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2158			ERROR REG INCORRECT
2159	003370	064001	EM243
2160	003372	066357	EM3032
2161	003374	053572	DT175
2162	003376	054412	DF175
2163			ERROR 211: ATTEMPTING TO CLEAR CONTROLLER ERROR
2164			CS1 INCORRECT
2165	003400	064102	EM244
2166	003402	064134	EM3000
2167	003404	053616	DT211
2168	003406	054436	DF211
2169			ERROR 212: ATTEMPTING TO CLEAR CONTROLLER ERROR
2170			CS2 INCORRECT
2171	003410	064102	EM244
2172	003412	065457	EM3018
2173	003414	053616	DT211

2174	003416	054436		DF211
2175			:	ERROR 213: ATTEMPTING TO CLEAR CONTROLLER
2176			:	DRIVE STATUS REG INCORRECT
2177	003420	064102		EM244
2178	003422	066324		EM3031
2179	003424	053616		DT211
2180	003426	054436		DF211
2181			:	ERROR 214: ATTEMPTINT TO CLEAR CONTROLLER
2182			:	ERROR REG INCORRECT
2183	003430	064102		EM244
2184	003432	066357		EM3032
2185	003434	053616		DT211
2186	003436	054436		DF211

```

2187          .SBTTL  TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2188
2189 003440 000000 T.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
2190 003442 000000 T.WC: .WORD 0 ;WORD COUNT REGISTER
2191 003444 000000 T.BA: .WORD 0 ;BUS ADDRESS REGISTER
2192 003446 000000 T.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2193 003450 000000 T.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2194 003452 000000 T.DS: .WORD 0 ;DRIVE STATUS REGISTER
2195 003454 000000 T.ER: .WORD 0 ;ERROR REGISTER
2196 003456 000000 T.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2197 003460 000000 T.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2198 003462 000000 T.DB: .WORD 0 ;DATA BUFFER
2199 003464 000000 T.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2200 003466 000000 T.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2201 003470 000000 T.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2202 003472 000000 T.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2203 003474 000000 T.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2204 003476 000000 T.SPAR: .WORD 0 ;SPARE REGISTER
2205
2206          .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS
2207
2208 003500 000000 E.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
2209 003502 000000 E.WC: .WORD 0 ;WORD COUNT REGISTER
2210 003504 000000 E.BA: .WORD 0 ;BUS ADDRESS REGISTER
2211 003506 000000 E.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2212 003510 000000 E.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2213 003512 000000 E.DS: .WORD 0 ;DRIVE STATUS REGISTER
2214 003514 000000 E.ER: .WORD 0 ;ERROR REGISTER
2215 003516 000000 E.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2216 003520 000000 E.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2217 003522 000000 E.DB: .WORD 0 ;DATA BUFFER
2218 003524 000000 E.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2219 003526 000000 E.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2220 003530 000000 E.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2221 003532 000000 E.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2222 003534 000000 E.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2223 003536 000000 E.SPAR: .WORD 0 ;SPARE REGISTER
2224
2225          .SBTTL  PREVIOUS RK611 CONTROLLER REGISTERS
2226
2227 003540 000000 P.CS1: .WORD 0 ;CONTROL AND STATUS REGISTER 1
2228 003542 000000 P.WC: .WORD 0 ;WORD COUNT REGISTER
2229 003544 000000 P.BA: .WORD 0 ;BUS ADDRESS REGISTER
2230 003546 000000 P.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2231 003550 000000 P.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2232 003552 000000 P.DS: .WORD 0 ;DRIVE STATUS REGISTER
2233 003554 000000 P.ER: .WORD 0 ;ERROR REGISTER
2234 003556 000000 P.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2235 003560 000000 P.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2236 003562 000000 P.DB: .WORD 0 ;DATA BUFFER
2237 003564 000000 P.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2238 003566 000000 P.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2239 003570 000000 P.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2240 003572 000000 P.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2241 003574 000000 P.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2242 003576 000000 P.SPAR: .WORD 0 ;SPARE REGISTER

```

```

2243      .SBTTL  PROGRAM DEFINED VARIABLES
2244
2245 003600 000210      RKVEC:  .WORD  210      ;RK611 VECTOR
2246 003602 000240      RKPRI:  .WORD  PR5      ;RK611 PRIORITY
2247 003604 000000      TRAPPC: .WORD  0      ;PC FOR MEMORY CHECK ENABLE TRAP
2248 003606 000000      SRTFLG: .WORD  0      ;START FLAG
2249                                     ; 0 = 200
2250                                     ; 1 = 214
2251                                     ; -1 = 204
2252 003610 000000      ERRCNT:  .WORD  0      ;ERROR COUNT FOR SWITCH 12 ABORT
2253 003612 000000      P1.BIT:  .WORD  0      ;NEXT BIT IN DATA SIMULATION
2254 003614 000000      PR.BIT:  .WORD  0      ;PRESENT BIT IN DATA SIMULATION
2255 003616 000000      M1.BIT:  .WORD  0      ;PREVIOUS BIT IN DATA SIMULATION
2256 003620 000000      M2.BIT:  .WORD  0      ;BIT BEFORE PREVIOUS BIT
2257 003622 000000      BITCNT: .WORD  0      ;BIT POSITION
2258 003624 000000      WRDCNT: .WORD  0      ;WORD COUNT FOR NPR TRANSFER
2259 003626 000000      SECCNT: .WORD  0      ;SECTOR COUNT
2260 003630 000000      MEMPAR: .WORD  0      ;MEMORY EMABLE ON FIRST 24K
2261 003632 000015      WAITIM: .WORD  15     ;WAIT TIME FOR CONTROLLER READY
2262 003634 000000      SAVSWR: .WORD  0      ;STORAGE FOR SWITCH REGISTER
2263 003636 000000      PARPRE: .WORD  0      ;PARITY OPTION PRESENT FOR
2264                                     ; LOCATION USED
2265 003640 000000      CSRPTR: .WORD  0      ;POINTER TO CSR TO BE USED
    
```

```

2266 .SBTTL PROGRAM SETUP
2267
2268 003642 012737 000001 003606 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2269 003650 000406 BR START1
2270
2271 003652 012737 177777 003606 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2272 003660 000402 BR START1
2273
2274 003662 005037 003606 START: CLR SRTFLG ;CLEAR START FLAG
2275 003666 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2276 003670 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2277 003674 004737 051302 JSR PC,$TKINT ;INIT KEYBOARD
2278 003700 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2279 003704 012746 003712 MOV #1$,-(SP) ;LOAD START OF PROGRAM
2280 003710 000002 RTI ;LOAD PSW
2281
2282 003712
2283 1$:
2284 .SBTTL INITIALIZE THE COMMON TAGS
2285 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2286 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2287 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2288 CMP #SWR,R6 ;;DONE?
2289 BNE -6 ;;LOOP BACK IF NO
2290 MOV #STACK,SP ;;SETUP THE STACK POINTER
2291 ;;INITIALIZE A FEW VECTORS
2292 MOV #SSCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2293 MOV #340,@IOTVEC+2 ;;LEVEL 7
2294 MOV #SERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2295 MOV #340,@EMTVEC+2 ;;LEVEL 7
2296 MOV #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2297 MOV #340,@TRAPVEC+2 ;;LEVEL 7
2298 MOV #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
2299 MOV #340,@PWRVEC+2 ;;LEVEL 7
2300 MOV SENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
2301 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
2302 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2303 MOV #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
2304 MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2305 MOV #,$SLPERR ;;SETUP THE ERROR LOOP ADDRESS
2306 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2307 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2308 MOV @ERRVEC,-(SP) ;;SAVE ERROR VECTOR
2309 MOV #64$,@ERRVEC ;;SET UP ERROR VECTOR
2310 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
2311 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2312 CMP #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR
2313 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2314 BR 65$ ;;AND THE HARDWARE SWR IS NOT = -1
2315 BR 65$ ;;BRANCH IF NO TIMEOUT
2316 MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
2317 MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
2318 MOV #DISPREG,DISPLAY ;;
2319 MOV (SP)+,@ERRVEC ;;RESTORE ERROR VECTOR
2320
2321 004140 005037 001222 CLR $PASS ;;CLEAR PASS COUNT

```

```

2322 004144 132737 000200 001235 BITB #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
2323 004152 001403 BEQ 67$ ;;YES,USE NON-APT SWITCH
2324 004154 012737 001236 001140 MOV #$$SWREG,$SWR ;;NO,USE APT SWITCH REGISTER
2325 004162 67$: CLR ERRCNT ;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
2326 004162 005037 003610 .SBTTL TYPE PROGRAM NAME
2327 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2328 INC #-1 ;;FIRST TIME?
2329 004166 005227 177777 BNE 68$ ;;BRANCH IF NO
2330 004172 001056 CMP #SENDAD,@#42 ;;ACT-11?
2331 004174 022737 044674 000042 BEQ 68$ ;;BRANCH IF YES
2332 004202 001452 TYPE 69$ ;TYPE ASCIZ STRING
2333 004204 104401 004252 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2334 TST @#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
2335 004210 005737 000042 BNE 70$ ;;BRANCH IF YES
2336 004214 001012 CMPB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
2337 004216 123727 001234 000001 BEQ 70$ ;;BRANCH IF YES
2338 004224 001406 CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
2339 004226 023727 001140 000176 BNE 71$ ;;BRANCH IF NO
2340 004234 001005 GTSWR ;;GET SOFT-SWR SETTINGS
2341 004236 104406 BR 71$
2342 004240 000403 BR 71$
2343 004242 112737 000001 001134 70$: MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
2344 004250 71$: BR 68$
2345 004250 000427 ;;69$: .ASCIZ <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 3 CZR6CCO/<CRLF>
2346 68$: INC #-1 ;TEST IF FIRST PASS
2347 004330 005227 177777 BNE 6$ ;NO - SKIP
2348 004330 001002 TYPE OPRO06 ;TYPE RUN TIME MESSAGE
2349 004334 104401 054604 003606 6$: CMP #1,$RTFLG ;CHECK IF PARAMETER START
2350 004336 022737 000001 003606 6$: BNE 15$ ;NO,CONTINUE SETUP
2351 004342 022737 000001 003606 6$: TYPE "RK611 BUS ADDRESS ( ) ="
2352 004350 001122 5$: MOV $BASE,-(SP) ;SAVE $BASE FOR TYPEOUT
2353 004352 104401 054472 5$: TYPOC ,OPRO01 ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2354 004356 013746 001270 TYPE ,OPRO02
2355 004362 104402 RDOCT ;GET VALUE
2356 004364 104401 054524 MOV (SP)+,$TMPD
2357 004370 104412 BEQ 7$ ;CHECK IF <CR>
2358 004372 012637 001160 CMP #160000,$TMPD ;CHECK IF IN I/O PAGE
2359 004376 001407 BHI 5$
2360 004400 022737 160000 001160 MOV $TMPD,$BASE ;LOAD NEW BUS ADDRESS
2361 004406 101361 001160 001270 7$: TYPE OPRO03 ;TYPE "RK611 VECTOR ADDRESS ( ) ="
2362 004410 013737 001160 001270 7$: MOV $VECT1,-(SP) ;TYPE OUT VECTOR ADDRESS
2363 004416 104401 054532 BIC #160000,(SP)
2364 004422 013746 001264 TYPOC ,OPRO02
2365 004426 042716 160000 RDOCT ;GET VALUE
2366 004432 104402 MOV (SP)+,$TMPD
2367 004434 104401 054524 BEQ 10$ ;CHECK IF <CR>
2368 004440 104412 CMP #1000,$TMPD ;CHECK IF LEGAL
2369 004442 012637 001160 BLOS 7$
2370 004446 001412 BIC #17777,$VECT1 ;LOAD NEW VECTOR ADDRESS
2371 004450 022737 001000 001160 BIS $TMPD,$VECT1
2372 004456 101757 001000 001160 TYPE OPRO04 ;TYPE "RK611 PRIORITY ( ) ="
2373 004460 042737 017777 001264 CLR -(SP)
2374 004466 053737 001160 001264 10$: MOV $VECT1+1,(SP)
2375 004474 104401 054562
2376 004500 005046
2377 004502 113716 001265

```

2378	004506	006216				ASR	(SP)		;SHIFT 5 BITS RIGHT
2379	004510	006216				ASR	(SP)		
2380	004512	006216				ASR	(SP)		
2381	004514	006216				ASR	(SP)		
2382	004516	006216				ASR	(SP)		
2383	004520	104402				TYPOC			
2384	004522	104401	054524			TYPE	,OPR002		
2385	004526	104412				RDOCT			;GET VALUE
2386	004530	012637	001160			MOV	(SP)+,\$TMPO		
2387	004534	001430				BEQ	15\$;CHECK FOR DEFAULT
2388	004536	022737	000007	001160		CMP	#7,\$TMPO		;CHECK IF LEGAL
2389	004544	103753				BLO	10\$		
2390	004546	022737	000004	001160		CMP	#4,\$TMPO		
2391	004554	101347				BHI	10\$		
2392	004556	006337	001160			ASL	\$TMPO		;SHIFT 5 BITS LEFT
2393	004562	006337	001160			ASL	\$TMPO		
2394	004566	006337	001160			ASL	\$TMPO		
2395	004572	006337	001160			ASL	\$TMPO		
2396	004576	006337	001160			ASL	\$TMPO		
2397	004602	042737	160000	001264		BIC	#160000,\$VECT1		;STORE NEW PRIORITY
2398	004610	153737	001160	001265		BISB	\$TMPO,\$VECT1+1		
2399	004616	013737	001264	003600	15\$:	MOV	\$VECT1,RKVEC		;STORE RK611 VECTOR
2400	004624	042737	160000	003600		BIC	#160000,RKVEC		
2401	004632	113737	001265	003602		MOVB	\$VECT1+1,RKPRI		;STORE PRIORITY
2402	004640	004737	046336			JSR	PC,\$SIZE		;SIZE MEMORY
2403	004644	013702	001270			MOV	\$BASE,R2		;SET RK611 BASE
2404	004650	005037	001202			CLR	\$ESCAPE		;CLEAR ESCAPE
2405									
2406	004654	004737	044740		NEWPAS:	JSR	PC,PARCHK		;CHECK OF MEMORY CHECK ENABLE
2407	004660	012746	000000			MOV	#PRO,-(SP)		;ALLOW ALL INTERRUPTS
2408	004664	012746	004672			MOV	#TST1,-(SP)		
2409	004670	000002				RTI			

.SBTTL **DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

*TEST 1 READ HEADER SEEK MESSAGE

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

*ST1: SCOPE

MOV #100, \$TIMES ; DO 100. ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #1777, RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ; LOAD TRACK
MOV #7, RKCS2(R2) ; LOAD DRIVE NUM.
MOV #CDT!CFMT!RDHEAD, RKCS1(R2) ; ISSUE RDHEAD WITH CDT SET IN
; 24 SECTOR FORMAT
; CLOCK IN DRIVE MESSAGE
1\$: MOV #3*4+2, RO
MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC RO
1\$ BNE 1\$
MOV RKCS1(R2), T.CS1 ; STORE COMMAND STATUS REG. 1
MOV RKMR2(R2), T.MR2 ; STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2), T.MR3 ; STORE MAINT REG. 3 (MESS B)
MOV #CDT!CFMT!RDHEAD, E.CS1 ; LOAD EXPECTED CS1
MOV #S.SEEK!S.FMT!70007, E.MR2 ; LOAD EXPECTED MR2
MOV #37760, E.MR3 ; LOAD EXPECTED MR3
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2\$; YES, CHECK MESSAGE A
ERROR 1 ; CS1 INCORRECT
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
BR TST2 ; GO TO NEXT TEST
2\$: CMP E.MR2, T.MR2 ; CHECK MESS A CORRECT
BEQ 3\$; YES, CHECK MESSAGE B
ERROR 2 ; MESS A INCORRECT
3\$: CMP E.MR3, T.MR3 ; CHECK MESS B CORRECT
BEQ TST2 ; YES, GO ON TO NEXT TEST
ERROR 3 ; MESS B INCORRECT

*TEST 2 WRITE HEADER SEEK MESSAGE

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

2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465

004672 000004
004674 012737 000144 001200
004702 013702 001270
004706 012762 100000 000000
004714 012762 000040 000026
004722 012762 001777 000020
004730 012762 003400 000006
004736 012762 000007 000010
004744 012762 012025 000000

004752 012700 000016
004756 012762 000440 000026 1\$:
004764 012762 000040 000026
004772 005300
004774 001370
004776 016237 000000 003440
005004 016237 000034 003466
005012 016237 000036 003470
005020 012737 012025 003500
005026 012737 071027 003526
005034 012737 037760 003530
005042 023737 003500 003440
005050 001405
005052 104001
005054 012762 100000 000000
005062 000412
005064 023737 003526 003466 2\$:
005072 001401
005074 104002
005076 023737 003530 003470 3\$:
005104 001401
005106 104003

```

2466      ;*      CYLINDER 1777, HEAD 7, DRIVE 7.
2467      ;*
2468      ;* *****
2469      TST2:  SCOPE
2470      MOV      #100., $TIMES      ;; DO 100. ITERATIONS
2471      MOV      $BASE, R2          ;; LOAD RK611 BASE
2472      MOV      #CCLR, RKCS1(R2)   ;; CLEAR RK611
2473      MOV      #DMD, RKMR1(R2)    ;; PUT RK611 IN DIAGNOSTIC MODE
2474      MOV      #1777, RKDCYL(R2)  ;; LOAD CYLINDER ADDRESS REG.
2475      MOV      #3400, RKDA(R2)    ;; LOAD TRACK
2476      MOV      #7, RKCS2(R2)     ;; LOAD DRIVE NUM.
2477      MOV      #CDT!CFMT!WRHEAD, RKCS1(R2) ;; ISSUE WRHEAD WITH CDT SET IN
2478      ;*      24 SECTOR FORMAT
2479      MOV      #3*4+2, RD          ;; CLOCK IN DRIVE MESSAGE
2480      1$:    MOV      #DMD!MCLK, RKMR1(R2)
2481      MOV      #DMD, RKMR1(R2)
2482      DEC      RD
2483      BNE     1$
2484      MOV      RKCS1(R2), T.CS1    ;; STORE COMMAND STATUS REG. 1
2485      MOV      RKMR2(R2), T.MR2    ;; STORE MAINT REG. 2 (MESS A)
2486      MOV      RKMR3(R2), T.MR3    ;; STORE MAINT REG. 3 (MESS B)
2487      MOV      #CDT!CFMT!WRHEAD, E.CS1 ;; LOAD EXPECTED CS1
2488      MOV      #S.SEEK!S.RTC!S.FMT!70007, E.MR2 ;; LOAD EXPECTED MR2
2489      MOV      #37760, E.MR3      ;; LOAD EXPECTED MR3
2490      CMP      E.CS1, T.CS1      ;; CHECK COMMAND AND STATUS REG. 1 CORRECT
2491      BEQ      2$
2492      ERROR    4                  ;; YES, CHECK MESSAGE A
2493      MOV      #CCLR, RKCS1(R2)    ;; CS1 INCORRECT
2494      BR       TST3              ;; GO TO NEXT TEST
2495      2$:    CMP      E.MR2, T.MR2  ;; CHECK MESS A CORRECT
2496      BEQ      3$
2497      ERROR    5                  ;; YES, CHECK MESSAGE B
2498      3$:    CMP      E.MR3, T.MR3  ;; MESS A INCORRECT
2499      BEQ      TST3              ;; CHECK MESS B CORRECT
2500      ERROR    6                  ;; YES, GO ON TO NEXT TEST
2501      ;*      MESS 'B' INCORRECT
2502      ;* *****
2503      ;* TEST 3      READ HEADER DRIVE CLEAR MESSAGE
2504      ;*
2505      ;*      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
2506      ;*      IN DIAGNOSTIC MODE.  ISSUE A READ HEADER WITH CDT SET
2507      ;*      IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
2508      ;*      CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS
2509      ;*      GENERATED AND THE PROPER BITS ARE SET.
2510      ;* *****
2511      TST3:  SCOPE
2512      MOV      #100., $TIMES      ;; DO 100. ITERATIONS
2513      MOV      $BASE, R2          ;; LOAD RK611 BASE
2514      MOV      #CCLR, RKCS1(R2)   ;; CLEAR RK611
2515      MOV      #DMD, RKMR1(R2)    ;; PUT RK611 IN DIAGNOSTIC MODE
2516      MOV      #1777, RKDCYL(R2)  ;; LOAD CYLINDER ADDRESS REG.
2517      MOV      #3400, RKDA(R2)    ;; LOAD TRACK
2518      MOV      #7, RKCS2(R2)     ;; LOAD DRIVE NUMBER
2519      MOV      #CDT!CFMT!RDHEAD, RKCS1(R2) ;; ISSUE COMMAND WITH CDT SET IN
2520      ;*      24 SECTOR FORMAT
2521

```

L04

CZR6CCD RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 50
T3 READ HEADER DRIVE CLEAR MESSAGE

SEQ 0050

2522	005406	012700	000156			MOV	#27.*4+2,RO	:LOAD COUNT TO LOAD DRIVE CLEAR
2523	005412	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2524	005420	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2525	005426	005300				DEC	RO	
2526	005430	001370				BNE	1\$	
2527	005432	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2528	005440	016237	000034	003466		MOV	RKMR2(R2),T.MR2	:STORE MAINT. REG. 2 (MESS A)
2529	005446	016237	000036	003470		MOV	RKMR3(R2),T.MR3	:STORE MAINT. REG. 3 (MESS B)
2530	005454	012737	012025	003500		MOV	#CDT!CFMT!RDHEAD,E.CS1	:LOAD EXPECTED CS1
2531	005462	012737	071407	003526		MOV	#S.CLR!S.FMT!70007,E.MR2	:LOAD EXPECTED MAINT REG. 2
2532	005470	005037	003530			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2533	005474	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
2534	005502	001405				BEQ	2\$:YES, CHECK CS2
2535	005504	104007				ERROR	7	:CS1 INCORRECT
2536	005506	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2537	005514	000412				BR	TST4	:GO TO NEXT TEST
2538	005516	023737	003526	003466	2\$:	CMP	E.MR2,T.MR2	:CHECK MESS A CORRECT
2539	005524	001401				BEQ	3\$:YES, CHECK MESS B
2540	005526	104010				ERROR	10	:MESS A INCORRECT
2541	005530	023737	003530	003470	3\$:	CMP	E.MR3,T.MR3	:CHECK MESS B CORRECT
2542	005536	001401				BEQ	TST4	:YES, GO ON TO NEXT TEST
2543	005540	104011				ERROR	11	:MESS B INCORRECT

 *TEST 4 WRITE HEADER DRIVE CLEAR MESSAGE
 *
 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER WITH CDT SET
 * IN 24 SECTOR FORMAT, CYLINDER 1777 HEAD 7 DRIVE 7.
 * CLOCK SEEK MESSAGE AND LOAD GENERATED DRIVE CLEAR
 * INTO SHIFT REGISTER. MAKE SURE THE DRIVE CLEAR IS
 * GENERATED AND THE PROPER BITS ARE SET.
 *

2544								
2545								
2546								
2547								
2548								
2549								
2550								
2551								
2552								
2553								
2554								
2555								
2556	005542	000004			TST4:	SCOPE		
2557	005544	012737	000144	001200		MOV	#100,\$TIMES	:DO 100. ITERATIONS
2558	005552	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
2559	005556	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2560	005564	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
2561	005572	012762	001777	000020		MOV	#1777,RKDCYL(R2)	:LOAD CYLINDER ADDRESS REG.
2562	005600	012762	003400	000006		MOV	#3400,RKDA(R2)	:LOAD TRACK
2563	005606	012762	000007	000010		MOV	#7,RKCS2(R2)	:LOAD DRIVE NUMBER
2564	005614	012762	012027	000000		MOV	#CDT!CFMT!WRHEAD,RKCS1(R2)	:ISSUE COMMAND WITH CDT SET IN : 24 SECTOR FORMAT
2565								
2566	005622	012700	000156			MOV	#27.*4+2,RO	:LOAD COUNT TO LOAD DRIVE CLEAR
2567	005626	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2568	005634	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2569	005642	005300				DEC	RO	
2570	005644	001370				BNE	1\$	
2571	005646	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2572	005654	016237	000034	003466		MOV	RKMR2(R2),T.MR2	:STORE MAINT. REG. 2 (MESS A)
2573	005662	016237	000036	003470		MOV	RKMR3(R2),T.MR3	:STORE MAINT. REG. 3 (MESS B)
2574	005670	012737	012027	003500		MOV	#CDT!CFMT!WRHEAD,E.CS1	:LOAD EXPECTED CS1
2575	005676	012737	071407	003526		MOV	#S.CLR!S.FMT!70007,E.MR2	:LOAD EXPECTED MAINT REG. 2
2576	005704	005037	003530			CLR	E.MR3	:LOAD EXPECTED MAINT REG.
2577	005710	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT

```

2578 005716 001405 BEQ 2$ ;YES, CHECK CS2
2579 005720 104012 ERROR 12 ;CS1 INCORRECT
2580 005722 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2581 005730 000412 BR TST5 ;GO TO NEXT TEST
2582 005732 023737 003526 003466 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2583 005740 001401 BEQ 3$ ;YES, CHECK MESS B
2584 005742 104013 ERROR 13 ;MESS A INCORRECT
2585 005744 023737 003530 003470 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2586 005752 001401 BEQ TST5 ;YES, GO ON TO NEXT TEST
2587 005754 104014 ERROR 14 ;MESS B INCORRECT

```

.SBTTL **INDEX AND SECTOR PULSE DETECT ON

```

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

```

```

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

```

MAKE SURE READ GATE DOES SET.

```

*****
TST5: SCOPE
*****

```

```

2604 005756 000004 TST5: SCOPE
2605 005760 012737 000012 001200 MOV #10,STIMES ;DO 10. ITERATIONS
2606 005766 013702 001270 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2607 005772 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2608 006000 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2609 006006 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2610 006014 012762 000040 000026 MOV #DMD,RKMR1(R2)
2611 006022 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2612 006030 012700 000312 000000 MOV #50,#4+2,RO ;CLOCK UNTIL READY FOR SECTOR PULSE
2613 006034 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2614 006042 012762 000040 000026 MOV #DMD,RKMR1(R2)
2615 006050 005300 DEC RO
2616 006052 001370 BNE 1$
2617 006054 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2618 006062 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2619 006070 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2620 006076 001405 BEQ 2$ ;YES, CLOCK ZEROES
2621 006100 104015 ERROR 15 ;CS1 INCORRECT
2622 006102 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2623 006110 000553 BR TST6 ;GO ON TO NEXT TEST
2624
2625 006112 012762 000140 000026 2$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
2626 006120 012762 000040 000026 MOV #DMD,RKMR1(R2)
2627 006126 012737 022040 003524 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECT MAINT REG. 1
2628 006134 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
2629 006140 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2630 006144 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROES
2631 006150 012700 000200 MOV #128,RO ;GENERATE 128 ZEROS UNTIL READ GATE
2632 006154 004737 046164 5$: JSR PC,RDBIT ;READ A ZERO
2633 006160 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT REG. 1

```

2634	006166	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINTENANCE REG. 1 CORRECT
2635	006174	001405				BEQ	6\$:YES, SIMULATE NEXT BIT
2636	006176	104017				ERROR	17	:MAINT REG. 1 INCORRECT
2637	006200	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2638	006206	000514				BR	TST6	::GO ON TO NEXT TEST
2639								
2640	006210	005237	003622		6\$:	INC	BITCNT	:INCREMENT BIT COUNT
2641	006214	005300				DEC	RO	:CHECK READY OF READ GATE
2642	006216	001356				BNE	5\$:NO, CONTINUE
2643	006220	012737	122040	003524		MOV	#DMD!MEWD!ECCW!RDGATE,E.MR1	:LOAD EXPECTED MR1
2644	006226	012700	000177			MOV	#127,RO	:GENERATE 127 ZEROS
2645	006232	004737	046164		10\$:	JSR	PC,RDBIT	:READ A ZERO
2646	006236	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1
2647	006244	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINT REG. 1 CORRECT
2648	006252	001405				BEQ	11\$:YES, SIMULATE NEXT BIT
2649	006254	104017				ERROR	17	:MAINT REG. 1 INCORRECT
2650	006256	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2651	006264	000465				BR	TST6	::GO ON TO NEXT TEST
2652								
2653	006266	005237	003622		11\$:	INC	BITCNT	:INCREMENT BIT COUNT
2654	006272	005300				DEC	RO	:CHECK IF ALL ZEROS ISSUED
2655	006274	001356				BNE	10\$:NO, CONTINUE
2656	006276	012737	000001	003614		MOV	#1,PR.BIT	:LOAD ONE FOR READING 1
2657	006304	004737	046164			JSR	PC,RDBIT	:READ A ONE
2658	006310	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINTENANCE REG.
2659	006316	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINT REG. 1
2660	006324	001405				BEQ	12\$:YES, CONTINUE
2661	006326	104017				ERROR	17	:MAINTENANCE REG. 1 INCORRECT
2662	006330	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2663	006336	000440				BR	TST6	::GO ON TO NEXT TEST
2664	006340	005237	003622		12\$:	INC	BITCNT	:INCREMENT BIT COUNT
2665	006344	013737	003614	003616		MOV	PR.BIT,M1.BIT	:LOAD ZERO FOR NEXT BIT
2666	006352	005037	003614			CLR	PR.BIT	
2667	006356	004737	046164			JSR	PC,RDBIT	:SIMULATE ZERO
2668	006362	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINTENANCE REG. 1
2669	006370	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINT REG. 1 CORRECT
2670	006376	001405				BEQ	13\$:CHECK CSI CORRECT
2671	006400	104017				ERROR	17	:MAINTENANCE REG. 1 INCORRECT
2672	006402	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
2673	006410	000413				BR	TST6	::GO TO NEXT TEST
2674								
2675	006412	016237	000000	003440	13\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2676	006420	012737	000025	003500		MOV	#RDHEAD,E.CS1	:LOAD EXPECTED CS1
2677	006426	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CSI CORRECT
2678	006434	001401				BEQ	TST6	:YES, GO TO NEXT TEST
2679	006436	104016				ERROR	16	:CSI INCORRECT

```

2680
2681 *****
2682 *TEST 6 SECTOR PULSE DETECT IN READ HEADER (PART 2)
2683 *
2684 *
2685 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2686 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
2687 * IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
2688 * CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
2689 * SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.

```

```

2690          ;*      MAKE SURE READ GATE DOES NOT SET.
2691          ;*
2692          ;* *****
2693 006440 000004          †ST6: SCOPE
2694 006442 012737 000012 001200      MOV      #10, $TIMES          ;; DO 10. ITERATIONS
2695 006450 013702 001270          MOV      $BASE, R2          ;; LOAD RK611 BASE
2696 006454 012762 100000 000000      MOV      #CCLR, RKCS1(R2)  ;; CLEAR RK611
2697 006462 012762 000040 000000      MOV      #DMD, RKCS1(R2)  ;; PUT RK611 IN DIAGNOSTIC MODE
2698 006470 012762 000140 000026      MOV      #DMD!MSP, RKMR1(R2) ; INITIALIZE ROM ADDRESS
2699 006476 012762 000040 000026      MOV      #DMD, RKMR1(R2)
2700 006504 012762 000025 000000      MOV      #RDHEAD, RKCS1(R2) ; ISSUE READ HEADER
2701 006512 012700 000312          MOV      #50, #4+2, RO     ;; CLOCK UNTIL READY FOR SECTOR PULSE
2702 006516 012762 000440 000026 1$:  MOV      #DMD!MCLK, RKMR1(R2)
2703 006524 012762 000040 000026      MOV      #DMD, RKMR1(R2)
2704 006532 005300          DEC      RO
2705 006534 001370          BNE     1$
2706 006536 016237 000000 003440      MOV      RKCS1(R2), T.CS1  ;; STORE COMMAND AND STATUS REG. 1
2707 006544 012737 000025 003500      MOV      #RDHEAD, E.CS1   ;; LOAD EXPECTED CS1
2708 006552 023737 003500 003440      CMP      E.CS1, T.CS1     ;; CHECK COMMAND AND STATUS REG. 1 CORRECT
2709 006560 001405          BEQ     2$                ;; YES, CLOCK IN ZEROS
2710 006562 104015          ERROR  1$
2711 006564 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ; CLEAR RK611
2712 006572 000535          BR      TST7             ;; GO ON TO NEXT TEST
2713 006574          2$:
2714 006574 012762 000240 000026      MOV      #DMD!MIND, RKMR1(R2) ; SIMULATE INDX PULSE
2715 006602 012700 000004          MOV      #4, RO
2716 006606 012762 000640 000026 3$:  MOV      #DMD!MIND!MCLK, RKMR1(R2)
2717 006614 012762 000240 000026      MOV      #DMD!MIND, RKMR1(R2)
2718 006622 005300          DEC      RO
2719 006624 001370          BNE     3$
2720 006626 012762 000040 000026      MOV      #DMD, RKMR1(R2)
2721 006634 005037 003622          CLR      BITCNT          ;; INITIALIZE BIT COUNT
2722 006640 012737 022040 003524      MOV      #DMD!MEWD!ECCW, E.MR1 ; LOAD EXPECTED MAINTENANCE REG. 1
2723 006646 005037 003614          CLR      PR.BIT          ;; INITIALIZE PRESENT AND PREVIOUS
2724 006652 005037 003616          CLR      M1.BIT          ;; BITS TO GENERATE ZEROS
2725 006656 012700 000377          MOV      #255, RO        ;; GENERATE 255 ZEROES
2726 006662 004737 046164          JSR     PC, RDBIT        ;; READ A ZERO
2727 006666 016237 000026 003464 5$:  MOV      RKMR1(R2), T.MR1  ;; STORE MAINTENANCE REG. 1
2728 006674 023737 003524 003464      CMP      E.MR1, T.MR1    ;; CHECK READ GATE NOT SET
2729 006702 001405          BEQ     6$                ;; READ GATE NOT SET SIMULATE NEXT BIT
2730 006704 104020          ERROR  2$                ;; MAINT REG. 1 INCORRECT
2731 006706 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ; ISSUE CONTROLLER CLEAR
2732 006714 000464          BR      TST7             ;; GO ON TO NEXT TEST
2733
2734 006716 005237 003622          6$:  INC      BITCNT          ;; INCREMENT BIT COUNT
2735 006722 005300          DEC      RO              ;; CHECK IF ALL ZEROES ISSUED
2736 006724 001356          BNE     5$                ;; NO, CONTINUE
2737 006726 012737 000001 003614      MOV      #1, PR.BIT      ;; LOAD ONE FOR READING 1
2738 006734 004737 046164          JSR     PC, RDBIT        ;; READ A ONE
2739 006740 016237 000026 003464      MOV      RKMR1(R2), T.MR1 ; STORE MAINTENANCE REG. 1
2740 006746 023737 003524 003464      CMP      E.MR1, T.MR1    ;; CHECK READ GATE NOT SET
2741 006754 001405          BEQ     7$                ;; YES, CONTINUE
2742 006756 104020          ERROR  3$                ;; MAINT REG. 1 INCORRECT
2743 006760 012762 100000 000000      MOV      #CCLR, RKCS1(R2) ; ISSUE CONTROLLER CLEAR
2744 006766 000437          BR      TST7             ;; GO ON TO NEXT TEST
2745

```

C05

CZR6CC0 RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T6

02-DEC-77 09:56 PAGE 54
SECTOR PULSE DETECT IN READ HEADER (PART 2)

SEQ 0054

```

2746 006770 005237 003622      7$:  INC      BITCNT      ;INCREMENT BIT COUNT
2747 006774 013737 003614 003616  MOV      PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2748 007002 005037 003614      CLR      PR.BIT
2749 007006 004737 046164      JSR      PC,RDBIT      ;SIMULATE ZERO
2750 007012 016237 000026 003464  MOV      RKMRI(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2751 007020 023737 003524 003464  CMP      E.MR1,T.MR1    ;CHECK MAINTENANCE REG. 1 CORRECT
2752 007026 001404      BEQ      9$            ;CHECK CSI CORRECT
2753 007030 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2754 007036 000413      BR       TST7         ;GO TO NEXT TEST
2755
2756 007040 016237 000000 003440 9$:  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2757 007046 012737 000025 003500  MOV      #RDHEAD,E.CS1  ;LOAD EXPECTED CS1
2758 007054 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK CSI CORRECT
2759 007062 001401      BEQ      TST7         ;YES, GO TO NEXT TEST
2760 007064 104016      ERROR   16           ;CSI INCORRECT
2761

```

```

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

```

```

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

```

```

: MAKE SURE READ GATE DOES NOT SET.

```

```

*****

```

```

2773
2774 007066 000004      TST7: SCOPE
2775 007070 012737 000012 001200  MOV      #10,STIMES    ;DO 10. ITERATIONS
2776 007076 013702 001270      MOV      $BASE,R2      ;LOAD RK611 BASE
2777 007102 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2778 007110 012762 000040 000000  MOV      #DMD,RKCS1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2779 007116 012762 000140 000026  MOV      #DMD!MSP,RKMRI(R2) ;INITIALIZE ROM ADDRESS
2780 007124 012762 000040 000026  MOV      #DMD,RKMRI(R2)
2781 007132 012762 000025 000000  MOV      #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2782 007140 012700 000312      MOV      #50,#4+2,RO   ;CLOCK UNTIL READY FOR SECTOR PULSE
2783 007144 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMRI(R2)
2784 007152 012762 000040 000026  MOV      #DMD,RKMRI(R2)
2785 007160 005300      DEC      RO
2786 007162 001370      BNE     1$
2787 007164 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2788 007172 012737 000025 003500  MOV      #RDHEAD,E.CS1  ;LOAD EXPECTED CS1
2789 007200 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2790 007206 001405      BEQ      2$            ;YES, CLOCK IN ZEROS
2791 007210 104015      ERROR   15
2792 007212 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2793 007220 000515      BR       TST10        ;GO ON TO NEXT TEST
2794
2795 007222 005037 003622      2$:  CLR      BITCNT      ;INITIALIZE BIT COUNT
2796 007226 012737 022040 003524  MOV      #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MAINTENANCE REG. 1
2797 007234 005037 003614      CLR      PR.BIT        ;INITIALIZE PRESENT AND PREVIOUS
2798 007240 005037 003616      CLR      M1.BIT        ;BITS TO GENERATE ZEROS
2799 007244 012700 000377      MOV      #255,RO       ;GENERATE 255 ZEROES
2800 007250 004737 046164      JSR      PC,RDBIT      ;READ A ZERO
2801 007254 016237 000026 003464  MOV      RKMRI(R2),T.MR1 ;STORE MAINTENANCE REG. 1

```

2802	007262	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK READ GATE NOT SET
2803	007270	001405				BEQ	6\$;READ GATE NOT SET SIMULATE NEXT BIT
2804	007272	104021				ERROR	21	;MAINT REG. 1 INCORRECT
2805	007274	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;ISSUE CONTROLLER CLEAR
2806	007302	000464				BR	TST10	;GO ON TO NEXT TEST
2807								
2808	007304	005237	003622		6\$:	INC	BITCNT	;INCREMENT BIT COUNT
2809	007310	005300				DEC	R0	;CHECK IF ALL ZEROES ISSUED
2810	007312	001356				BNE	5\$;NO, CONTINUE
2811	007314	012737	000001	003614		MOV	#1,PR.BIT	;LOAD ONE FOR READING 1
2812	007322	004737	046164			JSR	PC,RDBIT	;READ A ONE
2813	007326	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MAINTENANCE REG. 1
2814	007334	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK READ GATE NOT SET
2815	007342	001405				BEQ	7\$;YES, CONTINUE
2816	007344	104021				ERROR	21	;MAINT REG. 1 INCORRECT
2817	007346	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;ISSUE CONTROLLER CLEAR
2818	007354	000437				BR	TST10	;GO ON TO NEXT TEST
2819								
2820	007356	005237	003622		7\$:	INC	BITCNT	;INCREMENT BIT COUNT
2821	007362	013737	003614	003616		MOV	PR.BIT,M1.BIT	;LOAD ZERO FOR NEXT BIT
2822	007370	005037	003614			CLR	PR.BIT	
2823	007374	004737	046164			JSR	PC,RDBIT	;SIMULATE ZERO
2824	007400	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MAINTENANCE REG. 1
2825	007406	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK MAINTENANCE REG. 1 CORRECT
2826	007414	001404				BEQ	9\$;CHECK CS1 CORRECT
2827	007416	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
2828	007424	000413				BR	TST10	;GO TO NEXT TEST
2829								
2830	007426	016237	000000	003440	9\$:	MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
2831	007434	012737	000025	003500		MOV	#RDHEAD,E.CS1	;LOAD EXPECTED CS1
2832	007442	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2833	007450	001401				BEQ	TST10	;YES, GO TO NEXT TEST
2834	007452	104016				ERROR	16	;CS1 INCORRECT

```

*****
;TEST 10 INDEX PULSE DETECTION IN WRITE HEADER
*****
;
; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
; THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER
; TO AN RK06 IN 26 SECTOR FORMAT TO CYLINDER 0, HEAD 0,
; DRIVE 0, WITH A ONE WORD TRANSFER. CLOCK THROUGH THE
; SEEK AND THE DRIVE CLEAR MESSAGES. ISSUE 200 CONTROLLER
; CLOCKS AND MAKE SURE WRITE GATE DOES NOT SET. SIMULATE
; SECTOR PULSE AND 200 CONTROLLER CLOCKS MAKING SURE WRITE
; GATE DOES NOT SET. SIMULATE INDEX PULSE AND MAKE SURE
; WRITE GATE SETS.
*****

```

2849								
2850	007454	000004			TST10:	SCOPE		
2851	007456	012737	000012	001200		MOV	#10,\$TIMES	;DO 10. ITERATIONS
2852	007464	013702	001270			MOV	\$BASE,R2	;LOAD RK611 BASE
2853	007470	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
2854	007476	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
2855	007504	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;INITIALIZE ROM ADDRESS
2856	007512	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2857	007520	012762	067264	000004		MOV	#WRBUFF,RKBA(R2)	;LOAD BUS ADDRESS

E05

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T10

02-DEC-77 09:56 PAGE 56
INDEX PULSE DETECTION IN WRITE HEADER

SEQ 0056

2858	007526	012762	177777	000002		MOV	#-1,RKWC(R2)	:LOAD WORD COUNT
2859	007534	012762	000027	000000		MOV	#WRHEAD,RKCS1(R2)	:ISSUE WRITE HEADER
2860	007542	012700	000312			MOV	#50,*4+2,RO	:CLOCK UNTIL READY FOR INDEX PULSE
2861	007546	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2862	007554	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2863	007562	005300				DEC	RO	
2864	007564	001370				BNE	1\$	
2865	007566	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2866	007574	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
2867	007602	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
2868	007610	012737	000027	003500		MOV	#WRHEAD,E.CS1	:LOAD EXPECTED CS1
2869	007616	012737	067264	003504		MOV	#WRBUFF,E.BA	:LOAD EXPECTED BUS ADDRESS
2870	007624	012737	177777	003502		MOV	#-1,E.WC	:LOAD EXPECTED WORD COUNT
2871	007632	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
2872	007640	001403				BEQ	2\$:YES, CONTINUE
2873	007642	104022				ERROR	22	:CS1 INCORRECT
2874	007644	000137	010572			JMP	60\$:CLEAR CONTROLLER
2875								
2876	007650	023737	003504	003444	2\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
2877	007656	001403				BEQ	3\$:YES, CONTINUE
2878	007660	104023				ERROR	23	:BUS ADDRESS INCORRECT
2879	007662	000137	010572			JMP	60\$:CLEAR RK611
2880								
2881	007666	023737	003502	003442	3\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
2882	007674	001403				BEQ	4\$:YES, CONTINUE
2883	007676	104024				ERROR	24	:WORD COUNT INCORRECT
2884	007700	000137	010572			JMP	60\$:CLEAR RK611
2885								
2886	007704	012737	022040	003524	4\$:	MOV	#ECCW!MEWD!DMD,E.MR1	:LOAD EXPECTED MAINT REG. 1
2887	007712	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
2888	007716	012700	000310			MOV	#200,RO	:ISSUE 200 MAINT BITS
2889	007722	012762	000440	000026	5\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2890	007730	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2891	007736	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	
2892	007744	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2893	007752	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1
2894	007760	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK MAINTENANCE REG. 1 CORRECT
2895	007766	001403				BEQ	6\$:YES, CONTINUE
2896	007770	104025				ERROR	25	:MAINT REG 1 INCORRECT
2897	007772	000137	010572			JMP	60\$:CLEAR RK611
2898								
2899	007776	005300			6\$:	DEC	RO	:CHECK IF READY FOR SECTOR PULSE
2900	010000	001350				BNE	5\$:NO, GET NEXT BIT
2901	010002	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
2902	010010	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG
2903	010016	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG
2904	010024	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
2905	010032	001403				BEQ	10\$:YES, CONTINUE
2906	010034	104026				ERROR	26	:CS1 INCORRECT
2907	010036	000137	010572			JMP	60\$:CLEAR RK611
2908								
2909	010042	023737	003504	003444	10\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
2910	010050	001403				BEQ	11\$:YES, CONTINUE
2911	010052	104027				ERROR	27	:BUS ADDRESS INCORRECT
2912	010054	000137	010572			JMP	60\$:CLEAR RK611
2913								

F05

CZR6CCO RK611 DSKLS CTRL PRT3 MACY11 30(1046) 02-DEC-77 09:56 PAGE 57
 CZR6CC.P11 02-DEC-77 09:46 T10 INDEX PULSE DETECTION IN WRITE HEADER

SEQ 0057

2914	010060	023737	003502	003442	11\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT REG. CORRECT
2915	010066	001403				BEQ	12\$;YES, CONTINUE
2916	010070	104030				ERROR	30	;WORD COUNT INCORRECT
2917	010072	000137	010572			JMP	60\$;CLEAR RK611
2918								
2919	010076	012700	000004		12\$:	MOV	#4,RO	;SIMULATE SECTOR PULSE
2920	010102	012762	000140	000026		MOV	#DMD:MSP,RKMR1(R2)	
2921	010110	012762	000540	000026	13\$:	MOV	#DMD:MSP!MCLK,RKMR1(R2)	
2922	010116	012762	000140	000026		MOV	#DMD:MSP,RKMR1(R2)	
2923	010124	005300				DEC	RO	
2924	010126	001370				BNE	13\$	
2925	010130	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2926	010136	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
2927	010144	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS AND REG.
2928	010152	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT REG.
2929	010160	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK COMMAND AND STATUS REG. 1 INCORRECT
2930	010166	001402				BEQ	15\$;YES, CONTINUE
2931	010170	104026				ERROR	26	;CS1 INCORRECT
2932	010172	000577				BR	60\$;CLEAR RK611
2933								
2934	010174	023737	003504	003444	15\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
2935	010202	001402				BEQ	16\$;YES, CONTINUE
2936	010204	104027				ERROR	27	;BUS ADDRESS INCORRECT
2937	010206	000571				BR	60\$;CLEAR RK611
2938								
2939	010210	023737	003502	003442	16\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
2940	010216	001402				BEQ	20\$;YES, CONTINUE
2941	010220	104030				ERROR	30	;WORD COUNT INCORRECT
2942	010222	000563				BR	60\$;CLEAR RK611
2943								
2944	010224	005037	003622		20\$:	CLR	BITCNT	;INITIALIZE BIT COUNT
2945	010230	012700	000310			MOV	#200,RO	;ISSUE 200 MAINT BITS
2946	010234	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2947	010242	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2948	010250	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	
2949	010256	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2950	010264	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MAINT REG. 1
2951	010272	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK MAINT REG. 1 CORRECT
2952	010300	001402				BEQ	22\$;YES, CONTINUE
2953	010302	104025				ERROR	25	;MAINT REG. 1 INCORRECT
2954	010304	000532				BR	60\$;CLEAR RK611
2955								
2956	010306	005300			22\$:	DEC	RO	;CHECK IF READY FOR INDEX PULSE
2957	010310	001351				BNE	21\$;NO, GET NEXT BIT
2958	010312	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
2959	010320	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS
2960	010326	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
2961	010334	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2962	010342	001402				BEQ	23\$;YES, CONTINUE
2963	010344	104026				ERROR	26	;CS1 INCORRECT
2964	010346	000511				BR	60\$;CLEAR RK611
2965								
2966	010350	023737	003504	003444	23\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
2967	010356	001402				BEQ	24\$;YES, CONTINUE
2968	010360	104027				ERROR	27	;BUS ADDRESS INCORRECT
2969	010362	000503				BR	60\$;CLEAR RK611

G05

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T10

02-DEC-77 09:56 PAGE 58
INDEX PULSE DETECTION IN WRITE HEADER

SEQ 0058

```

2970
2971 010364 023737 003502 003442 24$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
2972 010372 001402 BEQ 25$ ;YES, CONTINUE
2973 010374 104030 ERROR 30 ;WORD COUNT INCORRECT
2974 010376 000475 BR 60$ ;CLEAR RK611
2975
2976 010400 012762 000240 000026 25$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE PULSE
2977 010406 012700 000004 MOV #4,R0
2978 010412 012762 000640 000026 26$: MOV #DMD!MIND!MCLK,RKMR1(R2)
2979 010420 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
2980 010426 005300 DEC R0
2981 010430 001370 BNE 26$
2982 010432 012762 000040 000026 MOV #DMD,RKMR1(R2)
2983 010440 012700 000002 MOV #2,R0 ;SIMULATE TWO CLOCK PULSES FOR WRITE
; GATE TO COME UP
2984
2985 010444 012762 000440 000026 27$: MOV #DMD!MCLK,RKMR1(R2)
2986 010452 012762 000040 000026 MOV #DMD,RKMR1(R2)
2987 010460 005300 DEC R0
2988 010462 001370 BNE 27$
2989 010464 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.
2990 010472 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
2991 010500 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
2992 010506 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
2993 010514 012737 062040 003524 MOV #WRTGAT!MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MAINT REG. 1
2994 010522 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
2995 010530 001401 BEQ 28$ ;YES, CONTINUE
2996 010532 104031 ERROR 31 ;CS1 INCORRECT
2997 010534 023737 003504 003444 28$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
2998 010542 001401 BEQ 29$ ;YES, CONTINUE
2999 010544 104032 ERROR 32 ;BUS ADDRESS INCORRECT
3000 010546 023737 003502 003442 29$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3001 010554 001401 BEQ 30$ ;YES, CONTINUE
3002 010556 104033 ERROR 33 ;WORD COUNT INCORRECT
3003 010560 023737 003524 003464 30$: CMP E.MR1,T.MR1 ;CHECK MAINT REG 1 CORRECT
3004 010566 001401 BEQ 60$ ;YES, CLEAR RK611
3005 010570 104034 ERROR 34 ;MAINT REG. 1 INCORRECT
3006 010572 012762 100000 000000 60$: MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

.SBTTL **NPR READING OF MEMORY

```

*****
*TEST 11 NPR OUTPUT DATA TRANSFER
*****
*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 777, HEAD 7, DRIVE 7.
* SPECIFY A ONE WORD DATA TRANSFER. CLOCK BOTH SEEK
* AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE.
* CLOCK IN FIRST WORD OF NPR TRANSFER. CHECK INPUT READY,
* OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF
* THE SILO. REPEAT FOR 8 DIFFERENT DATA PATTERNS.
*
*****

```

```

3023 010600 000004 TST11: SCOPE
3024 010602 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS
3025 010610 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE

```

H05

CZR6CCD RK611 DSKLS CTRL PRT3 MACY11 30(1046) 02-DEC-77 09:56 PAGE 59
 CZR6CC.P11 02-DEC-77 09:46 T11 NPR OUTPUT DATA TRANSFER

SEQ 0059

3026	010614	012703	066744			MOV	#PATTERN,R3	;LOAD PATTERN ADDRESS
3027	010620	012704	000010			MOV	#8.,R4	;LOAD PATTERN COUNT
3028	010624	012737	067266	003504		MOV	#WRBUFF+2,E.BA	;LOAD EXPECTED BUS ADDRESS
3029	010632	012737	000027	003500		MOV	#WRHEAD,E.CS1	;LOAD EXPECTED CS1
3030	010640	005037	003502			CLR	E.WC	;LOAD EXPECTED WORD COUNT
3031	010644	012737	010652	001110		MOV	#1\$,SLPERR	;LOAD LOOP ON ERROR LOCATION FOR ;SUBTEST LOOP
3032								
3033								
3034	010652				1\$:			
3035	010652	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
3036	010660	011337	067264			MOV	(R3),WRBUFF	;LOAD BUFFER FOR WRITE HEADER
3037	010664	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN MAINT MODE
3038	010672	012762	000777	000020		MOV	#777,RKDCYL(R2)	;LOAD CYLINDER ADDRESS
3039	010700	012762	003400	000006		MOV	#3400,RKDA(R2)	;LOAD DISK ADDRESS
3040	010706	012762	067264	000004		MOV	#WRBUFF,RKBA(R2)	;LOAD BUS ADDRESS
3041	010714	012762	000007	000010		MOV	#7,RKCS2(R2)	;LOAD OTHER NUMBER
3042	010722	012762	177777	000002		MOV	#-1,RKWC(R2)	;LOAD WORD COUNT
3043	010730	012762	000027	000000		MOV	#WRHEAD,RKCS1(R2)	;ISSUE WRITE HEADER
3044	010736	012700	000312			MOV	#50.#4+2,RO	;ISSUE CLOCKS UNTIL READY FOR ;INDEX PULSE
3045								
3046	010742	012762	000440	000026	2\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3047	010750	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3048	010756	005300				DEC	RO	
3049	010760	001370				BNE	2\$	
3050	010762	012700	000004			MOV	#4,RO	;SIMULATE INDEX PULSE
3051	010766	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3052	010774	012762	000640	000026	3\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
3053	011002	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3054	011010	005300				DEC	RO	
3055	011012	001370				BNE	3\$	
3056	011014	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3057	011022	012700	000045			MOV	#37,RO	;SIMULATE 1 NPR TRANSFER
3058	011026	012762	000440	000026	4\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3059	011034	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3060	011042	005300				DEC	RO	
3061	011044	001370				BNE	4\$	
3062	011046	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3063	011054	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS REG
3064	011062	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
3065	011070	012700	000024			MOV	#20.,RO	;WAIT FOR OUTPUT READY
3066	011074	005300			5\$:	DEC	RO	
3067	011076	001376				BNE	5\$	
3068	011100	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3069	011106	012737	000307	003510		MOV	#OR!IR!7,E.CS2	;LOAD EXPECTED CS2
3070	011114	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
3071	011122	001401				BEG	6\$;YES, CHECK CS2
3072	011124	104035				ERROR	35	;CS1 INCORRECT
3073	011126	023737	003510	003450	6\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
3074	011134	001401				BEG	7\$;YES, CONTINUE
3075	011136	104036				ERROR	36	;CS2 INCORRECT
3076	011140	023737	003504	003444	7\$:	CMP	E.BA,T.BA	;CHECK IF BUS ADDRESS INCREMENT OCCURRED
3077	011146	001401				BEG	8\$;YES, CONTINUE
3078	011150	104037				ERROR	37	;BUS ADDRESS INCORRECT
3079	011152	023737	003502	003442	8\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT REG CORRECT
3080	011160	001401				BEG	9\$;YES, CONTINUE
3081	011162	104040				ERROR	40	;WORD COUNT INCORRECT

```

3082 011164 016237 000024 003462 9$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
3083 011172 011337 003522 MOV (R3),E.DB ;LOAD EXPECTED DATA BUFFER
3084 011176 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF DATA CORRECT
3085 011204 001401 BEQ 15$ ;YES, CONTINUE
3086 011206 104041 ERROR 41 ;DATA BUFFER INCORRECT
3087 011210 016237 000000 003440 15$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3088 011216 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3089 011224 012737 000107 003510 MOV #IR!7,E.CS2 ;LOAD EXPECTED CS2
3090 011232 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
3091 011240 001401 BEQ 17$ ;YES, CONTINUE
3092 011242 104042 ERROR 42 ;CS1 INCORRECT
3093 011244 023737 003510 003450 17$: CMP E.CS2,T.CS2 ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3094 011252 001401 BEQ 20$ ;YES, CONTINUE
3095 011254 104043 ERROR 43 ;CS2 INCORRECT
3096 011256 104415 SCOP1 ;CHECK IF LOOP ON ERROR
3097 011260 005723 TST (R3)+ ;GENERATE ADDRESS OF NEXT CONFIG
3098 011262 005304 DEC R4 ;CHECK IF ALL 8 CONFIGS TRIED
3099 011264 001000 BNE TST12 ;NO, TRY NEXT DATA PATTERN

```

```

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

```

```

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

```

```

3114 *****
3115 011266 000004 TST12: SCOPE
3116 011270 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS
3117 011276 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3118 011302 012737 000001 001160 MOV #1,$TMPO ;LOAD NUMBER OF WORDS FOR DATA TRANSFER
3119 011310 012704 000065 MOV #65,R4 ;LOAD ITERATION COUNT
3120 011314 012737 000027 003500 MOV #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
3121 011322 012737 011330 001110 MOV #1$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
; SUBTEST LOOP
3122
3123
3124 011330 1$: MOV #CCLR,RKCS1(R2) ;CLEAR RK611
3125 011330 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3126 011336 012762 000040 000026 MOV #NPRBUF,RKBA(R2) ;LOAD BUS ADDRESS
3127 011344 012762 067042 000004 MOV #NPRBUF,E.BA
3128 011352 012737 067042 003504 MOV $TMPO,E.WC ;LOAD WORD COUNT
3129 011360 013737 001160 003502 MOV E.WC
3130 011366 005437 003502 NEG E.WC,RKWC(R2)
3131 011372 013762 003502 000002 MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3132 011400 012762 000027 000000 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
3133 011406 012700 000312 ; INDEX PULSE
3134
3135 011412 012762 000440 000026 2$: MOV #DMD!MCLK,RKMR1(R2)
3136 011420 012762 000040 000026 MOV #DMD,RKMR1(R2)
3137 011426 005300 DEC R0

```

3138	011430	001370				BNE	2\$	
3139	011432	012700	000004			MOV	#4,RO	:SIMULATE INDEX PULSE
3140	011436	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
3141	011444	012762	000640	000026	3\$:	MOV	#DMD:MIND,MCLK,RKMR1(R2)	
3142	011452	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
3143	011460	005300				DEC	RO	
3144	011462	001370				BNE	3\$	
3145	011464	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3146	011472	012737	000300	003510		MOV	#IR:OR,E.CS2	:LOAD EXPECTED CS2
3147	011500	012700	000045		4\$:	MOV	#37,RO	:SIMULATE 1 NPR TRANSFER
3148	011504	012762	000440	000026	5\$:	MOV	#DMD:MCLK,RKMR1(R2)	
3149	011512	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3150	011520	005300				DEC	RO	
3151	011522	001370				BNE	5\$	
3152	011524	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3153	011532	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS
3154	011540	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3155	011546	022737	067042	003504		CMP	#NPRBUF,E.BA	:CHECK IF FIRST WORD
3156	011554	001004				BNE	7\$:NO, STORE CS2
3157	011556	012700	000024			MOV	#20.,RO	:WAIT FOR OUTPUT READY
3158	011562	005300			6\$:	DEC	RO	
3159	011564	001376				BNE	6\$	
3160	011566	016237	000010	003450	7\$:	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3161	011574	062737	000002	003504		ADD	#2,E.BA	:INCREMENT WORD COUNT AND BUS ADD
3162	011602	005237	003502			INC	E.WC	
3163	011606	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3164	011614	001401				BEQ	8\$:YES, CHECK CS2
3165	011616	104044				ERROR	44	:CS1 INCORRECT
3166	011620	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3167	011626	001401				BEQ	9\$:YES, CHECK BUSS ADDRESS REG.
3168	011630	104045				ERROR	45	:CS2 INCORRECT
3169	011632	023737	003504	003444	9\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3170	011640	001401				BEQ	10\$:YES, CHECK WORD COUNT
3171	011642	104046				ERROR	46	:BUS ADDRESS INCORRECT
3172	011644	023737	003502	003442	10\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3173	011652	001401				BEQ	11\$:YES, CHECK IF ALL WORDS TRANSFERRED
3174	011654	104047				ERROR	47	:WORD COUNT INCORRECT
3175	011656	005737	003502		11\$:	TST	E.WC	:CHECK IF FINISHED
3176	011662	001306				BNE	4\$:NO, TRANSFER NEXT WORD
3177	011664	012700	000112			MOV	#2*37,RO	:ISSUE ENOUGH CLOCKS FOR
3178	011670	012762	000440	000026	15\$:	MOV	#DMD:MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3179	011676	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3180	011704	005300				DEC	RO	
3181	011706	001370				BNE	15\$	
3182	011710	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3183	011716	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3184	011724	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3185	011732	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3186	011740	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3187	011746	001401				BEQ	16\$:YES, CHECK CS2
3188	011750	104050				ERROR	50	:CS1 INCORRECT
3189	011752	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3190	011760	001401				BEQ	17\$:YES, CHECK BUS ADDRESS
3191	011762	104051				ERROR	51	:CS2 INCORRECT
3192	011764	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3193	011772	001401				BEQ	18\$:YES, CHECK WORD COUNT

3250	012252	012762	000440	000026	1\$:	MOV	#DMD:MCLK,RKMR1(R2)	
3251	012260	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3252	012266	005300				DEC	RO	
3253	012270	001370				BNE	1\$	
3254	012272	012700	000004			MOV	#4,RO	;SIMULATE INDEX PULSE
3255	012276	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
3256	012304	012762	000640	000026	2\$:	MOV	#DMD:MIND,MCLK,RKMR1(R2)	
3257	012312	012762	000240	000026		MOV	#DMD:MIND,RKMR1(R2)	
3258	012320	005300				DEC	RO	
3259	012322	001370				BNE	2\$	
3260	012324	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3261	012332	012737	000300	003510		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
3262	012340	012700	000045		4\$:	MOV	#37,RO	;SIMULATE 1 NPR TRANSFER
3263	012344	012762	000440	000026	5\$:	MOV	#DMD:MCLK,RKMR1(R2)	
3264	012352	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3265	012360	005300				DEC	RO	
3266	012362	001370				BNE	5\$	
3267	012364	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3268	012372	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS
3269	012400	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
3270	012406	022737	067044	003504		CMP	#NPRBUF+2,E.BA	;CHECK IF FIRST WORD
3271	012414	001004				BNE	7\$;NO STORE CS2
3272	012416	012700	000024			MOV	#20.,RO	;WAIT FOR OUTPUT READY
3273	012422	005300			6\$:	DEC	RO	
3274	012424	001376				BNE	6\$	
3275	012426	016237	000010	003450	7\$:	MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3276	012434	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK COMMAND STATUS REG. 1 CORRECT
3277	012442	001401				BEQ	8\$;YES, CHECK CS2
3278	012444	104044				ERROR	44	;CS1 INCORRECT
3279	012446	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	;CHECK COMMAND STATUS REG. 2 CORRECT
3280	012454	001401				BEQ	9\$;YES, CHECK BUS ADDRESS
3281	012456	104045				ERROR	45	;CS2 INCORRECT
3282	012460	023737	003504	003444	9\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
3283	012466	001401				BEQ	10\$;YES, CHECK WORD COUNT
3284	012470	104046				ERROR	46	;BUS ADDRESS INCORRECT
3285	012472	023737	003502	003442	10\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
3286	012500	001401				BEQ	11\$;YES, CHECK IF ALL WORDS TRANSFERRED
3287	012502	104047				ERROR	47	;WORD COUNT INCORRECT
3288	012504	062737	000002	003504	11\$:	ADD	#2,E.BA	;INCREMENT WORD COUNT AND BUS ADDRESS
3289	012512	005237	003502			INC	E.WC	
3290	012516	100710				BMI	4\$;CHECK IF FINISHED (NO, BRANCH)
3291	012520	001004				BNE	12\$;CHECK IF LAST WORD
3292	012522	012737	000200	003510		MOV	#OR,E.CS2	;LOAD EXPECTED CS2
3293	012530	000703				BR	4\$;PROCESS LAST WORD
3294								
3295	012532	005037	003502		12\$:	CLR	E.WC	;ADJUST EXPECTED WORD COUNT
3296	012536	162737	000002	003504		SUB	#2,E.BA	;AND BUS ADDRESS
3297	012544	012700	000112			MOV	#2*37,RO	;ISSUE ENOUGH CLOCKS FOR
3298	012550	012762	000440	000026	15\$:	MOV	#DMD:MCLK,RKMR1(R2)	; 2 NPR TRANSFERS
3299	012556	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3300	012564	005300				DEC	RO	
3301	012566	001370				BNE	15\$	
3302	012570	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
3303	012576	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
3304	012604	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS REG
3305	012612	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT

M05

CZR6CCO RK611 DSKLS CTRL PRT3 MACY11 30(1046) 02-DEC-77 09:56 PAGE 64
 CZR6CC.P11 02-DEC-77 09:46 T13 SILO FILLING WITH NPR TRANSFERS

SEQ 0064

3306	012620	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3307	012626	001401				BEQ	16\$:YES, CHECK CS2
3308	012630	104050				ERROR	50	:CS1 INCORRECT
3309	012632	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3310	012640	001401				BEQ	17\$:YES, CHECK BUS ADDRESS
3311	012642	104051				ERROR	51	:CS2 INCORRECT
3312	012644	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3313	012652	001401				BEQ	18\$:YES, CHECK WORD COUNT
3314	012654	104052				ERROR	52	:BUS ADDRESS INCORRECT
3315	012656	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3316	012664	001401				BEQ	19\$:YES, CHECK DATA
3317	012666	104053				ERROR	53	:WORD COUNT INCORRECT
3318	012670	012701	000102		19\$:	MOV	#66,R1	:LOAD NUMBER OF WORDS LOADED
3319	012674	012703	067042			MOV	#NPRBUF,R3	:LOAD START ADDRESS
3320	012700	005037	003624			CLR	WRDCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3321	012704	012737	000300	003510		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3322	012712	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3323	012720	012337	003522			MOV	(R3)+,E.DB	:LOAD EXPECTED DATA BUFFER
3324	012724	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3325	012732	001401				BEQ	26\$:YES, CONTINUE
3326	012734	104054				ERROR	54	:DATA BUFFER INCORRECT
3327	012736	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
3328	012744	005737	003624			TST	WRDCNT	:CHECK IF FIRST WORD
3329	012750	001015				BNE	28\$:NO, GET CS2
3330	012752	012700	000024			MOV	#20.,R0	:WAIT FOR INPUT READY TO SET
3331	012756	005300			27\$:	DEC	R0	
3332	012760	001376				BNE	27\$	
3333	012762	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3334	012770	022701	000001			CMP	#1 R1	:CHECK IF LAST WORD IN SILO
3335	012774	001003				BNE	28\$:NO, CONTINUE
3336	012776	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3337	013004	023737	003500	003440	28\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3338	013012	001401				BEQ	29\$:YES, CHECK CS2
3339	013014	104055				ERROR	55	:CS1 INCORRECT
3340	013016	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND AND STATUS REG. 2 CORRECT
3341	013024	001401				BEQ	30\$:YES, GET NEXT WORD
3342	013026	104056				ERROR	56	:CS2 INCORRECT
3343	013030	005237	003624		30\$:	INC	WRDCNT	:INCREMENT WORD COUNT
3344	013034	005301				DEC	R1	:DECREMENT WORDS READ
3345	013036	001325				BNE	25\$:CHECK IF ALL WORDS READ

```

3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
*****
*TEST 14          SILO CAPICITY WITH NPR TRANSFERS
*
* CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A 68 WORD DATA TRANSFER.  CLOCK IN 66
* WORDS INTO THE SILO.  MAKE SURE THAT SILO WILL STOP
* FILLING AT 66 WORDS.  TAKE ONE WORD FROM SILO AND
* CHECK IT.  CLOCK IN NEXT WORD.  MAKE SURE NO MORE
* THAN ONE WORD IS CLOCKED IN THE SILO.  TAKE ONE
* WORD FROM SILO AND CHECK IT.  CLOCK IN NEXT WORD.
* CLOCK IN NEXT WORD.  MAKE SURE NO MORE THAN ONE WORD IS
* CLOCKED IN THE SILO.  TAKE ONE WORD FROM SILO AND
* CHECK IT.  ATTEMPT TO CLOCK IN NEXT WORD AND

```

N05

CZR6CCD RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 65
T14 SILO CAPACITY WITH NPR TRANSFERS

SEQ 0065

```

3362      ;*      MAKE SURE NO WORDS ARE CLOCK INTO SILO.  UNLOAD THE
3363      ;*      SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.
3364      ;*
3365      ;*****
3366      013040 000004      †ST14: SCOPE
3367      013042 012737 000144 001200      MOV      #100.,$TIMES      ;;DO 100. ITERATIONS
3368      013050 013702 001270      MOV      $BASE,R2      ;;LOAD RK611 BASE
3369      013054 012737 000027 003500      MOV      #WRHEAD,E.CS1  ;;LOAD EXPECTED CS1
3370      013062 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
3371      013070 012762 000040 000026      MOV      #DMD,RKMR1(R2)  ;;PUT RK611 IN MAINT MODE
3372      013076 012737 067044 003504      MOV      #NPRBUF+2,E.BA  ;;LOAD BUS ADDRESS
3373      013104 012762 067042 000004      MOV      #NPRBUF,RKBA(R2)
3374      013112 012737 177675 003502      MOV      #-67.,E.WC      ;;LOAD WORD COUNT
3375      013120 012762 177674 000002      MOV      #-68.,RKWC(R2)
3376      013126 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ;;ISSUE WRITE HEADER
3377      013134 012700 000312      MOV      #50.#4+2,R0      ;;ISSUE CLOCKS UNTIL READY FOR
3378      ;          INDEX PULSE
3379      013140 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
3380      013146 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3381      013154 005300      DEC      R0
3382      013156 001370      BNE     1$
3383      013160 012700 000004      MOV      #4,R0      ;;SIMULATE INDEX PULSE
3384      013164 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3385      013172 012762 000640 000026 2$:      MOV      #DMD!MIND!MCLK,RKMR1(R2)
3386      013200 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3387      013206 005300      DEC      R0
3388      013210 001370      BNE     2$
3389      013212 012737 000040 000026      MOV      #DMD,RKMR1
3390      013220 012737 000300 003510      MOV      #IR!OR,E.CS2      ;;LOAD EXPECTED CS2
3391      013226 012700 000045      MOV      #37.,R0      ;;SIMULATE 1 NPR TRANSFER
3392      013232 012762 000440 000026 4$:      MOV      #DMD!MCLK,RKMR1(R2)
3393      013240 012762 000040 000026 5$:      MOV      #DMD,RKMR1(R2)
3394      013246 005300      DEC      R0
3395      013250 001370      BNE     5$
3396      013252 016237 000000 003440      MOV      RKCS1(R2),T.CS1  ;;STORE COMMAND AND REG. 1
3397      013260 016237 000004 003444      MOV      RKBA(R2),T.BA    ;;STORE BUS ADDRESS
3398      013266 016237 000002 003442      MOV      RKWC(R2),T.WC    ;;STORE WORD COUNT
3399      013274 022737 067044 003504      CMP      #NPRBUF+2,E.BA  ;;CHECK IF FIRST WORD
3400      013302 001004      BNE     7$      ;;NO, STORE CS2
3401      013304 012700 000024      MOV      #20.,R0      ;;WAIT FOR OUTPUT READY
3402      013310 005300      DEC      R0
3403      013312 001376      BNE     6$
3404      013314 016237 000010 003450 7$:      MOV      RKCS2(R2),T.CS2  ;;STORE COMMAND AND STATUS REG. 2
3405      013322 023737 003500 003440      CMP      E.CS1,T.CS1      ;;CHECK COMMAND STATUS REG. 1 CORRECT
3406      013330 001401      BEQ     8$      ;;YES, CHECK CS2
3407      013332 104044      ERROR   44      ;;CS1 INCORRECT
3408      013334 023737 003510 003450 8$:      CMP      E.CS2,T.CS2      ;;CHECK COMMAND STATUS REG. 2 CORRECT
3409      013342 001401      BEQ     9$      ;;YES, CHECK BUS ADDRESS
3410      013344 104045      ERROR   45      ;;CS2 INCORRECT
3411      013346 023737 003504 003444 9$:      CMP      E.BA,T.BA      ;;CHECK BUS ADDRESS CORRECT
3412      013354 001401      BEQ    10$      ;;YES, CHECK WORD COUNT
3413      013356 104046      ERROR   46      ;;BUS ADDRESS INCORRECT
3414      013360 023737 003502 003442 10$:      CMP      E.WC,T.WC      ;;CHECK WORD COUNT CORRECT
3415      013366 001401      BEQ    11$      ;;YES, CHECK IF 1ST 65 WORDS TRANSFERRED
3416      013370 104047      ERROR   47      ;;WORD COUNT INCORRECT
3417      013372 062737 000002 003504 11$:      ADD      #2,E.BA      ;;INCREMENT BUS ADD AND WORD COUNT
    
```



```

3474 013732 001410 BEQ 32$ ;YES, NO NPR WILL TAKE PLACE
3475 013734 062737 000002 003504 ADD #2, E.BA ;INCREMENT BUS ADD AND WORD COUNT
3476 013742 005237 003502 INC E.WC
3477 013746 012737 000200 003510 MOV #0R, E.CS2 ;LOAD EXPECTED CS2
3478 013754 016237 000000 003440 32$: MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
3479 013762 016237 000010 003450 MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
3480 013770 016237 000004 003444 MOV RKBA(R2), T.BA ;STORE BUS ADDRESS
3481 013776 016237 000002 003442 MOV RKWC(R2), T.WC ;STORE WORD COUNT
3482 014004 023737 003500 003440 CMP E.CS1, T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3483 014012 001401 BEQ 33$ ;YES, CHECK CS2
3484 014014 104044 ERROR 44 ;CS1 INCORRECT
3485 014016 023737 003510 003450 33$: CMP E.CS2, T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3486 014024 001401 BEQ 34$ ;YES, CHECK BUS ADD
3487 014026 104045 ERROR 45 ;CS2 INCORRECT
3488 014030 023737 003504 003444 34$: CMP E.BA, T.BA ;CHECK BUS ADD CORRECT
3489 014036 001401 BEQ 35$ ;YES, CHECK WORD COUNT
3490 014040 104046 ERROR 46 ;BUS ADD INCORRECT
3491 014042 023737 003502 003442 35$: CMP E.WC, T.WC ;CHECK WORD COUNT CORRECT
3492 014050 001401 BEQ 36$ ;YES, CONTINUE
3493 014052 104047 ERROR 47 ;WORD COUNT INCORRECT
3494 014054 005237 003624 36$: INC WRDCNT
3495 014060 005301 DEC R1 ;CHECK IF READ TO UNLOAD SILO
3496 014062 001402 BEQ 39$ ;YES, READ DATA; BUFFER
3497 014064 000137 013454 JMP 13$ ;NO, INPUT NEXT WORD
3498
3499 014070 162737 000002 003504 39$: SUB #2, E.BA ;ADJUST WORD COUNT AND BUS ADDRESS
3500 014076 005037 003502 CLR E.WC
3501 014102 012737 000300 003510 MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
3502 014110 012701 000101 MOV #65, R1 ;LOAD NUMBER OF WORDS LEFT
3503 014114 016237 000024 003462 40$: MOV RKDB(R2), T.DB ;READ DATA BUFFER
3504 014122 012337 003522 MOV (R3)+, E.DB ;LOAD EXPECTED DATA BUFFER
3505 014126 023737 003522 003462 CMP E.DB, T.DB ;CHECK IF DATA CORRECT
3506 014134 001401 BEQ 41$ ;YES, CHECK CS1
3507 014136 104054 ERROR 54 ;DATA BUFFER INCORRECT
3508 014140 016237 000000 003440 41$: MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
3509 014146 022737 000002 003624 CMP #2, WRDCNT ;CHECK IF FIRST WORDS
3510 014154 001004 BNE 43$ ;NO, DO NOT WAIT FOR INPUT READY
3511 014156 012700 000024 MOV #20., R0 ;WAIT FOR INPUT READY
3512 014162 005300 42$: DEC R0
3513 014164 001376 BNE 42$
3514 014166 016237 000010 003450 43$: MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
3515 014174 022701 000001 CMP #1, R1 ;CHECK IF LAST WORD
3516 014200 001003 BNE 44$ ;NO, CONTINUE
3517 014202 012737 000100 003510 MOV #IR, E.CS2 ;LOAD EXPECTED CS2
3518 014210 023737 003500 003440 44$: CMP E.CS1, T.CS1 ;CHECK CS1 CORRECT
3519 014216 001401 BEQ 45$ ;YES, CHECK CS2
3520 014220 104055 ERROR 55 ;CS1 INCORRECT
3521 014222 023737 003510 003450 45$: CMP E.CS2, T.CS2 ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3522 014230 001401 BEQ 46$ ;YES, READ NEXT WORD ON SILO
3523 014232 104056 ERROR 56 ;CS2 INCORRECT
3524 014234 005237 003624 46$: INC WRDCNT ;INCREMENT WORD COUNT
3525 014240 005301 DEC R1 ;CHECK IF ALL WORDS READ
3526 014242 001324 BNE 40$ ;NO, READ NEXT WORD

```

```

3527
3528 ;*****
3529 ;*TEST 15 BUS ADDRESS INHIBIT

```

3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585

014244 000004
014246 012737 000144 001200
014254 013702 001270
014260 012737 000027 003500
014266 012762 100000 000000
014274 012762 000040 000026
014302 012737 067042 003504
014310 012762 067042 000004
014316 012737 177677 003502
014324 012762 177676 000002
014332 012762 000020 000010
014340 012762 000027 000000
014346 012700 000312

014352 012762 000440 000026 1\$:
014360 012762 000040 000026
014366 005300
014370 001370
014372 012700 000004
014376 012762 000240 000026
014404 012762 000640 000026 2\$:
014412 012762 000240 000026
014420 005300
014422 001370
014424 012762 000040 000026
014432 012737 000320 003510
014440 012700 000045 4\$:
014444 012762 000440 000026 5\$:
014452 012762 000040 000026
014460 005300
014462 001370
014464 016237 000000 003440
014472 016237 000004 003444
014500 016237 000002 003442
014506 022737 000101 003502
014514 001004
014516 012700 000024
014522 005300 6\$:
014524 001376
014526 016237 000010 003450 7\$:
014534 023737 003500 003440
014542 001401
014544 104057
014546 023737 003510 003450 8\$:
014554 001401
014556 104060

```
*****
: CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
: IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN
: RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
: SPECIFY A 66 WORD DATA TRANSFER WITH BUS ADDRESS
: INHIBIT INCREMENT. CHECK WORD COUNT, BUS ADDRESS,
: INPUT READY, OUTPUT READY, AND MAKE SURE ALL THE
: WORDS IN THE SILO ARE THE CORRECT SAME WORD.
*****
TST15: SCOPE
MOV #100.,STIMES ;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
MOV #NPRBUF,E.BA ;LOAD BUS ADDRESS
MOV #NPRBUF,RKBA(R2)
MOV #-65.,E.WC ;LOAD WORD COUNT
MOV #-66.,RKWC(R2)
MOV #BAI,RKCS2(R2) ;SET BUS ADDRESS INCREMENT INHIBIT
MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
MOV #50.#4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
; INDEX PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
R0
DEC
BNE 1$
MOV #4,R0 ;SIMULATE INDEX PULSE
MOV #DMD!MIND,RKMR1(R2)
2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
R0
DEC
BNE 2$
MOV #DMD,RKMR1(R2)
MOV #IR!OR!BAI,E.CS2 ;LOAD EXPECTED CS2
3$: MOV #37.,R0 ;SIMULATE 1 NPR TRANSFER
5$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
R0
DEC
BNE 5$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
MOV RKWC(R2),T.WC ;STORE WORD COUNT
CMP #65.,E.WC ;CHECK IF FIRST WORD
BNE 7$ ;NO, STORE CS2
MOV #20.,R0 ;WAIT FOR OUTPUT READY
6$: DEC
R0
BNE 6$
7$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
BEQ 8$ ;YES, CHECK CS2
ERROR 57 ;CS1 INCORRECT
8$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
BEQ 9$ ;YES, CHECK BUS ADDRESS
ERROR 60 ;CS2 INCORRECT
```

3586	014560	023737	003504	003444	9\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS
3587	014566	001401				BEQ	10\$:YES, CHECK WORD COUNT
3588	014570	104061				ERROR	61	:BUS ADDRESS INCORRECT
3589	014572	023737	003502	003442	10\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3590	014600	001401				BEQ	11\$:YES, CHECK IF ALL WORDS TRANSFERRED
3591	014602	104062				ERROR	62	:WORD COUNT INCORRECT
3592	014604	005237	003502		11\$:	INC	E.WC	:INCREMENT WORD COUNT
3593	014610	100713				BMI	4\$:CHECK IF FINISHED (NO, BRANCH)
3594	014612	001004				BNE	12\$:CHECK IF LAST WORD
3595	014614	012737	000220	003510		MOV	#OR!BAI,E.CS2	:LOAD EXPECTED COMMAND STATUS REG. 2
3596	014622	000706				BR	4\$:PROCESS THE LAST WORD
3597								
3598	014624	005037	003502		12\$:	CLR	E.WC	:ADJUST WORD COUNT
3599	014630	012700	000112			MOV	#2*37.,RO	:ISSUE ENOUGH CLOCKS FOR
3600	014634	012762	000440	000026	15\$:	MOV	#DMD!MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3601	014642	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3602	014650	005300				DEC	RO	
3603	014652	001370				BNE	15\$	
3604	014654	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3605	014662	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3606	014670	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3607	014676	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
3608	014704	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3609	014712	001401				BEQ	16\$:YES, CHECK CS2
3610	014714	104063				ERROR	63	:CS1 INCORRECT
3611	014716	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3612	014724	001401				BEQ	17\$:YES, CHECK BUS ADDRESS
3613	014726	104064				ERROR	64	:CS2 INCORRECT
3614	014730	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3615	014736	001401				BEQ	18\$:YES, CHECK WORD COUNT
3616	014740	104065				ERROR	65	:BUS ADDRESS INCORRECT
3617	014742	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3618	014750	001401				BEQ	19\$:YES, CHECK DATA
3619	014752	104066				ERROR	66	:WORD COUNT INCORRECT
3620	014754	012701	000102		19\$:	MOV	#66.,R1	:LOAD NUMBERS OF WORDS LOADED
3621	014760	013737	067042	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3622	014766	005037	003624			CLR	WRDCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3623	014772	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3624	015000	012737	000320	003510		MOV	#IR!OR!BAI,E.CS2	:LOAD EXPECTED CS2
3625	015006	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3626	015014	001401				BEQ	26\$:YES, CONTINUE
3627	015016	104067				ERROR	67	:DATA BUFFER INCORRECT
3628	015020	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3629	015026	005737	003624			TST	WRDCNT	:CHECK IF FIRST WORD
3630	015032	001015				BNE	28\$:NO, GET CS2
3631	015034	012700	000024			MOV	#20.,RO	:WAIT FOR INPUT READY TO SET
3632	015040	005300			27\$:	DEC	RO	
3633	015042	001376				BNE	27\$	
3634	015044	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3635	015052	022701	000001			CMP	#1,R1	:CHECK IF LAST WORD
3636	015056	001003				BNE	28\$:NO, CONTINUE
3637	015060	012737	000120	003510		MOV	#IR!BAI,E.CS2	:LOAD EXPECTED CS2
3638	015066	023737	003500	003440	28\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3639	015074	001401				BEQ	29\$:YES, CHECK CS2
3640	015076	104070				ERROR	70	:CS1 INCORRECT
3641	015100	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG 2 CORRECT

3642 015106 001401
3643 015110 104071
3644 015112 005237 003624
3645 015116 005301
3646 015120 001324

BEQ 30\$;YES, GET NEXT WORD
ERROR 71 ;CS2 INCORRECT
30\$: INC WRDCNT ;INCREMENT WORD COUNT
DEC R1 ;DECREMENT WORDS READ
BNE 25\$;CHECK IF ALL WORDS READ

*TEST 16 NON-EXISTENT MEMORY
*
* CLEAR RK611 WITH A CONTROLLER CLEAR, PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS
* (760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR
* OCCURS IN THE RK611 CONTROLLER.

3658
3659 015122 000004
3660 015124 012737 000144 001200
3661 015132 013702 001270
3662 015136 012762 100000 000000
3663 015144 012762 000040 000026
3664 015152 012737 160002 003504
3665 015160 012762 160000 000004
3666 015166 012737 177677 003502
3667 015174 012762 177676 000002
3668 015202 012737 101626 003500
3669 015210 012762 001427 000000
3670 015216 012700 000312
3671
3672 015222 012762 000440 000026 1\$:
3673 015230 012762 000040 000026
3674 015236 005300
3675 015240 001370
3676 015242 012700 000004
3677 015246 012762 000240 000026
3678 015254 012762 000640 000026 2\$:
3679 015262 012762 000240 000026
3680 015270 005300
3681 015272 001370
3682 015274 012762 000040 000026
3683 015302 012700 000045
3684 015306 012762 000440 000026 5\$:
3685 015314 012762 000040 000026
3686 015322 005300
3687 015324 001370
3688 015326 016237 000000 003440
3689 015334 016237 000010 003450
3690 015342 016237 000004 003444
3691 015350 016237 000002 003442
3692 015356 016237 000014 003454
3693 015364 005037 003514
3694 015370 012737 004100 003510
3695 015376 032737 000200 003450
3696 015404 001403
3697 015406 052737 000200 003510

↑ST16: SCOPE
MOV #100, \$TIMES ;DO 100. ITERATIONS
MOV \$BASE, R2 ;LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;CLEAR RK611
MOV #DMD, RKMR1(R2) ;PUT RK611 IN MAINT MODE
MOV #160002, E.BA ;LOAD BUS ADDRESS
MOV #160000, RKBA(R2)
MOV #-65, E.WC ;LOAD WORD COUNT
MOV #-66, RKWC(R2)
MOV #CERR!RDY!BA16!BA17!WRHEAD<↑C<GO>>, E.CS1
MOV #BA17!BA16!WRHEAD, RKCS1(R2) ;ISSUE WRITE HEADER
MOV #50.*4+2, R0 ;ISSUE CLOCKS UNTIL READY FOR
; INDEX PULSE
1\$: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
1\$
MOV #4, R0 ;SIMULATE INDEX PULSE
2\$: MOV #DMD!MIND, RKMR1(R2)
MOV #DMD!MIND!MCLK, RKMR1(R2)
MOV #DMD!MIND, RKMR1(R2)
DEC R0
2\$
MOV #DMD, RKMR1(R2)
MOV #37, R0 ;SIMULATE 1 NPR TRANSFER
5\$: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
5\$
MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
MOV RKBA(R2), T.BA ;STORE BUS ADDRESS
MOV RKWC(R2), T.WC ;STORE WORD COUNT
MOV RKER(R2), T.ER ;STORE ERROR REG.
CLR E.ER ;LOAD EXPECTED ERROR REG.
MOV #IR!NEM, E.CS2 ;LOAD EXPECTED CS2
BIT #OR, T.CS2
BEQ 7\$
BIS #OR, E.CS2

3698	015414	023737	003500	003440	7\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3699	015422	001401				BEQ	8\$:YES, CHECK CS2
3700	015424	104072				ERROR	72	:CS1 INCORRECT
3701	015426	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3702	015434	001401				BEQ	9\$:YES, CHECK ERROR REG.
3703	015436	104073				ERROR	73	:CS2 INCORRECT
3704	015440	023737	003514	003454	9\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3705	015446	001401				BEQ	10\$:CHECK BUS ADDRESS
3706	015450	104074				ERROR	74	:ERROR REG INCORRECT
3707	015452	023737	003504	003444	10\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3708	015460	001401				BEQ	11\$:YES, CHECK WORD COUNT
3709	015462	104075				ERROR	75	:BUS ADDRESS INCORRECT
3710	015464	023737	003502	003442	11\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT REG.
3711	015472	001401				BEQ	12\$:YES, CLEAR RK611
3712	015474	104076				ERROR	76	:WORD COUNT INCORRECT
3713	015476	012762	100000	000000	12\$:	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3714	015504	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3715	015512	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3716	015520	012737	000200	003500		MOV	#RDY,E.CS1	:LOAD EXPECTED CS1
3717	015526	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3718	015534	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1
3719	015542	001401				BEQ	15\$:YES, CHECK CS2
3720	015544	104077				ERROR	77	:CS1 INCORRECT
3721	015546	023737	003510	003450	15\$:	CMP	E.CS2,T.CS2	:CHECK IF NEM CLEARED
3722	015554	001401				BEQ	TST17	:YES, GO ON TO NEXT TEST
3723	015556	104100				ERROR	100	:CS2 INCORRECT

*TEST 17 BUS ADDRESS BIT 16

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* SPECIFY A ONE WORD DATA TRANSFER FROM 200000.
* READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
* REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.
* CHECK BUS ADDRESS AND WORD COUNT.
*
* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
* OF MEMORY IS ON THE SYSTEM.
*

3740	015560	000004				TST17:	SCOPE	
3741	015562	012737	000144	001200		MOV	#100,\$TIMES	:DO 100. ITERATIONS
3742	015570	005737	046374			TST	\$KT11	:CHECK FOR MEMORY MANAGEMENT
3743	015574	100004				BPL	1\$:NO, BYPASS TEST
3744	015576	022737	002000	046642		CMP	#2000,\$LSTBK	:CHECK IF ENOUGH MEMORY
3745	015604	103417				BLO	2\$:YES, DO TEST
3746	015606				1\$:			
3747	015606	012737	000001	001200		MOV	#1 \$TIMES	:FORCE INTERATION COUNT TO 1
3748	015614	005227	177777			INC	#-1	:ONLY DO ONCE
3749	015620	001007				BNE	64\$:NO, GO TO NEXT TEST
3750	015622	104401	055013			TYPE	TSTBY1	:TYPE TEST N BYPASSED
3751	015626	013746	001220			MOV	\$TESTN,-(SP)	:SAVE \$TESTN FOR TYPEOUT
3752	015632	104402				TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
3753	015634	104401	055023			TYPE	,TSTBY2	

```

3754 015640 000137 016674          64$: JMP      TST20          ;GO TO NEXT TEST
3755
3756 015644 013702 001270          2$:  MOV      $BASE,R2          ;LOAD K611 BASE
3757 015650 012737 002000 172354  MOV      #2000,KIPAR6        ;LOAD PAGE ADDRESS 6 FOR DATA
3758 015656 005237 177572          INC      SR0                  ;TURN ON MEMORY MANAGEMENT
3759 015662 013737 067042 140000  MOV      NPRBUF,140000        ;LOAD WORD IN MEMORY
3760 015670 005037 177572          CLR      SR0                  ;TURN OFF MEMORY MANAGEMENT
3761 015674 012737 015702 001110  MOV      #3$,SLPERR          ;LOAD LOOP ON ERROR LOCATION FOR
3762                                     ; SUBTEST LOOP
3763
3764 015702          3$:  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3765 015702 012762 100000 000000  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3766 015710 012762 000040 000026  MOV      #-1,RKWC(R2)
3767 015716 012762 177777 000002  MOV      #BA16:WRHEAD,E.CS1 ;LOAD COMMAND
3768 015724 012737 000427 003500  MOV      #BA16:WRHEAD,RKCS1(R2)
3769 015732 012762 000427 000000  MOV      #50,#4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
3770 015740 012700 000312          MOV      #DMD:MCLK,RKMR1(R2) ; INDEX PULSE
3771 015744 012762 000440 000026  5$:  MOV      #DMD,RKMR1(R2)
3772 015752 012762 000040 000026  DEC      R0
3773 015760 005300          BNE     5$
3774 015762 001370          MOV      #4,R0 ;SIMULATE INDEX PULSE
3775 015764 012700 000004          MOV      #DMD:MIND,RKMR1(R2)
3776 015770 012762 000240 000026  6$:  MOV      #DMD:MIND:MCLK,RKMR1(R2)
3777 015776 012762 000640 000026  MOV      #DMD:MIND,RKMR1(R2)
3778 016004 012762 000240 000026  DEC      R0
3779 016012 005300          BNE     6$
3780 016014 001370          MOV      #DMD,RKMR1(R2)
3781 016016 012762 000040 000026  MOV      #37,R0 ;ISSUE 1 NPR TRANSFER
3782 016024 012700 000045          MOV      #DMD:MCLK,RKMR1(R2)
3783 016030 012762 000440 000026  7$:  MOV      #DMD,RKMR1(R2)
3784 016036 012762 000040 000026  DEC      R0
3785 016044 005300          BNE     7$
3786 016046 001370          MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3787 016050 016237 000000 003440  MOV      RKBA(R2),T.BA ;STORE BUS ADDRESS
3788 016056 016237 000004 003444  MOV      RKWC(R2),T.WC ;STORE WOR
3789 016064 016237 000002 003442  MOV      #20.,R0 ;WAIT FOR OUTPUT READY
3790 016072 012700 000024          8$:  DEC      R0
3791 016076 005300          BNE     8$
3792 016100 001376          MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3793 016102 016237 000010 003450  MOV      #IR:OR,E.CS2 ;LOAD EXPECTED CS2
3794 016110 012737 000300 003510  MOV      #2,E.BA ;LOAD EXPECTED BUS ADDRESS
3795 016116 012737 000002 003504  CLR      E.WC ;LOAD EXPECTED WORD COUNT
3796 016124 005037 003502          CMP     E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3797 016130 023737 003500 003440  BEQ     10$ ;YES, CHECK CS2
3798 016136 001401          ERROR  101 ;CS1 INCORRECT
3799 016140 104101          CMP     E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
3800 016142 023737 003510 003450  10$: BEQ     11$ ;YES, CHECK BUS ADDRESS CORRECT
3801 016150 001401          ERROR  102 ;CS2 INCORRECT
3802 016152 104102          CMP     E.BA,T.BA ;CHECK IF BUS ADD INCORRECT
3803 016154 023737 003504 003444  11$: BEQ     12$ ;YES, CHECK WORD COUNT CORRECT
3804 016162 001401          ERROR  103 ;BUS ADDRESS INCORRECT
3805 016164 104103          CMP     E.WC,T.WC ;CHECK IF WORD COUNT CORRECT
3806 016166 023737 003502 003442  12$: BEQ     13$ ;YES, CHECK DATA BUFFER
3807 016174 001401          ERROR  104 ;WORD COUNT INCORRECT
3808 016176 104104          MOV      RKDB(R2),T.DB ;READ DATA BUFFER
3809 016200 016237 000024 003462  13$:

```

3810	016206	013737	067042	003522		MOV	NPRBUF, E.DB	; LOAD EXPECTED DATA BUFFER
3811	016214	023737	003522	003462		CMP	E.DB, T.DB	; CHECK IF DATA CORRECT
3812	016222	001401				BEQ	15\$; YES, CHECK IF LOOP ON ERROR
3813	016224	104105				ERROR	105	; DATA INCORRECT
3814	016226	104415			15\$:	SCOP1		; CHECK IF LOOP ON ERROR
3815	016230	012737	001777	172354		MOV	#2000-1, KIPAR6	; LOAD PAGE ADDRESS 6 FOR DATA
3816	016236	005237	177572			INC	SRO	; TURN ON MEMORY MANAGEMENT
3817	016242	013737	067042	140076		MOV	NPRBUF, 140076	; LOAD WORDS IN MEMORY
3818	016250	013737	067044	140100		MOV	NPRBUF+2, 140100	
3819	016256	005037	177572			CLR	SRO	; TURN OFF MEMORY MANAGEMENT
3820	016262	012737	016270	001110		MOV	#20\$, \$LPERR	; LOAD LOOP ON ERROR LOCATION FOR
3821								; SUBTEST LOOP
3822								
3823	016270				20\$:			
3824	016270	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	; CLEAR RK611
3825	016276	012762	000040	000026		MOV	#DMD, RKMR1(R2)	; PUT RK611 IN DIAGNOSTIC MODE
3826	016304	012737	177777	003502		MOV	#-1, E.WC	; LOAD WORD COUNT REG.
3827	016312	012762	177776	000002		MOV	#-2, RKWC(R2)	
3828	016320	005037	003504			CLR	E.BA	; LOAD BUS ADDRESS
3829	016324	012762	177776	000004		MOV	#177776, RKBA(R2)	
3830	016332	012737	000427	003500		MOV	#BA16!WRHEAD, E.CS1	; LOAD COMMAND
3831	016340	012762	000027	000000		MOV	#WRHEAD, RKCS1(R2)	
3832	016346	012700	000312			MOV	#50, #4+2, RO	; ISSUE ENOUGH CLOCKS UNTIL
3833	016352	012762	000440	000026	21\$:	MOV	#DMD!MCLK, RKMR1(R2)	; INDEX PULSE
3834	016360	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3835	016366	005300				DEC	RO	
3836	016370	001370				BNE	21\$	
3837	016372	012700	000004			MOV	#4, RO	; SIMULATE INDEX PULSE
3838	016376	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3839	016404	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
3840	016412	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
3841	016420	005300				DEC	RO	
3842	016422	001370				BNE	22\$	
3843	016424	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3844	016432	012701	000002			MOV	#2, R1	; ISSUE 2 NPR TRANSFERS
3845	016436	012700	000045		23\$:	MOV	#37, RO	
3846	016442	012762	000440	000026	24\$:	MOV	#DMD!MCLK, RKMR1(R2)	
3847	016450	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
3848	016456	005300				DEC	RO	
3849	016460	001370				BNE	24\$	
3850	016462	016237	000000	003440		MOV	RKCS1(R2), T.CS1	; STORE COMMAND AND STATUS REG. 1
3851	016470	016237	000004	003444		MOV	RKBA(R2), T.BA	; STORE BUS ADDRESS REG.
3852	016476	016237	000002	003442		MOV	RKWC(R2), T.WC	; STORE WORD COUNT
3853	016504	005737	003502			TST	E.WC	; CHECK IF FIRST WORD
3854	016510	001404				BEQ	26\$; NO, GET CS2
3855	016512	012700	000024			MOV	#20., RO	; WAIT FOR OUTPUT READY
3856	016516	005300			25\$:	DEC	RO	
3857	016520	001376				BNE	25\$	
3858	016522	016237	000010	003450	26\$:	MOV	RKCS2(R2), T.CS2	; STORE COMMAND AND STATUS REG 2
3859	016530	012737	000300	003510		MOV	#IR!OR, E.CS2	; LOAD EXPECTED CS2
3860	016536	023737	003500	003440		CMP	E.CS1, T.CS1	; CHECK COMMAND STATUS REG 1 CORRECT
3861	016544	001401				BEQ	28\$; YES, CHECK CS2
3862	016546	104101				ERROR	101	; CS1 INCORRECT
3863	016550	023737	003510	003450	28\$:	CMP	E.CS2, T.CS2	; CHECK COMMAND STATUS REG 2 CORRECT
3864	016556	001401				BEQ	29\$; YES, CHECK BUS ADDRESS
3865	016560	104102				ERROR	102	; CS2 INCORRECT

3866	016562	023737	003504	003444	29\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
3867	016570	001401				BEQ	30\$;YES, CHECK WORD COUNT
3868	016572	104103				ERROR	103	;BUS ADDRESS INCORRECT
3869	016574	023737	003502	003442	30\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
3870	016602	001401				BEQ	31\$;YES, GET TEXT WORD
3871	016604	104104				ERROR	104	;WORD COUNT INCORRECT
3872	016606	062737	000002	003504	31\$:	ADD	#2,E.BA	;INCREMENT BUS ADDRESS AND
3873	016614	005237	003502			INC	E.WC	;WORD COUNT
3874	016620	005301				DEC	R1	;CHECK IF FINISHED
3875	016622	001305				BNE	23\$;NO, GET SECOND WORD
3876	016624	012700	000002			MOV	#2,RO	;LOAD COUNT AND ADDRESS WORD
3877	016630	012703	067042			MOV	#NPRBUF,R3	;DATA COMPARE
3878	016634	016237	000024	003462	35\$:	MOV	RKDB(R2),T.DB	;READ DATA BUFFER
3879	016642	012337	003522			MOV	(R3)+,E.DB	;GET EXPECTED DATA
3880	016646	023737	003522	003462		CMP	E.DB,T.DB	;CHECK IF CORRECT
3881	016654	001401				BEQ	36\$;YES, CHECK IF FINISHED
3882	016656	104105				ERROR	105	;DATA BUFFER INCORRECT
3883	016660	005300			36\$:	DEC	RO	;CHECK IF FINISHED
3884	016662	001364				BNE	35\$;NO READ SECOND WORD
3885	016664	104415				SCOP1		;CHECK IF LOOP ON ERROR
3886	016666	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
3887								
3888								
3889								
3890								
3891								
3892								
3893								
3894								
3895								
3896								
3897								
3898								
3899								
3900								
3901								
3902								
3903	016674	000004				†ST20:	SCOPE	
3904	016676	012737	000144	001200		MOV	#100, \$TIMES	;DO 100. ITERATIONS
3905	016704	005737	046374			TST	\$KT11	;CHECK FOR MEMORY MANAGEMENT
3906	016710	100004				BPL	1\$;NO, BYPASS TEST
3907	016712	022737	004000	046642		CMP	#4000, \$LSTBK	;CHECK IF ENOUGH MEMORY
3908	016720	103417				BLO	2\$;YES, DO TEST
3909	016722				1\$:			
3910	016722	012737	000001	001200		MOV	#1, \$TIMES	;FORCE INTERATION COUNT TO 1
3911	016730	005227	177777			INC	#-1	;ONLY DO ONCE
3912	016734	001007				BNE	64\$;NO, GO TO NEXT TEST
3913	016736	104401	055013			TYPE	TSTBY1	;TYPE TEST N BYPASSED
3914	016742	013746	001220			MOV	\$TESTN, -(SP)	;SAVE \$TESTN FOR TYPEOUT
3915	016746	104402				TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
3916	016750	104401	055023			TYPE	TSTBY2	
3917	016754	000137	020010		64\$:	JMP	†ST21	;GO TO NEXT TEST
3918								
3919	016760	013702	001270		2\$:	MOV	\$BASE, R2	;LOAD K611 BASE
3920	016764	012737	004000	172354		MOV	#4000, KIPAR6	;LOAD PAGE ADDRESS 6 FOR DATA
3921	016772	005237	177572			INC	SRO	;TURN ON MEMORY MANAGEMENT


```

4034 017720 104104          ERROR 104          ;WORD COUNT INCORRECT
4035 017722 062737 000002 003504 31$: ADD #2,E.BA      ;INCREMENT BUS ADDRESS AND
4036 017730 005237 003502          INC E.WC          ;WORD COUNT
4037 017734 005301          DEC R1            ;CHECK IF FINISHED
4038 017736 001305          BNE 23$          ;NO GET SECOND WORD
4039 017740 012700 000002          MOV #2,R0         ;LOAD COUNT AND ADDRESS WORD
4040 017744 012703 067042          MOV #NPRBUF,R3   ;DATA COMPARE
4041 017750 016237 000024 003462 35$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
4042 017756 012337 003522          MOV (R3)+,E.DB   ;GET EXPECTED DATA
4043 017762 023737 003522 003462          CMP E.DB,T.DB    ;CHECK IF CORRECT
4044 017770 001401          BEQ 36$          ;YES CHECK IF FINISHED
4045 017772 104105          ERROR 105        ;DATA BUFFER INCORRECT
4046 017774 005300          DEC R0           ;CHECK IF FINISHED
4047 017776 001364          BNE 35$          ;NO READ SECOND WORD
4048 020000 104415          SCOP1           ;CHECK IF LOOP ON ERROR
4049 020002 012762 100000 000000          MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4050
4051 *****
4052 *TEST 21 ADDRESSING GREATER THAN 96K
4053 *
4054 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4055 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
4056 * IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0.
4057 * SPECIFY A ONE WORD DATA TRANSFER FROM 600000.
4058 * READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
4059 * REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 577776.
4060 * CHECK BUS ADDRESS AND WORD COUNT.
4061 *
4062 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K
4063 * OF MEMORY IS ON THE SYSTEM.
4064 *
4065 *****
4066 *ST21: SCOPE
4067 020010 000004          MOV #100,$TIMES  ;DO 100. ITERATIONS
4068 020012 012737 000144 001200          TST $KT11        ;CHECK FOR MEMORY MANAGEMENT
4069 020020 005737 046374          BPL 1$           ;NO BYPASS TEST
4070 020024 100004          CMP #6000,$LSTBK ;CHECK IF ENOUGH MEMORY
4071 020026 022737 006000 046642          BLO 2$           ;YES, DO TEST
4072 020034 103417          1$: MOV #1,$TIMES   ;FORCE INTERATION COUNT TO 1
4073 020036 012737 000001 001200          INC #-1          ;ONLY DO ONCE
4074 020044 005227 177777          BNE 64$          ;NO GO TO NEXT TEST
4075 020050 001007          TYPE TSTBY1     ;TYPE TEST N BYPASSED
4076 020052 104401 055013          MOV $TESTN,-(SP) ;SAVE $TESTN FOR TYPEOUT
4077 020056 013746 001220          TYP0C           ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4078 020062 104402          TYPE TSTBY2
4079 020064 104401 055023          JMP †ST22        ;GO TO NEXT TEST
4080 020070 000137 021124          64$:
4081
4082 020074 013702 001270          2$: MOV $BASE,R2   ;LOAD K611 BASE
4083 020100 012737 006000 172354          MOV #6000,KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
4084 020106 005237 177572          INC SR0          ;TURN ON MEMORY MANAGEMENT
4085 020112 013737 067042 140000          MOV NPRBUF,140000 ;LOAD WORD IN MEMORY
4086 020120 005037 177572          CLR SR0          ;TURN OFF MEMORY MANAGEMENT
4087 020124 012737 020132 001110          MOV #3,$LPERR    ;LOAD LOOP ON ERROR LOCATION FOR
4088 ; SUBTEST LOOP
4089

```

```

4090 020132          3$:
4091 020132 012762 100000 000000  MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4092 020140 012762 000040 000026  MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4093 020146 012762 177777 000002  MOV #-1,RKWC(R2)
4094 020154 012737 001427 003500  MOV #BA16:BA17:WRHEAD,E.CS1 ;LOAD COMMAND
4095 020162 012762 001427 000000  MOV #BA16:BA17:WRHEAD,RKCS1(R2)
4096 020170 012700 000312          MOV #50.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
4097 020174 012762 000440 000026 5$: MOV #DMD:MCLK,RKMR1(R2) ; INDEX PULSE
4098 020202 012762 000040 000026  MOV #DMD,RKMR1(R2)
4099 020210 005300          DEC RO
4100 020212 001370          BNE 5$
4101 020214 012700 000004          MOV #4,RO ;SIMULATE INDEX PULSE
4102 020220 012762 000240 000026  MOV #DMD:MIND,RKMR1(R2)
4103 020226 012762 000640 000026 6$: MOV #DMD:MIND:MCLK,RKMR1(R2)
4104 020234 012762 000240 000026  MOV #DMD:MIND,RKMR1(R2)
4105 020242 005300          DEC RO
4106 020244 001370          BNE 6$
4107 020246 012762 000040 000026  MOV #DMD,RKMR1(R2)
4108 020254 012700 000045          MOV #37,RO ;ISSUE 1 NPR TRANSFER
4109 020260 012762 000440 000026 7$: MOV #DMD:MCLK,RKMR1(R2)
4110 020266 012762 000040 000026  MOV #DMD,RKMR1(R2)
4111 020274 005300          DEC RO
4112 020276 001370          BNE 7$
4113 020300 016237 000000 003440  MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4114 020306 016237 000004 003444  MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
4115 020314 016237 000002 003442  MOV RKWC(R2),T.WC ;STORE WOR
4116 020322 012700 000024          MOV #20.,RO ;WAIT FOR OUTPUT READY
4117 020326 005300          DEC RO
4118 020330 001376          BNE 8$
4119 020332 016237 000010 003450  MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4120 020340 012737 000300 003510  MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4121 020346 012737 000002 003504  MOV #2,E.BA ;LOAD EXPECTED BUS ADDRESS
4122 020354 005037 003502          CLR E.WC ;LOAD EXPECTED WORD COUNT
4123 020360 023737 003500 003440  CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
4124 020366 001401          BEQ 10$ ;YES, CHECK CS2
4125 020370 104101          ERROR 101 ;CS1 INCORRECT
4126 020372 023737 003510 003450 10$: CMP E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
4127 020400 001401          BEQ 11$ ;YES, CHECK BUS ADDRESS CORRECT
4128 020402 104102          ERROR 102 ;CS2 INCORRECT
4129 020404 023737 003504 003444 11$: CMP E.BA,T.BA ;CHECK IF BUS ADD INCORRECT
4130 020412 001401          BEQ 12$ ;YES, CHECK WORD COUNT CORRECT
4131 020414 104103          ERROR 103 ;BUS ADDRESS INCORRECT
4132 020416 023737 003502 003442 12$: CMP E.WC,T.WC ;CHECK IF WORD COUNT CORRECT
4133 020424 001401          BEQ 13$ ;YES, CHECK DATA BUFFER
4134 020426 104104          ERROR 104 ;WORD COUNT INCORRECT
4135 020430 016237 000024 003462 13$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
4136 020436 013737 067042 003522  MOV NPRBUF,E.DB ;LOAD EXPECTED DATA BUFFER
4137 020444 023737 003522 003462  CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4138 020452 001401          BEQ 15$ ;YES, CHECK IF LOOP ON ERROR
4139 020454 104105          ERROR 105 ;DATA INCORRECT
4140 020456 104415          SCOP1 ;CHECK IF LOOP ON ERROR
4141 020460 012737 005777 172354 15$: MOV #6000-1,KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
4142 020466 005237 177572          INC SR0 ;TURN ON MEMORY MANAGEMENT
4143 020472 013737 067042 140076  MOV NPRBUF,140076 ;LOAD WORDS IN MEMORY
4144 020500 013737 067044 140100  MOV NPRBUF+2,140100
4145 020506 005037 177572          CLR SR0 ;TURN OFF MEMORY MANAGEMENT

```

```

4146 020512 012737 020520 001110 MOV #20$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
4147 ; SUBTEST LOOP
4148
4149 020520 20$:
4150 020520 012762 100000 000000 MOV #CCLR, RKCS1(R2) ;CLEAR RK611
4151 020526 012762 000040 000026 MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4152 020534 012737 177777 003502 MOV #-1, E.WC ;LOAD WORD COUNT REG.
4153 020542 012762 177776 000002 MOV #-2, RKWC(R2)
4154 020550 005037 003504 CLR E.BA ;LOAD BUS ADDRESS
4155 020554 012762 177776 000004 MOV #177776, RKBA(R2)
4156 020562 012737 001427 003500 MOV #BA16!BA17!WRHEAD, E.CS1 ;LOAD COMMAND
4157 020570 012762 001027 000000 MOV #BA17!WRHEAD, RKCS1(R2)
4158 020576 012700 000312 MOV #50, #4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL
4159 020602 012762 000440 000026 21$: MOV #DMD!MCLK, RKMR1(R2) ; INDEX PULSE
4160 020610 012762 000040 000026 MOV #DMD, RKMR1(R2)
4161 020616 005300 DEC R0
4162 020620 001370 BNE 21$
4163 020622 012700 000004 MOV #4, R0 ;SIMULATE INDEX PULSE
4164 020626 012762 000240 000026 MOV #DMD!MIND, RKMR1(R2)
4165 020634 012762 000640 000026 22$: MOV #DMD!MIND!MCLK, RKMR1(R2)
4166 020642 012762 000240 000026 MOV #DMD!MIND, RKMR1(R2)
4167 020650 005300 DEC R0
4168 020652 001370 BNE 22$
4169 020654 012762 000040 000026 MOV #DMD, RKMR1(R2)
4170 020662 012701 000002 MOV #2, R1 ;ISSUE 2 NPR TRANSFERS
4171 020666 012700 000045 23$: MOV #37, R0
4172 020672 012762 000440 000026 24$: MOV #DMD!MCLK, RKMR1(R2)
4173 020700 012762 000040 000026 MOV #DMD, RKMR1(R2)
4174 020706 005300 DEC R0
4175 020710 001370 BNE 24$
4176 020712 016237 000000 003440 MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
4177 020720 016237 000004 003444 MOV RKBA(R2), T.BA ;STORE BUS ADDRESS REG.
4178 020726 016237 000002 003442 MOV RKWC(R2), T.WC ;STORE WORD COUNT
4179 020734 005737 003502 TST E.WC ;CHECK IF FIRST WORD
4180 020740 001404 BEQ 26$ ;NO, GET CS2
4181 020742 012700 000024 MOV #20., R0 ;WAIT FOR OUTPUT READY
4182 020746 005300 25$: DEC R0
4183 020750 001376 BNE 25$
4184 020752 016237 000010 003450 26$: MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG 2
4185 020760 012737 000300 003510 MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
4186 020766 023737 003500 003440 CMP E.CS1, T.CS1 ;CHECK COMMAND STATUS REG 1 CORRECT
4187 020774 001401 BEQ 28$ ;YES, CHECK CS2
4188 020776 104101 ERROR 101 ;CS1 INCORRECT
4189 021000 023737 003510 003450 28$: CMP E.CS2, T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
4190 021006 001401 BEQ 29$ ;YES, CHECK BUS ADDRESS
4191 021010 104102 ERROR 102 ;CS2 INCORRECT
4192 021012 023737 003504 003444 29$: CMP E.BA, T.BA ;CHECK BUS ADDRESS CORRECT
4193 021020 001401 BEQ 30$ ;YES, CHECK WORD COUNT
4194 021022 104103 ERROR 103 ;BUS ADDRESS INCORRECT
4195 021024 023737 003502 003442 30$: CMP E.WC, T.WC ;CHECK WORD COUNT CORRECT
4196 021032 001401 BEQ 31$ ;YES, GET TEXT WORD
4197 021034 104104 ERROR 104 ;WORD COUNT INCORRECT
4198 021036 062737 000002 003504 31$: ADD #2, E.BA ;INCREMENT BUS ADDRESS AND
4199 021044 005237 003502 INC E.WC ;WORD COUNT
4200 021050 005301 DEC R1 ;CHECK IF FINISHED
4201 021052 001305 BNE 23$ ;NO, GET SECOND WORD

```



```

4258      021216      012762      100000      000000      2$:      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4259      021216      012762      000040      000026      MOV      #DMD,RKMR1(R2)
4260      021224      012762      000040      000026      MOV      #-1,RKWC(R2) ;LOAD WORD COUNT NEG
4261      021232      012762      177777      000002      MOV      E.WC
4262      021240      005037      003502      000004      CLR      #BADPAR,RKBA(R2);LOAD BUS ADDRESS
4263      021244      012762      067300      000004      MOV      #BADPAR+2,E.BA
4264      021252      012737      067302      003504      MOV      #CERR!RDY!WRHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4265      021252      012737      100226      003500      MOV      #WRHEAD,RKCS1(R2) ;LOAD COMMAND
4266      021260      012762      000027      000000      MOV      #50.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
4267      021266      012762      000312      000000      MOV      #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
4268      021274      012700      000440      000026      5$:      MOV      #DMD,RKMR1(R2)
4269      021300      012762      000040      000026      MOV      RO
4270      021306      012762      000040      000026      DEC     5$
4271      021314      005300      000004      000026      BNE     #4,RO ;SIMULATE INDEX PULSE
4272      021316      001370      000040      000026      MOV      #DMD!MIND,RKMR1(R2)
4273      021320      012700      000240      000026      6$:      MOV      #DMD!MIND!MCLK,RKMR1(R2)
4274      021324      012762      000240      000026      MOV      #DMD!MIND,RKMR1(R2)
4275      021332      012762      000640      000026      DEC     RO
4276      021340      012762      000240      000026      BNE     6$
4277      021346      005300      000040      000026      MOV      #DMD,RKMR1(R2)
4278      021350      001370      000045      000026      MOV      #37,RO ;ISSUE 1 NPR TRANSFER
4279      021352      012762      000040      000026      7$:      MOV      #DMD!MCLK,RKMR1(R2)
4280      021360      012700      000040      000026      MOV      #DMD,RKMR1(R2)
4281      021364      012762      000040      000026      DEC     RO
4282      021372      012762      000040      000026      BNE     7$
4283      021400      005300      000000      003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG.1
4284      021402      001370      000004      003444      MOV      RKBA(R2),T.BA ;STORE BUS ADDRESS
4285      021404      016237      000002      003442      MOV      RKWC(R2),T.WC ;STORE WORD COUNT
4286      021412      016237      000014      003454      MOV      RKER(R2),T.ER ;STORE ERROR REG
4287      021420      016237      000024      000024      MOV      #20.,RO ;WAIT FOR OUTPUT READY
4288      021426      016237      000024      000024      DEC     RO
4289      021434      012700      000024      000024      BNE     8$
4290      021440      005300      000010      003450      8$:      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG.2
4291      021442      001376      000030      003510      MOV      #IR!OR!UPE,E.CS2 ;LOAD EXPECTED CS2
4292      021444      016237      000010      003450      MOV      E.ER ;LOAD EXPECTED ERROR REG
4293      021452      012737      020300      003510      CLR     E.CS1,T.CS1 ;CHECK CS1 CORRECT
4294      021460      005037      003514      003440      CMP     10$ ;YES CHECK CS2
4295      021464      023737      003500      003440      BEQ     10$ ;CS1 INCORRECT
4296      021472      001401      003510      003450      ERROR  106 ;CHECK CS2 CORRECT
4297      021474      104106      003510      003450      10$:    CMP     E.CS2,T.CS2 ;YES, CHECK BUS ADDRESS
4298      021476      023737      003510      003450      BEQ     107 ;CS2 INCORRECT
4299      021504      001401      003504      003444      11$:    CMP     E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
4300      021506      104107      003504      003444      BEQ     107 ;YES, CHECK WORD COUNT
4301      021510      023737      003504      003444      ERROR  110 ;BUS ADDRESS INCORRECT
4302      021516      001401      003502      003442      12$:    CMP     E.WC,T.WC ;CHECK WORD COUNT CORRECT
4303      021520      104110      003502      003442      BEQ     13$ ;YES,CHECK ERROR REG.
4304      021522      023737      003502      003442      ERROR  111 ;WORD COUNT INCORRECT
4305      021530      001401      003514      003454      13$:    CMP     E.ER,T.ER ;CHECK ERROR REG CORRECT
4306      021532      104111      003514      003454      BEQ     14$ ;YES,CLEAR ERROR
4307      021534      023737      003514      003454      ERROR  116 ;ERROR REG CORRECT
4308      021542      001401      100000      000000      14$:    MOV     #CCLR,RKCS1(R2) ;CLEAR RK611
4309      021544      104116      000000      003440      MOV     RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
4310      021546      012762      000010      003450      MOV     RKCS2(R2),T.CS2 ;STORE COMMAND STATUS REG. 2
4311      021554      016237      000010      003450      MOV     #RDY,E.CS1 ;LOAD EXPECTED CS1
4312      021562      016237      000200      003500      MOV
4313      021570      012737
  
```

```

4314 021576 012737 000100 003510      MOV      #IR,E.CS2      ;LOAD EXPECTED CS2
4315 021604 023737 003500 003440      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4316 021612 001401                BEQ      15$           ;YES, CHECK CS2
4317 021614 104117                ERROR    117           ;CS1 INCORRECT
4318 021616 023737 003510 003450 15$:      CMP      E.CS2,T.CS2   ;CHECK IF CS2 CORRECT
4319 021624 001401                BEQ      16$           ;YES, CONTINUE
4320 021626 104120                ERROR    120           ;CS2 INCORRECT
4321 021630 104415                SCOP1    ;CHECK IF LOOP ON ERROR
4322 021632 012737 021640 001110 16$:      MOV      #20$,SLPERR   ;LOAD LOOP ON ERROR LOCATION FOR
4323                                ; SUBTEST LOOP
4324
4325 021640                20$:
4326 021640 012762 100000 000000      MOV      #CLR,RKCS1(R2) ;CLEAR RK611
4327 021646 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611-IN-DIAGNOSTIC MODE
4328 021654 012762 177776 000002      MOV      #-2,RKWC(R2)   ;LOAD WORD COUNT REG
4329 021662 012737 177777 003502      MOV      #-1,E.WC
4330 021670 012762 067276 000004      MOV      #BADPAR-2,RKBA(R2) ;LOAD BUS ADDRESS REG
4331 021676 012737 067300 003504      MOV      #BADPAR,E.BA
4332 021704 012737 000027 003500      MOV      #WRHEAD,E.CS1  ;LOAD EXPECTED CS1
4333 021712 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ;LOAD COMMAND
4334 021720 012700 000312                MOV      #50,#4+2,RO    ;ISSUE ENOUGH CLOCKS UNTIL
4335 021724 012762 000440 000026 21$:      MOV      #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
4336 021732 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4337 021740 005300                DEC      RO
4338 021742 001370                BNE     21$
4339 021744 012700 000004                MOV      #4,RO          ;SIMULATE INDEX PULSE
4340 021750 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
4341 021756 012762 000640 000026 22$:      MOV      #DMD!MIND!MCLK,RKMR1(R2)
4342 021764 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
4343 021772 005300                DEC      RO
4344 021774 001370                BNE     22$
4345 021776 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4346 022004 012700 000045                MOV      #37,RO        ;ISSUE 1 NPR TRANSFER
4347 022010 012762 000440 000026 23$:      MOV      #DMD!MCLK,RKMR1(R2)
4348 022016 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4349 022024 005300                DEC      RO
4350 022026 001370                BNE     23$
4351 022030 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4352 022036 016237 000004 003444      MOV      RKBA(R2),T.BA  ;STORE BUS ADDRESS
4353 022044 016237 000002 003442      MOV      RKWC(R2),T.WC ;STORE WORD COUNT
4354 022052 012700 000024                MOV      #20.,RO       ;WAIT FOR OUTPUT READY
4355 022056 005300                24$:      DEC      RO
4356 022060 001376                BNE     24$
4357 022062 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4358 022070 012737 000300 003510      MOV      #IR!OR,E.CS2  ;LOAD EXPECTED CS2
4359 022076 023737 003500 003440      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4360 022104 001401                BEQ      25$           ;YES,CHECK CS2
4361 022106 104112                ERROR    112           ;CS1 INCORRECT
4362 022110 023737 003510 003450 25$:      CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4363 022116 001401                BEQ      26$           ;YES, CHECK BUS ADDRESS
4364 022120 104113                ERROR    113           ;CS2 INCORRECT
4365 022122 023737 003504 003444 26$:      CMP      E.BA,T.BA     ;CHECK BUS ADDRESS CORRECT
4366 022130 001401                BEQ      27$           ;YES, CHECK WORD COUNT
4367 022132 104114                ERROR    114           ;BUS ADDRESS INCORRECT
4368 022134 023737 003502 003442 27$:      CMP      E.WC,T.WC     ;CHECK WORD COUNT CORRECT
4369 022142 001401                BEQ      28$           ;YES, DO NEXT NPR TRANSFER

```

```

4370 022144 104115          ERROR 115          ;WORD COUNT INCORRECT
4371 022146 012700 000045 28$: MOV #37, R0      ;ISSUE 1 NPR TRANSFER
4372 022152 012762 000440 000026 30$: MOV #DMD!MCLK, RKMR1(R2)
4373 022160 012762 000040 000026 MOV #DMD, RKMR1(R2)
4374 022166 005300          DEC R0
4375 022170 001370          BNE 30$
4376 022172 016237 000000 003440 MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG 1
4377 022200 016237 000010 003450 MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG 2
4378 022206 016237 000004 003444 MOV RKBA(R2), T.BA   ;STORE BUS ADDRESS
4379 022214 016237 000002 003442 MOV RKWC(R2), T.WC  ;STORE WORD COUNT
4380 022222 016237 000014 003454 MOV RKER(R2), T.ER  ;STORE ERROR REG
4381 022230 012737 100226 003500 MOV #CERR!RDY!WRHEAD<↑C<GO>>, E.CS1 ;LOAD EXPECTED CS1
4382 022236 012737 020300 003510 MOV #OR!IR!UPE, E.CS2 ;LOAD EXPECTED CS2
4383 022244 005237 003502          INC E.WC          ;LOAD EXPECTED WORD COUNT
4384 022250 062737 000002 003504 ADD #2, E.BA      ;LOAD EXPECTED BUS ADDRESS
4385 022256 005037 003514          CLR E.ER          ;LOAD EXPECTED WORD COUNT
4386 022262 023737 003500 003440 CMP E.CS1, T.CS1  ;CHECK CS1 CORRECT
4387 022270 001401          BEQ 31$          ;YES, CHECK CS2
4388 022272 104106          ERROR 106         ;CS1 INCORRECT
4389 022274 023737 003510 003450 31$: CMP E.CS2, T.CS2  ;CHECK CS2 CORRECT
4390 022302 001401          BEQ 32$          ;YES, CHECK BUS ADDRESS
4391 022304 104107          ERROR 107         ;CS2 INCORRECT
4392 022306 023737 003504 003444 32$: CMP E.BA, T.BA   ;CHECK BUS ADDRESS CORRECT
4393 022314 001401          BEQ 33$          ;YES, CHECK WORD COUNT
4394 022316 104110          ERROR 110         ;BUS ADDRESS INCORRECT
4395 022320 023737 003502 003442 33$: CMP E.WC, T.WC   ;CHECK WORD COUNT CORRECT
4396 022326 001401          BEQ 34$          ;YES, CHECK ERROR REG
4397 022330 104111          ERROR 111         ;WORD COUNT INCORRECT
4398 022332 023737 003514 003454 34$: CMP E.ER, T.ER   ;CHECK ERROR REG CORRECT
4399 022340 001401          BEQ 35$          ;YES, CONTINUE
4400 022342 104116          ERROR 116         ;WORD COUNT INCORRECT
4401 022344 104415          SCOPI          ;CHECK IF WORD ON ERROR
4402 022346 012762 100000 000000 MOV #CLR, RKCS1(R2) ;CLEAR RK611
4403 022354 013703 003640          MOV CSRPTR, R3    ;RESET THE PARITY BIT
4404 022360 042713 000007          BIC #7, (R3)      ;24-OCT-77 THE ABOVE TWO LINES
4405 022364 012737 000000 067300 MOV #0, BADPAR    ;WRITE GOOD PARITY

```

```

4406
4407 *****
4408 *TEST 23          SILO FILL IN 18 BIT MODE
4409 *
4410 * CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER
4411 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
4412 * IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0. DRIVE 0,
4413 * SPECIFY 66 WORD DATA TRANSFER. CLOCK
4414 * ALL 66 WORD INTO THE SILO. CHECK THAT ALL 66
4415 * WORDS ARE CORRECT (16 LEAST SIGNIFICANT BITS).
4416 *
4417 *****

```

```

4418 ST23: SCOPE
4419 MOV #100, $TIMES ;DO 100. ITERATIONS
4420 MOV $BASE, R2    ;LOAD RK611 BASE
4421 MOV #CLR, RKCS1(R2) ;CLEAR RK611
4422 MOV #DMD, RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
4423 MOV #-66, RKWC(R2) ;LOAD WORD COUNT
4424 MOV #-65, E.WC
4425 MOV #NPRBUF, RKBA(R2);LOAD BUS ADDRESS

```

4426	022444	012737	067044	003504		MOV	#NPRBUF+2, E.BA	
4427	022452	012737	010027	003500		MOV	#WRHEAD:CFMT, E.CS1	: LOAD EXPECTED CS1
4428	022460	012762	010027	000000		MOV	#WRHEAD:CFMT, RKCS1(R2)	: ISSUE COMMAND
4429	022466	012700	000312			MOV	#SO.*4+2, RO	: ISSUE ENOUGH CLOCKS DATA
4430								: INDEX PULSE
4431	022472	012762	000440	000026	5\$:	MOV	#DMD:MCLK, RKMR1(R2)	
4432	022500	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4433	022506	005300				DEC	RO	
4434	022510	001370				BNE	5\$	
4435	022512	012700	000004			MOV	#4, RO	: SIMULATE INDEX PULSE
4436	022516	012762	000240	000026		MOV	#DMD:MIND, RKMR1(R2)	
4437	022524	012762	000640	000026	6\$:	MOV	#DMD:MIND:MCLK, RKMR1(R2)	
4438	022532	012762	000240	000026		MOV	#DMD:MIND, RKMR1(R2)	
4439	022540	005300				DEC	RO	
4440	022542	001370				BNE	6\$	
4441	022544	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4442	022552	012737	000300	003510		MOV	#IR:OR, E.CS2	: LOAD EXPECTED CS2
4443	022560	012701	000102			MOV	#66., R1	: ISSUE 66 NPR TRANSFERS
4444	022564	012700	000050		7\$:	MOV	#40., RO	
4445	022570	012762	000440	000026	8\$:	MOV	#DMD:MCLK, RKMR1(R2)	
4446	022576	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4447	022604	005300				DEC	RO	
4448	022606	001370				BNE	8\$	
4449	022610	016237	000000	003440		MOV	RKCS1(R2), T.CS1	: STORE COMMAND AND STATUS REG. 1
4450	022616	016237	000004	003444		MOV	RKBA(R2), T.BA	: STORE BUS ADDRESS REG. 2
4451	022624	016237	000002	003442		MOV	RKWC(R2), T.WC	: STORE WORD COUNT
4452	022632	022737	067044	003504		CMP	#NPRBUF+2, E.BA	: CHECK IF FIRST WORD
4453	022640	001004				BNE	10\$: NO, CONTINUE
4454	022642	012700	000024			MOV	#20., RO	: WAIT FOR OUTPUT READY
4455	022646	005300			9\$:	DEC	RO	
4456	022650	001376				BNE	9\$	
4457	022652	016237	000010	003450	10\$:	MOV	RKCS2(R2), T.CS2	: STORE COMMAND AND STATUS REG. 2
4458	022660	005737	003502			TST	E.WC	: CHECK IF LAST WORD
4459	022664	001003				BNE	11\$: NO, CONTINUE
4460	022666	012737	000200	003510		MOV	#OR, E.CS2	: LOAD EXPECTED CS2
4461	022674	023737	003500	003440	11\$:	CMP	E.CS1, T.CS1	: CHECK CS1 CORRECT
4462	022702	001401				BEQ	12\$: YES, CONTINUE
4463	022704	104121				ERROR	121	: CS1 INCORRECT
4464	022706	023737	003510	003450	12\$:	CMP	E.CS2, T.CS2	: CHECK CS2 CORRECT
4465	022714	001401				BEQ	13\$: YES, CONTINUE
4466	022716	104122				ERROR	122	: CS2 INCORRECT
4467	022720	023737	003504	003444	13\$:	CMP	E.BA, T.BA	: CHECK BUS ADDRESS CORRECT
4468	022726	001401				BEQ	14\$: YES, CONTINUE
4469	022730	104123				ERROR	123	: BUS ADDRESS INCORRECT
4470	022732	023737	003502	003442	14\$:	CMP	E.WC, T.WC	: CHECK WORD COUNT REG CORRECT
4471	022740	001401				BEQ	15\$: YES, CONTINUE
4472	022742	104124				ERROR	124	: WORD COUNT INCORRECT
4473	022744	062737	000002	003504	15\$:	ADD	#2, E.BA	: INCREMENT BUS ADDRESS AND
4474	022752	005237	003502			INC	E.WC	: WORD COUNT
4475	022756	005301				DEC	R1	: CHECK IF SILO FULL
4476	022760	001301				BNE	7\$: NO, GET NEXT WORD
4477	022762	012700	000120			MOV	#2*40., RO	: ISSUE CLOCKS FOR TWO NPR'S
4478	022766	012762	000440	000026	20\$:	MOV	#DMD:MCLK, RKMR1(R2)	
4479	022774	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4480	023002	005300				DEC	RO	
4481	023004	001370				BNE	20\$	

H07

```

4482 023006 162737 000002 003504 SUB #2,E.BA ;ADJUST BUS ADDRESS AND WORD COUNT
4483 023014 005037 003502 CLR E.WC
4484 023020 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG: 1
4485 023026 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG: 2
4486 023034 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
4487 023042 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
4488 023050 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4489 023056 001401 BEQ 21$ ;YES, CONTINUE
4490 023060 104125 ERROR 125 ;CS1 INCORRECT
4491 023062 023737 003510 003450 21$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4492 023070 001401 BEQ 22$ ;YES, CONTINUE
4493 023072 104126 ERROR 126 ;CS2 INCORRECT
4494 023074 023737 003504 003444 22$: CMP E.BA,T.BA ;CHECK BUS ADDRESS INCORRECT
4495 023102 001401 BEQ 23$ ;YES, CONTINUE
4496 023104 104127 ERROR 127 ;BUS ADDRESS INCORRECT
4497 023106 023737 003502 003442 23$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
4498 023114 001401 BEQ 24$ ;YES, CONTINUE
4499 023116 104130 ERROR 130 ;WORD COUNT INCORRECT
4500 023120 012703 067042 24$: MOV #NPRBUF,R3 ;LOAD BUFFER ADDRESS FOR COMPARE
4501 023124 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4502 023132 012701 000102 MOV #66,R1 ;LOAD COUNT
4503 023136 005037 003624 CLR WRDCNT ;INITIALIZE WORD COUNT
4504 023142 016237 000024 003462 25$: MOV RKDB(R2),T.DB ;GET DATA BUFFER
4505 023150 012337 003522 (R3)+,E.DB ;GET EXPECTED DATA
4506 023154 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4507 023162 001401 BEQ 26$ ;YES, CONTINUE
4508 023164 104131 ERROR 131 ;DATA READ INCORRECT
4509 023166 012700 000050 26$: MOV #40.,R0 ;SET STALL
4510 023172 005300 27$: DEC R0 ;RUN STALL TO ZERO
4511 023174 001376 BNE 27$
4512 023176 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
4513 023204 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
4514 023212 022701 000001 CMP #1,R1 ;CHECK IF LAST WORD
4515 023216 001003 BNE 28$ ;NO, CONTINUE
4516 023220 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4517 023226 023737 003500 003440 28$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4518 023234 001401 BEQ 29$ ;YES, CONTINUE
4519 023236 104132 ERROR 132 ;CS1 INCORRECT
4520 023240 023737 003510 003450 29$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4521 023246 001401 BEQ 30$ ;YES, CONTINUE
4522 023250 104133 ERROR 133 ;CS2 INCORRECT
4523 023252 005237 003624 30$: INC WRDCNT ;INCREMENT WORD COUNT
4524 023256 005301 DEC R1 ;CHECK IF FINISHED
4525 023260 001330 BNE 25$ ;NO, CONTINUE

```

```

4526 *****
4527 ;*TEST 24 BIT 16 AND 17 READING WITH NPR
4528 ;*
4529 ;*
4530 ;* CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER
4531 ;* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
4532 ;* IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4533 ;* SPECIFY ONE WORD DATA TRANSFER FROM A LOCATION
4534 ;* WITH BAD PARITY. MAKE SURE A UNIBUS PARITY
4535 ;* DOES NOT OCCUR.
4536 ;*
4537 ;* NOTE: THIS TEST IS EXECUTED ONLY IF MEMORY PARITY

```

```

4538 ;* ENABLE EXISTS FOR SPECIFIED LOCATION AND IS
4539 ;* NOT RUN ON AN 11/70.
4540
4541 ;*
4542 ;* *****
4543 023262 000004 †ST24: SCOPE
4544 023264 012737 000144 001200 MOV #100, $TIMES ; DO 100. ITERATIONS
4545 023272 005737 003636 TST PARPRÉ ; CHECK IF MEMORY PARITY AVAILABLE
4546 023276 001017 BNE 1$ ; YES, DO TEST
4547 023300 012737 000001 001200 MOV #1, $TIMES ; FORCE INTERATION COUNT TO 1
4548 023306 005227 177777 INC #-1 ; ONLY DO ONCE
4549 023312 001007 BNE 64$ ; NO, GO TO NEXT TEST
4550 023314 104401 055013 TYPE TSTBY1 ; TYPE TEST N BYPASSED
4551 023320 013746 001220 MOV $TESTN, -(SP) ; SAVE $TESTN FOR TYPEOUT
4552 023324 104402 TYPOC ; GO TYPE--OCTAL ASCII(ALL DIGITS)
4553 023326 104401 055023 TYPE TSTBY2
4554 023332 000137 024116 64$: JMP †ST25 ; GO TO NEXT TEST
4555
4556 023336 013702 001270 1$: MOV $BASE, R2 ; LOAD RK611 BASE
4557 023342 004737 044714 JSR PC, WRTPAR ; GENERATE BAD PARITY
4558 023346 012762 100000 000000 MOV #CLR, RKCS1(R2) ; CLEAR RK611
4559 023354 012762 000040 000026 MOV #DMD, RKMR1(R2) ; PUT RK611 IN MAINTENANCE MODE
4560 023362 012762 177777 000002 MOV #-1, RKWC(R2) ; LOAD WORD COUNT
4561 023370 005037 003502 CLR E.WC
4562 023374 012762 067300 000004 MOV #BADPAR, RKBA(R2) ; LOAD BUS ADDRESS
4563 023402 012737 067302 003504 MOV #BADPAR+2, E.BA
4564 023410 012737 010027 003500 MOV #WRHEAD:CFMT, E.CS1 ; LOAD EXPECTED CS1
4565 023416 012762 010027 000000 MOV #WRHEAD:CFMT, RKCS1(R2) ; ISSUE COMMAND
4566 023424 012700 000312 MOV #50, *4+2, RO ; ISSUE ENOUGH CLOCKS UNTIL
4567 ; INDEX PULSE
4568 023430 012762 000440 000026 5$: MOV #DMD, MCLK, RKMR1(R2)
4569 023436 012762 000040 000026 MOV #DMD, RKMR1(R2)
4570 023444 005300 DEC RO
4571 023446 001370 BNE 5$
4572 023450 012700 000004 MOV #4, RO ; SIMULATE INDEX PULSE
4573 023454 012762 000240 000026 MOV #DMD, MIND, RKMR1(R2)
4574 023462 012762 000640 000026 6$: MOV #DMD, MIND, MCLK, RKMR1(R2)
4575 023470 012762 000240 000026 MOV #DMD, MIND, RKMR1(R2)
4576 023476 005300 DEC RO
4577 023500 001370 BNE 6$
4578 023502 012762 000040 000026 MOV #DMD, RKMR1(R2)
4579 023510 012737 000300 003510 MOV #IR!OR, E.CS2 ; LOAD EXPECTED CS2
4580 023516 012700 000050 MOV #40, RO ; ISSUE NPR TRANSFER
4581 023522 012762 000440 000026 8$: MOV #DMD, MCLK, RKMR1(R2)
4582 023530 012762 000040 000026 MOV #DMD, RKMR1(R2)
4583 023536 005300 DEC RO
4584 023540 001370 BNE 8$
4585 023542 016237 000000 003440 MOV RKCS1(R2), T.CS1 ; STORE COMMAND AND STATUS REG. 1
4586 023550 016237 000004 003444 MOV RKBA(R2), T.BA ; STORE BUS ADDRESS
4587 023556 016237 000002 003442 MOV RKWC(R2), T.WC ; STORE WORD COUNT
4588 023564 012700 000024 MOV #20, RO ; WAIT FOR OUTPUT READY
4589 023570 005300 9$: DEC RO
4590 023572 001376 BNE 9$
4591 023574 016237 000010 003450 MOV RKCS2(R2), T.CS2 ; STORE COMMAND AND STATUS REG. 2
4592 023602 023737 003500 003440 CMP E.CS1, T.CS1 ; CHECK CS1 CORRECT
4593 023610 001401 BEQ 12$ ; YES, CHECK CS2

```

4594	023612	104134				ERROR	134	; CS1 INCORRECT
4595	023614	023737	003510	003450	12\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4596	023622	001401				BEQ	13\$;YES, CHECK BUS ADDRESS
4597	023624	104135				ERROR	13\$;CS2 INCORRECT
4598	023626	023737	003504	003444	13\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
4599	023634	001401				BEQ	14\$;YES, CHECK WORD COUNT
4600	023636	104136				ERROR	136	;BUS ADDRESS INCORRECT
4601	023640	023737	003502	003442	14\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT CORRECT
4602	023646	001401				BEQ	15\$;YES, CONTINUE
4603	023650	104137				ERROR	137	;WORD COUNT INCORRECT
4604	023652	012700	000120		15\$:	MOV	#2*40,RO	;ISSUE CLOCKS FOR TWO NPR'S
4605	023656	012762	000440	000026	20\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4606	023664	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4607	023672	005300				DEC	RO	
4608	023674	001370				BNE	20\$	
4609	023676	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND STATUS REG. 1
4610	023704	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
4611	023712	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS REG.
4612	023720	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
4613	023726	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4614	023734	001401				BEQ	21\$;YES, CHECK CS2
4615	023736	104140				ERROR	140	;CS1 INCORRECT
4616	023740	023737	003510	003450	21\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4617	023746	001401				BEQ	22\$;YES, CHECK BUS ADDRESS
4618	023750	104141				ERROR	141	;CS2 INCORRECT
4619	023752	023737	003504	003444	22\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
4620	023760	001401				BEQ	23\$;YES, CHECK WORD COUNT
4621	023762	104142				ERROR	142	;BUS ADDRESS INCORRECT
4622	023764	023737	003502	003442	23\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT REG. CORRECT
4623	023772	001401				BEQ	24\$;YES, CHECK DATA
4624	023774	104143				ERROR	143	;WORD COUNT INCORRECT
4625	023776	016237	000024	003462	24\$:	MOV	RKDB(R2),T.DB	;READ DATA BUFFER
4626	024004	012737	000157	003522		MOV	#157,E.DB	;LOAD EXPECT DATA
4627	024012	023737	003522	003462		CMP	E.DB,T.DB	;CHECK TO MAKE SURE CORRECT
4628	024020	001401				BEQ	26\$;YES, CONTINUE
4629	024022	104144				ERROR	144	;DATA INCORRECT
4630	024024	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG 1
4631	024032	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG 2
4632	024040	012737	000100	003510		MOV	#IR,E.CS2	;LOAD EXPECTED CS2
4633	024046	023737	003500	003440		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4634	024054	001401				BEQ	29\$;YES, CHECK SC2
4635	024056	104145				ERROR	145	;CS1 INCORRECT
4636	024060	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	;CHECK CS1 CORRECT
4637	024066	001401				BEQ	30\$;YES, CONTINUE
4638	024070	104146				ERROR	146	;CS2 INCORRECT
4639	024072				30\$:			
4640	024072	013703	003640			MOV	CSRPTR,R3	;RESET THE PARITY BIT
4641	024076	042713	000007			BIC	#7,(R3)	;PARITY BIT FOR THE WRONG MEM IS DISABLED
4642	024102	012737	000000	067300		MOV	#0,BADPAR	;WRITE GOOD PARITY
4643	024110	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611

.SBTTL **MFM READ LOOPBACK TESTS

```

*****
;TEST 25 READ LOOPBACK (PART 1)
;*
```

4644
4645
4646
4647
4648
4649

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

177777
000000
177777

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

4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666 024116 000004
4667 024120 012737 000144 001200
4668 024126 013702 001270
4669 024132 012762 100000 000000
4670 024140 012762 000040 000026
4671 024146 012762 000025 000000
4672 024154 012700 000312
4673 024160 012762 000440 000026 1\$:
4674 024166 012762 000040 000026
4675 024174 005300
4676 024176 001370
4677 024200 012762 000140 000026
4678 024206 012762 000040 000026
4679 024214 005037 003614
4680 024220 005037 003616
4681 024224 012700 000341
4682 024230 004737 046164 2\$:
4683 024234 005300
4684 024236 001374
4685 024240 012737 000001 003614
4686 024246 004737 046164
4687 024252 012701 000003
4688 024256 012703 066762
4689 024262 012737 000025 003500
4690 024270 012304 5\$:
4691 024272 012700 000020
4692 024276 013737 003614 003616 6\$:
4693 024304 006004
4694 024306 103403
4695 024310 005037 003614
4696 024314 000403
4697
4698 024316 012737 000001 003614 7\$:
4699 024324 004737 046164 8\$:
4700 024330 016237 000000 003440
4701 024336 023737 003500 003440
4702 024344 001417
4703 024346 012737 000003 003624
4704 024354 160137 003624
4705 024360 012737 000020 003622

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

```

4706 024366 160037 003622 SUB RO,BITCNT
4707 024372 104150 ERROR 150 ;CS1 INCORRECT DURING HEADER
4708 024374 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4709 024402 000522 BR TST26 ;GO ON TO NEXT TEST
4710
4711 024404 005300 9$: DEC RO ;CHECK IF READY FOR NEXT WORD
4712 024406 001333 BNE 6$ ;NO, GET NEXT BIT
4713 024410 005301 DEC R1 ;CHECK IF HEADER FINISHED
4714 024412 001326 BNE 5$ ;NO, GET NEXT WORD
4715 024414 012700 000004 MOV #4,RO ;LOAD COUNT FOR POSTAMBLE
4716 024420 013737 003614 003616 15$: MOV PR,BIT,M1,BIT ;STORE LAST BIT
4717 024426 005037 003614 CLR PR,BIT ;LOAD NEXT BIT
4718 024432 004737 046164 JSR PC,RDBIT ;SIMULATE 1 BIT READ
4719 024436 005300 DEC RO ;CHECK IF TIME FOR READY
4720 024440 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
4721 024442 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4722 024450 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4723 024456 012737 000224 003500 MOV #RDY:RDHEAD<+C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4724 024464 012737 000300 003510 MOV #OR:IR,E.CS2 ;LOAD EXPECTED CS2
4725 024472 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4726 024500 001401 BEQ 16$ ;YES, CHECK CS2
4727 024502 104151 ERROR 151 ;CS1 INCORRECT
4728 024504 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4729 024512 001401 BEQ 17$ ;YES, CHECK DATA
4730 024514 104152 ERROR 152 ;CS2 INCORRECT
4731 024516 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
4732 024522 012703 066762 MOV #HEAD1,R3 ;GET ADDRESS OF DATA
4733 024526 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
4734 024532 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4735 024540 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4736 024546 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4737 024554 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
4738 024562 001003 BNE 21$ ;NO, CHECK CS1
4739 024564 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
4740 024572 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4741 024600 001402 BEQ 22$ ;YES, CHECK CS2
4742 024602 104153 ERROR 153 ;CS1 INCORRECT
4743 024604 000421 BR TST26 ;GO ON TO NEXT TEST
4744
4745 024606 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4746 024614 001402 BEQ 23$ ;YES, CHECK DATA
4747 024616 104154 ERROR 154 ;CS2 INCORRECT
4748 024620 000413 BR TST26 ;GO ON TO NEXT TEST
4749
4750 024622 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4751 024630 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
4752 024632 104155 ERROR 155 ;DATA INCORRECT
4753 024634 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
4754 024640 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
4755 024646 001327 BNE 20$ ;NO, GET NEXT WORD
4756
4757 *****
4758 ;*TEST 26 READ LOOPBACK (PART 2)
4759 ;*
4760 ;* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4761 ;* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06

```

IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE, 255 ZEROES, A ONE,
AND A HEADER CONSISTING OF THE THREE
FOLLOWING WORDS:

000000
177777
000000

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

4762				
4763				
4764				
4765				
4766				
4767				
4768				
4769				
4770				
4771				
4772				
4773				
4774				
4775				
4776	024650	000004		
4777	024652	012737	000144	001200
4778	024660	013702	001270	
4779	024664	012762	100000	000000
4780	024672	012762	000040	000026
4781	024700	012762	000025	000000
4782	024706	012700	000312	
4783	024712	012762	000440	000026
4784	024720	012762	000040	000026
4785	024726	005300		
4786	024730	001370		
4787	024732	012762	000140	000026
4788	024740	012762	000040	000026
4789	024746	005037	003614	
4790	024752	005037	003616	
4791	024756	012700	000341	
4792	024762	004737	046164	
4793	024766	005300		
4794	024770	001374		
4795	024772	012737	000001	003614
4796	025000	004737	046164	
4797	025004	012701	000003	
4798	025010	012703	066770	
4799	025014	012737	000025	003500
4800	025022	012304		
4801	025024	012700	000020	
4802	025030	013737	003614	003616
4803	025036	006004		
4804	025040	103403		
4805	025042	005037	003614	
4806	025046	000403		
4807				
4808	025050	012737	000001	003614
4809	025056	004737	046164	
4810	025062	016237	000000	003440
4811	025070	023737	003500	003440
4812	025076	001417		
4813	025100	012737	000003	003624
4814	025106	160137	003624	
4815	025112	012737	000020	003622
4816	025120	160037	003622	
4817	025124	104150		

```

TST26: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD, RKCS1(R2) ; ISSUE READ HEADER
MOV #50, #4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK, RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225, R0
2$: JSR PC, RDBIT ; SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1, PR.BIT
JSR PC, RDBIT
MOV #3, R1 ; LOAD NUMBER OF WORDS
MOV #HEAD2, R3 ; LOAD ADDRESS OF DATA
MOV #RDHEAD, E.CS1 ; LOAD EXPECTED CS1
5$: MOV (R3)+, R4 ; GET DATA
MOV #16, R0 ; LOAD BIT COUNT
6$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 7$ ; CHECK IF 1
CLR PR.BIT ; NO ZERO
BR 8$ ; SIMULATE READ DATA
7$: MOV #1, PR.BIT ; ONE
8$: JSR PC, RDBIT ; SIMULATE READ DATA
MOV RKCS1(R2), T.CS1 ; READ COMMAND AND STATUS REG. 1
CMP E.CS1, T.CS1 ; CHECK IF CS1 CORRECT
BEQ 9$ ; YES, SIMULATE NEXT BIT
MOV #3, WRDCNT ; LOAD WORD COUNT
SUB R1, WRDCNT
MOV #16, BITCNT ; LOAD BIT COUNT
SUB R0, BITCNT
ERROR 150 ; CS1 INCORRECT DURING HEADER

```

```

4818 025126 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4819 025134 000522                BR        TST27          ;;GO ON TO NEXT TEST
4820
4821          025136 005300          9$:      DEC        RO          ;CHECK IF READY FOR NEXT WORD
4822 025140 001333                BNE       6$            ;NO, GET NEXT BIT
4823 025142 005301                DEC        R1          ;CHECK IF HEADER FINISHED
4824 025144 001326                BNE       5$            ;NO, GET NEXT WORD
4825 025146 012700 000004      MOV      #4,RO          ;LOAD COUNT FOR POSTAMBLE
4826 025152 013737 003614 003616 15$:      MOV      PR,BIT,M1.BIT ;STORE LAST BIT
4827 025160 005037 003614      CLR      PR,BIT        ;LOAD NEXT BIT
4828 025164 004737 046164      JSR      PC,RDBIT      ;SIMULATE 1 BIT READ
4829 025170 005300                DEC        RO          ;CHECK IF TIME FOR READY
4830 025172 001367                BNE       15$          ;NO, CONTINUE WITH POSTAMBLE
4831 025174 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
4832 025202 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
4833 025210 012737 000224 003500      MOV      #RDY!RDHEAD<+C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4834 025216 012737 000300 003510      MOV      #OR!IR,E.CS2   ;LOAD EXPECTED CS2
4835 025224 023737 003500 003440      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4836 025232 001401                BEQ       16$          ;YES, CHECK CS2
4837 025234 104151                ERROR    151          ;CS1 INCORRECT
4838 025236 023737 003510 003450 16$:      CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4839 025244 001401                BEQ       17$          ;YES, CHECK DATA
4840 025246 104152                ERROR    152          ;CS2 INCORRECT
4841 025250 005037 003624          17$:      CLR      WRDCNT        ;INITIALIZE WORD COUNT
4842 025254 012703 066770      MOV      #HEAD2,R3     ;GET ADDRESS OF DATA
4843 025260 012337 003522      MOV      (R3)+,E.DB    ;GET EXPECTED DATA
4844 025264 016237 000024 003462      MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
4845 025272 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4846 025300 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4847 025306 022737 000002 003624      CMP      #2,WRDCNT     ;CHECK IF LAST WORD IN DATA BUFFER
4848 025314 001003                BNE       21$          ;NO, CHECK CS1
4849 025316 012737 000100 003510      MOV      #IR,E.CS2     ;STORE EXPECTED CS2
4850 025324 023737 003500 003440 21$:      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4851 025332 001402                BEQ       22$          ;YES, CHECK CS2
4852 025334 104153                ERROR    153          ;CS1 INCORRECT
4853 025336 000421                BR        TST27          ;;GO ON TO NEXT TEST
4854
4855 025340 023737 003510 003450 22$:      CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4856 025346 001402                BEQ       23$          ;YES, CHECK DATA
4857 025350 104154                ERROR    154          ;CS2 INCORRECT
4858 025352 000413                BR        TST27          ;;GO ON TO NEXT TEST
4859
4860 025354 023737 003522 003462 23$:      CMP      E.DB,T.DB     ;CHECK IF DATA CORRECT
4861 025362 001401                BEQ       24$          ;YES, GET NEXT HEADER WORD
4862 025364 104155                ERROR    155          ;DATA INCORRECT
4863 025366 005237 003624          24$:      INC      WRDCNT        ;INCREMENT WORD COUNT
4864 025372 022737 000003 003624      CMP      #3,WRDCNT     ;CHECK IF ALL THREE WORDS CHECK
4865 025400 001327                BNE       20$          ;NO, GET NEXT WORD

```

```

*****
*TEST 27      READ LOOPBACK (PART 3)
*
*      CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
*      IN DIAGNOSTIC MOVE. ISSUE A READ HEADER TO AN RK06
*      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
*      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES
*

```

```

4866
4867
4868
4869
4870
4871
4872
4873

```

```

4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886 025402 000004
4887 025404 012737 000144 001200
4888 025412 013702 001270
4889 025416 012762 100000 000000
4890 025424 012762 000040 000026
4891 025432 012762 000025 000000
4892 025440 012700 000312
4893 025444 012762 000440 000026 1$:
4894 025452 012762 000040 000026
4895 025460 005300
4896 025462 001370
4897 025464 012762 000140 000026
4898 025472 012762 000040 000026
4899 025500 005037 003614
4900 025504 005037 003616
4901 025510 012700 000341
4902 025514 004737 046164 2$:
4903 025520 005300
4904 025522 001374
4905 025524 012737 000001 003614
4906 025532 004737 046164
4907 025536 012701 000003
4908 025542 012703 066776
4909 025546 012737 000025 003500
4910 025554 012304 5$:
4911 025556 012700 000020
4912 025562 013737 003614 003616 6$:
4913 025570 006004
4914 025572 103403
4915 025574 005037 003614
4916 025600 000403
4917
4918 025602 012737 000001 003614 7$:
4919 025610 004737 046164 8$:
4920 025614 016237 000000 003440
4921 025622 023737 003500 003440
4922 025630 001417
4923 025632 012737 000003 003624
4924 025640 160137 003624
4925 025644 012737 000020 003622
4926 025652 160037 003622
4927 025656 104150
4928 025660 012762 100000 000000
4929 025666 000522

```

```

; * SIMULATE SECTOR PULSE, 255 ZEROES, A
; * ONE, AND A HEADER CONSISTING OF THE THREE
; * FOLLOWING WORDS:
; *
; * 125252
; * 052525
; * 125252
; *
; * MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
; * IS TRANSFERRED. CHECK THE SILC FOR CORRECT CONTENTS.
; *
; *****
TST27: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD, RKCS1(R2) ; ISSUE READ HEADER
MOV #50, *4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK, RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225, R0
2$: JSR PC, RDBIT ; SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1, PR.BIT
JSR PC, RDBIT
MOV #3, R1 ; LOAD NUMBER OF WORDS
MOV #HEAD3, R3 ; LOAD ADDRESS OF DATA
MOV #RDHEAD, E.CS1 ; LOAD EXPECTED CS1
5$: MOV (R3)+, R4 ; GET DATA
MOV #16, R0 ; LOAD BIT COUNT
6$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 7$ ; CHECK IF 1
CLR PR.BIT ; NO, ZERO
BR 8$ ; SIMULATE READ DATA
7$: MOV #1, PR.BIT ; ONE
8$: JSR PC, RDBIT ; SIMULATE READ DATA
MOV RKCS1(R2), T.CS1 ; READ COMMAND AND STATUS REG. 1
CMP E.CS1, T.CS1 ; CHECK IF CS1 CORRECT
BEQ 9$ ; YES, SIMULATE NEXT BIT
MOV #3, WRDCNT ; LOAD WORD COUNT
SUB R1, WRDCNT
MOV #16, BITCNT ; LOAD BIT COUNT
SUB R0, BITCNT
ERROR 150 ; CS1 INCORRECT DURING HEADER
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
BR TST30 ; GO ON TO NEXT TEST

```

```

4930
4931 025670 005300          9$: DEC      RO          ;CHECK IF READY FOR NEXT WORD
4932 025672 001333          BNE      6$          ;NO, GET NEXT BIT
4933 025674 005301          DEC      R1          ;CHECK IF HEADER FINISHED
4934 025676 001326          BNE      5$          ;NO, GET NEXT WORD
4935 025700 012700 000004    MOV      #4,RO      ;LOAD COUNT FOR POSTAMBLE
4936 025704 013737 003614 003616 15$: CLR      PR,BIT,M1.BIT ;STORE LAST BIT
4937 025712 005037 003614    MOV      PR,BIT     ;LOAD NEXT BIT
4938 025716 004737 046164    JSR      PC,RDBIT   ;SIMULATE 1 BIT READ
4939 025722 005300          DEC      RO          ;CHECK IF TIME FOR READY
4940 025724 001367          BNE      15$        ;NO, CONTINUE WITH POSTAMBLE
4941 025726 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
4942 025734 016237 000010 003450  MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
4943 025742 012737 000224 003500  MOV      #RDY!RDHEAD<1C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4944 025750 012737 000300 003510  MOV      #OR!IR,E.CS2  ;LOAD EXPECTED CS2
4945 025756 023737 003500 003440  CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
4946 025764 001401          BEQ      16$        ;YES, CHECK CS2
4947 025766 104151          ERROR    151        ;CS1 INCORRECT
4948 025770 023737 003510 003450 16$: CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
4949 025776 001401          BEQ      17$        ;YES, CHECK DATA
4950 026000 104152          ERROR    152        ;CS2 INCORRECT
4951 026002 005037 003624          CLR      WRDCNT     ;INITIALIZE WORD COUNT
4952 026006 012703 066776          MOV      #HEAD3,R3  ;GET ADDRESS OF DATA
4953 026012 012337 003522          MOV      (R3)+,E.DB ;GET EXPECTED DATA
4954 026016 016237 000024 003462  MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
4955 026024 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4956 026032 016237 000010 003450  MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4957 026040 022737 000002 003624  CMP      #2,WRDCNT  ;CHECK IF LAST WORD IN DATA BUFFER
4958 026046 001003          BNE      21$        ;NO, CHECK CS1
4959 026050 012737 000100 003510  MOV      #IR,E.CS2   ;STORE EXPECTED CS2
4960 026056 023737 003500 003440 21$: CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
4961 026064 001402          BEQ      22$        ;YES, CHECK CS2
4962 026066 104153          ERROR    153        ;CS1 INCORRECT
4963 026070 000421          BR       TST30     ;GO ON TO NEXT TEST
4964
4965 026072 023737 003510 003450 22$: CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
4966 026100 001402          BEQ      23$        ;YES, CHECK DATA
4967 026102 104154          ERROR    154        ;CS2 INCORRECT
4968 026104 000413          BR       TST30     ;GO ON TO NEXT TEST
4969
4970 026106 023737 003522 003462 23$: CMP      E.DB,T.DB   ;CHECK IF DATA CORRECT
4971 026114 001401          BEQ      24$        ;YES, GET NEXT HEADER WORD
4972 026116 104155          ERROR    155        ;DATA INCORRECT
4973 026120 005237 003624          INC      WRDCNT     ;INCREMENT WORD COUNT
4974 026124 022737 000003 003624 24$: CMP      #3,WRDCNT  ;CHECK IF ALL THREE WORDS CHECK
4975 026132 001327          BNE      20$        ;NO, GET NEXT WORD
4976

```

```

*****
;TEST 30 READ LOOPBACK (PART 4)
*****
;
; CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
; IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
; IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
; CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
; SIMULATE SECTOR PULSE, 225 ZEROES, A
; ONE, AND A HEADER CONSISTING OF THE THREE
;

```

```

4977
4978
4979
4980
4981
4982
4983
4984
4985

```

FOLLOWING WORDS:

044444
022222
111111

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

4986				
4987				
4988				
4989				
4990				
4991				
4992				
4993				
4994				
4995				
4996	026134	000004		
4997	026136	012737	000144	001200
4998	026144	013702	001270	
4999	026150	012762	100000	000000
5000	026156	012762	000040	000026
5001	026164	012762	000025	000000
5002	026172	012700	000312	
5003	026176	012762	000440	000026
5004	026204	012762	000040	000026
5005	026212	005300		
5006	026214	001370		
5007	026216	012762	000140	000026
5008	026224	012762	000040	000026
5009	026232	005037	003614	
5010	026236	005037	003616	
5011	026242	012700	000341	
5012	026246	004737	046164	
5013	026252	005300		
5014	026254	001374		
5015	026256	012737	000001	003614
5016	026264	004737	046164	
5017	026270	012701	000003	
5018	026274	012703	067012	
5019	026300	012737	000025	003500
5020	026306	012304		
5021	026310	012700	000020	
5022	026314	013737	003614	003616
5023	026322	006004		
5024	026324	103403		
5025	026326	005037	003614	
5026	026332	000403		
5027				
5028	026334	012737	000001	003614
5029	026342	004737	046164	
5030	026346	016237	000000	003440
5031	026354	023737	003500	003440
5032	026362	001417		
5033	026364	012737	000003	003624
5034	026372	160137	003624	
5035	026376	012737	000020	003622
5036	026404	160037	003622	
5037	026410	104150		
5038	026412	012762	100000	000000
5039	026420	000522		
5040				
5041	026422	005300		

```

TST30: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD, RKCS1(R2) ; ISSUE READ HEADER
MOV #50, #4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK, RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD, RKMR1(R2)
RO
DEC R0
BNE 1$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225, R0
2$: JSR PC, RDBIT ; SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1, PR.BIT
JSR PC, RDBIT
MOV #3, R1 ; LOAD NUMBER OF WORDS
MOV #HEAD4, R3 ; LOAD ADDRESS OF DATA
MOV #RDHEAD, E.CS1 ; LOAD EXPECTED CS1
5$: MOV (R3)+, R4 ; GET DATA
MOV #16, R0 ; LOAD BIT COUNT
6$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT
BCS 7$ ; CHECK IF 1
CLR PR.BIT ; NO ZERO
BR 8$ ; SIMULATE READ DATA
7$: MOV #1, PR.BIT ; ONE
8$: JSR PC, RDBIT ; SIMULATE READ DATA
MOV RKCS1(R2), T.CS1 ; READ COMMAND AND STATUS REG. 1
CMP E.CS1, T.CS1 ; CHECK IF CS1 CORRECT
BEQ 9$ ; YES, SIMULATE NEXT BIT
MOV #3, WRDCNT ; LOAD WORD COUNT
SUB R1, WRDCNT
MOV #16, BITCNT ; LOAD BIT COUNT
SUB R0, BITCNT
ERROR 150 ; CS1 INCORRECT DURING HEADER
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
BR TST31 ; GO ON TO NEXT TEST
9$: DEC R0 ; CHECK IF READY FOR NEXT WORD

```

5042	026424	001333				BNE	6\$:NO, GET NEXT BIT
5043	026426	005301				DEC	R1	:CHECK IF HEADER FINISHED
5044	026430	001326				BNE	5\$:NO, GET NEXT WORD
5045	026432	012700	000004			MOV	#4,R0	:LOAD COUNT FOR POSTAMBLE
5046	026436	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
5047	026444	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
5048	026450	004737	046164			JSR	PC,RDBIT	:SIMULATE 1 BIT READ
5049	026454	005300				DEC	R0	:CHECK IF TIME FOR READY
5050	026456	001367				BNE	15\$:NO, CONTINUE WITH POSTAMBLE
5051	026460	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
5052	026466	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
5053	026474	012737	000224	003500		MOV	#RDY!RDHEAD<1C<GO>>,E.CS1	:LOAD EXPECTED CS1
5054	026502	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
5055	026510	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
5056	026516	001401				BEQ	16\$:YES, CHECK CS2
5057	026520	104151				ERROR	151	:CS1 INCORRECT
5058	026522	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5059	026530	001401				BEQ	17\$:YES, CHECK DATA
5060	026532	104152				ERROR	152	:CS2 INCORRECT
5061	026534	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
5062	026540	012703	067012			MOV	#HEAD4,R3	:GET ADDRESS OF DATA
5063	026544	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
5064	026550	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
5065	026556	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
5066	026564	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
5067	026572	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
5068	026600	001003				BNE	21\$:NO, CHECK CS1
5069	026602	012737	000100	003510		MOV	#IR,E.CS2	:STORE EXPECTED CS2
5070	026610	023737	003500	003440	21\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
5071	026616	001402				BEQ	22\$:YES, CHECK CS2
5072	026620	104153				ERROR	153	:CS1 INCORRECT
5073	026622	000421				BR	TST31	:GO ON TO NEXT TEST
5074								
5075	026624	023737	003510	003450	22\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
5076	026632	001402				BEQ	23\$:YES, CHECK DATA
5077	026634	104154				ERROR	154	:CS2 INCORRECT
5078	026636	000413				BR	TST31	:GO ON TO NEXT TEST
5079								
5080	026640	023737	003522	003462	23\$:	CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
5081	026646	001401				BEQ	24\$:YES, GET NEXT HEADER WORD
5082	026650	104155				ERROR	155	:DATA INCORRECT
5083	026652	005237	003624		24\$:	INC	WRDCNT	:INCREMENT WORD COUNT
5084	026656	022737	000003	003624		CMP	#3,WRDCNT	:CHECK IF ALL THREE WORDS CHECK
5085	026664	001327				BNE	20\$:NO, GET NEXT WORD

```

5086
5087
5088
5089
5090
5091
5092
5093
5094
5095
5096
5097

```

```

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

```

```

5098      *           052012
5099      *           100520
5100      *           052012
5101      *
5102      *           MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
5103      *           IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
5104      *
5105      * *****
5106      026666 000004 TST31: SCOPE
5107      026670 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS
5108      026676 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5109      026702 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5110      026710 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5111      026716 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
5112      026724 012700 000312 MOV #50,#4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
5113      026730 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5114      026736 012762 000040 000026 MOV #DMD,RKMR1(R2)
5115      026744 005300 DEC R0
5116      026746 001370 BNE 1$
5117      026750 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5118      026756 012762 000040 000026 MOV #DMD,RKMR1(R2)
5119      026764 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT BIT AND
5120      026770 005037 003616 CLR M1.BIT ; PREVIOUS BIT
5121      026774 012700 000341 MOV #225,R0
5122      027000 004737 046164 2$: JSR PC,RDBIT ;SIMULATE SYNCH
5123      027004 005300 DEC R0
5124      027006 001374 BNE 2$
5125      027010 012737 000001 003614 MOV #1,PR.BIT
5126      027016 004737 046164 JSR PC,RDBIT
5127      027022 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
5128      027026 012703 067020 MOV #HEAD5,R3 ;LOAD ADDRESS OF DATA
5129      027032 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
5130      027040 012304 5$: MOV (R3)+,R4 ;GET DATA
5131      027042 012700 000020 MOV #16,R0 ;LOAD BIT COUNT
5132      027046 013737 003614 6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5133      027054 006004 ROR R4 ;GET NEXT BIT
5134      027056 103403 BCS 7$ ;CHECK IF 1
5135      027060 005037 003614 CLR PR.BIT ;NO, ZERO
5136      027064 000403 BR 8$ ;SIMULATE READ DATA
5137
5138      027066 012737 000001 003614 7$: MOV #1,PR.BIT ;ONE
5139      027074 004737 046164 8$: JSR PC,RDBIT ;SIMULATE READ DATA
5140      027100 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
5141      027106 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5142      027114 001417 BEQ 9$ ;YES, SIMULATE NEXT BIT
5143      027116 012737 000003 003624 MOV #3,WRDCNT ;LOAD WORD COUNT
5144      027124 160137 003624 SUB R1,WRDCNT
5145      027130 012737 000020 003622 MOV #16,BITCNT ;LOAD BIT COUNT
5146      027136 160037 003622 SUB R0,BITCNT
5147      027142 104150 ERROR 150 ;CS1 INCORRECT DURING HEADER
5148      027144 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5149      027152 000522 BR TST32 ;GO ON TO NEXT TEST
5150
5151      027154 005300 9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
5152      027156 001333 BNE 6$ ;NO, GET NEXT BIT
5153      027160 005301 DEC R1 ;CHECK IF HEADER FINISHED

```

```

5154 027162 001326          BNE      5$          ;NO, GET NEXT WORD
5155 027164 012700 000004    MOV      #4,RO      ;LOAD COUNT FOR POSTAMBLE
5156 027170 013737 003614 003616 15$:  MOV      PR.BIT,M1.BIT ;STORE LAST BIT
5157 027176 005037 003614    CLR      PR.BIT     ;LOAD NEXT BIT
5158 027202 004737 046164    JSR      PC,RDBIT   ;SIMULATE 1 BIT READ
5159 027206 005300          DEC      RO         ;CHECK IF TIME FOR READY
5160 027210 001367          BNE      15$       ;NO, CONTINUE WITH POSTAMBLE
5161 027212 016237 000000 003440    MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
5162 027220 016237 000010 003450    MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
5163 027226 012737 000224 003500    MOV      #RDY!RDHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5164 027234 012737 000300 003510    MOV      #OR!IR,E.CS2  ;LOAD EXPECTED CS2
5165 027242 023737 003500 003440    CMP      E.CS1,↑.CS1  ;CHECK CS1 CORRECT
5166 027250 001401          BEQ      16$       ;YES, CHECK CS2
5167 027252 104151          ERROR   151        ;CS1 INCORRECT
5168 027254 023737 003510 003450 16$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
5169 027262 001401          BEQ      17$       ;YES, CHECK DATA
5170 027264 104152          ERROR   152        ;CS2 INCORRECT
5171 027266 005037 003624          CLR      WRDCNT     ;INITIALIZE WORD COUNT
5172 027272 012703 067020          MOV      #HEAD5,R3   ;GET ADDRESS OF DATA
5173 027276 012337 003522          MOV      (R3)+,E.DB  ;GET EXPECTED DATA
5174 027302 016237 000024 003462    MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
5175 027310 016237 000000 003440    MOV      RKCS1(R2),↑.CS1 ;STORE COMMAND AND STATUS REG. 1
5176 027316 016237 000010 003450    MOV      RKCS2(R2),↑.CS2 ;STORE COMMAND AND STATUS REG. 2
5177 027324 022737 000002 003624    CMP      #2,WRDCNT   ;CHECK IF LAST WORD IN DATA BUFFER
5178 027332 001003          BNE      21$       ;NO, CHECK CS1
5179 027334 012737 000100 003510    MOV      #IR,E.CS2   ;STORE EXPECTED CS2
5180 027342 023737 003500 003440 21$:  CMP      E.CS1,↑.CS1  ;CHECK CS1 CORRECT
5181 027350 001402          BEQ      22$       ;YES, CHECK CS2
5182 027352 104153          ERROR   153        ;CS1 INCORRECT
5183 027354 000421          BR       TST32     ;GO ON TO NEXT TEST
5184
5185 027356 023737 003510 003450 22$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
5186 027364 001402          BEQ      23$       ;YES, CHECK DATA
5187 027366 104154          ERROR   154        ;CS2 INCORRECT
5188 027370 000413          BR       TST32     ;GO ON TO NEXT TEST
5189
5190 027372 023737 003522 003462 23$:  CMP      E.DB,T.DB   ;CHECK IF DATA CORRECT
5191 027400 001401          BEQ      24$       ;YES, GET NEXT HEADER WORD
5192 027402 104155          ERROR   155        ;DATA INCORRECT
5193 027404 005237 003624          INC      WRDCNT     ;INCREMENT WORD COUNT
5194 027410 022737 000003 003624    CMP      #3,WRDCNT   ;CHECK IF ALL THREE WORDS CHECK
5195 027416 001327          BNE      20$       ;NO, GET NEXT WORD

```

```

5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209

```

```

*****
*TEST 32      READ HEADER IN 18 BIT MODE
*
*      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
*      IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
*      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
*      SIMULATE SECTOR PULSE, 255 ZEROES, A
*      ONE, AND A HEADER CONSISTING OF THE THREE
*      FOLLOWING WORDS:
*
*              177777
*              000000
*              177777

```

```

5210
5211
5212
5213
5214
5215 027420 000004
5216 027422 012737 000144 001200
5217 027430 013702 001270
5218 027434 012762 100000 000000
5219 027442 012762 000040 000026
5220 027450 012762 010025 000000
5221 027456 012700 000312
5222 027462 012762 000440 000026 1$:
5223 027470 012762 000040 000026
5224 027476 005300
5225 027500 001370
5226 027502 012762 000140 000026
5227 027510 012762 000040 000026
5228 027516 005037 003614
5229 027522 005037 003616
5230 027526 012700 000377
5231 027532 004737 046164 2$:
5232 027536 005300
5233 027540 001374
5234 027542 012737 000001 003614
5235 027550 004737 046164
5236 027554 012701 000003
5237 027560 012703 066762
5238 027564 012737 010025 003500
5239 027572 012304 5$:
5240 027574 012700 000020
5241 027600 013737 003614 003616 6$:
5242 027606 006004
5243 027610 103403
5244 027612 005037 003614
5245 027616 000403
5246
5247 027620 012737 000001 003614 7$:
5248 027626 004737 046164 8$:
5249 027632 016237 000000 003440
5250 027640 023737 003500 003440
5251 027646 001417
5252 027650 012737 000003 003624
5253 027656 160137 003624
5254 027662 012737 000020 003622
5255 027670 160037 003622
5256 027674 104156
5257 027676 012762 100000 000000
5258 027704 000522
5259
5260 027706 005300 9$:
5261 027710 001333
5262 027712 005301
5263 027714 001326
5264 027716 012700 000004
5265 027722 013737 003614 003616 15$:

```

```

*****
MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
*****
TST32: SCOPE
MOV #100, $TIMES ; DO 100 ITERATIONS
MOV $BASE, R2 ; LOAD RK611
MOV #CLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #CFMT!RDHEAD, RKCS1(R2) ; ISSUE READ HEADER (24 SECTOR FORMAT)
MOV #50, #4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK, RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD, RKMR1(R2)
DEC R0
1$
BNE
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #255, R0
2$: JSR PC, RDBIT ; SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1, PR.BIT
JSR PC, RDBIT
MOV #3, R1 ; LOAD NUMBER OF WORDS
MOV #HEAD1, R3 ; LOAD ADDRESS OF DATA
MOV #CFMT!RDHEAD, E.CS1 ; LOAD EXPECTED CS1
5$: MOV (R3)+, R4 ; GET DATA
MOV #16, R0 ; LOAD BIT COUNT
6$: MOV PR.BIT, M1.BIT ; STORE PRESENT BIT
ROR R4 ; GET NEXT BIT
BCS 7$ ; CHECK IF 1
CLR PR.BIT ; NO, ZERO
BR 8$ ; SIMULATE READ DATA
7$: MOV #1, PR.BIT ; ONE
8$: JSR PC, RDBIT ; SIMULATE READ DATA
MOV RKCS1(R2), T.CS1 ; READ COMMAND AND STATUS REG. 1
CMP E.CS1, T.CS1 ; CHECK IF CS1 CORRECT
BEQ 9$ ; YES, SIMULATE NEXT BIT
MOV #3, WRDCNT ; LOAD WORD COUNT
SUB R1, WRDCNT
MOV #16, BITCNT ; LOAD BIT COUNT
SUB R0, BITCNT
ERROR 156 ; CS1 INCORRECT DURING HEADER
MOV #CLR, RKCS1(R2) ; CLEAR RK611
BR TST33 ; GO ON TO NEXT TEST
9$: DEC R0 ; CHECK IF READY FOR NEXT WORD
BNE 6$ ; NO, GET NEXT BIT
DEC R1 ; CHECK IF HEADER FINISHED
BNE 5$ ; NO, GET NEXT WORD
MOV #4, R0 ; LOAD COUNT FOR POSTAMBLE
15$: MOV PR.BIT, M1.BIT ; STORE LAST BIT

```

```

5266 027730 005037 003614 CLR PR.BIT ;LOAD NEXT BIT
5267 027734 004737 046164 JSR PC,RDBIT ;READ BIT
5268 027740 005300 DEC R0 ;CHECK IF TIME FOR READY
5269 027742 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
5270 027744 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
5271 027752 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
5272 027760 012737 010224 003500 MOV #CFMT!RDY!RDHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5273 027766 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
5274 027774 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5275 030002 001401 BEQ 16$ ;YES, CHECK CS2
5276 030004 104157 ERROR 157 ;CS1 INCORRECT
5277 030006 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5278 030014 001401 BEQ 17$ ;YES, CHECK DATA
5279 030016 104160 ERROR 160 ;CS2 INCORRECT
5280 030020 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
5281 030024 012703 066762 MOV #HEAD1,R3 ;GET ADDRESS OF DATA
5282 030030 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
5283 030034 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
5284 030042 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
5285 030050 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
5286 030056 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
5287 030064 001003 BNE 21$ ;NO, CHECK CS1
5288 030066 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
5289 030074 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5290 030102 001402 BEQ 22$ ;YES, CHECK CS2
5291 030104 104161 ERROR 161 ;CS1 INCORRECT
5292 030106 000421 BR TST33 ;GO ON TO NEXT TEST
5293
5294 030110 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5295 030116 001402 BEQ 23$ ;YES, CHECK DATA BUFFER
5296 030120 104162 ERROR 162 ;CS2 INCORRECT
5297 030122 000413 BR TST33 ;GO ON TO NEXT TEST
5298
5299 030124 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK DATA BUFFER CORRECT
5300 030132 001401 BEQ 24$ ;YES, GET NEXT WORD
5301 030134 104163 ERROR 163 ;DATA BUFFER INCORRECT
5302 030136 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
5303 030142 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF FINISHED
5304 030150 001327 BNE 20$ ;NO READ NEXT WORD
5305
5306 *****
5307 *TEST 33 SYNCH DETECT IN READ HEADER
5308 *
5309 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5310 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
5311 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5312 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
5313 * SIMULATE SECTOR PULSE AND 350 ZEROES. MAKE
5314 * SURE READY REMAINS RESET AND THE SILO REMAINS
5315 * EMPTY.
5316 *****
5317 †TST33: SCOPE
5318 030152 000004 MOV #100,$TIMES ;DO 100. ITERATIONS
5319 030154 012737 000144 001200 MOV $BASE,R2 ;LOAD RK611 BASE
5320 030162 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5321 030166 012762 100000 000000

```

J08

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T33

02-DEC-77 09:56 PAGE 100
SYNCH DETECT IN READ HEADER

SEQ 0100

```

5322 030174 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
5323 030202 012762 000025 000000      MOV      #RDHEAD,RKCS1(R2) ;ISSUE READ HEAD
5324 030210 012700 000312      MOV      #50.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL READY
5325 030214 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5326 030222 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5327 030230 005300      DEC      RO
5328 030232 001370      BNE      1$
5329 030234 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE TO SECTOR PULSE
5330 030242 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5331 030250 005037 003614      CLR      PR.BIT ;INITIALIZE PRESENT AND
5332 030254 005037 003616      CLR      M1.BIT ;PREVIOUS BIT
5333 030260 012737 000025 003500      MOV      #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
5334 030266 012737 000100 003510      MOV      #IR,E.CS2 ;LOAD EXPECTED CS2
5335 030274 005037 003622      CLR      BITCNT ;SIMULATE 350 ZEROES
5336 030300 004737 046164 2$:      JSR      PC,RDBIT
5337 030304 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE CS1
5338 030312 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE CS2
5339 030320 023737 003500 003440      CMP      E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5340 030326 001402      BEQ      3$ ;YES, CHECK CS2
5341 030330 104164      ERROR   164 ;CS1 INCORRECT
5342 030332 000447      BR       TST34 ;GO ON TO NEXT TEST
5343
5344 030334 023737 003510 003450 3$:      CMP      E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
5345 030342 001402      BEQ      4$ ;YES, CHECK IF SILO EMPTY
5346 030344 104165      ERROR   165 ;CS2 INCORRECT
5347 030346 000441      BR       TST34 ;GO ON TO NEXT TEST
5348 030350 005237 003622 4$:      INC      BITCNT ;INCREMENT BIT COUNT
5349 030354 022737 000536 003622      CMP      #350.,BITCNT ;CHECK IF FINISHED
5350 030362 001346      BNE      2$ ;NO, SIMULATE NEXT ZERO
5351 030364 005762 000024      TST      RKDB(R2) ;READ DATA BUFFER
5352 030370 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE CS1 AND CS2
5353 030376 016237 000010 003450      MOV      RKCS2(R2),T.CS2
5354 030404 012737 100224 003500      MOV      #CERR!RDY!RDHEAD<1C<GO>>,E.CS1 ;LOAD EXPECT CS1
5355 030412 012737 100100 003510      MOV      #DCK!IR,E.CS2 ;LOAD EXPECTED CS2
5356 030420 023737 003500 003440      CMP      E.CS1,T.CS1 ;CHECK FOR CONTROLLER ERROR
5357 030426 001401      BEQ      5$ ;YES, CHECK FOR DATA LATE
5358 030430 104166      ERROR   166 ;CS1 INCORRECT
5359 030432 023737 003510 003450 5$:      CMP      E.CS2,T.CS2 ;CHECK FOR DATA LATE
5360 030440 001401      BEQ      6$ ;YES, CHECK RK611
5361 030442 104167      ERROR   167 ;CS2 INCORRECT
5362 030444 012762 100000 000000 6$:      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611

```

```

5363
5364 *****
5365 *TEST 34 ZERO SYNCH ON READ
5366 *
5367 *
5368 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5369 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5370 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
5371 * SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF
5372 * BIT TIME, A ONE, AND A HEADER CONSISTING OF THE
5373 * THREE FOLLOWING WORDS:
5374 *
5375 * 177777
5376 * 000000
5377 * 177777

```

K08

CZR6CC0 RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 101
T34 ZERO SYNCH ON READ

SEQ 0101

```

5378      ;*      MAKE SURE THAT READY COMES AFTER THE THIRD WORD
5379      ;*      IS TRANSFERRED.  CHECK THE SILO FOR CORRECT CONTENTS.
5380      ;*
5381      ;*
5382      ;*****
5382 030452 000004          TST34: SCOPE
5383 030454 012737 000144 001200  MOV      #100,STIMES      ;;DO 100. ITERATIONS
5384 030462 013702 001270          MOV      $BASE,R2      ;;LOAD RK611 BASE
5385 030466 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
5386 030474 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;;PUT RK611 IN DIAGNOSTIC MODE
5387 030502 012762 000025 000000  MOV      #RDHEAD,RKCS1(R2) ;;ISSUE READ HEADER
5388 030510 012700 000312          MOV      #50,#4+2,RO    ;;ISSUE ENOUGH CLOCKS UNTIL READY
5389 030514 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5390 030522 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5391 030530 005300          DEC      RO
5392 030532 001370          BNE     1$
5393 030534 012762 000140 000026  MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5394 030542 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5395 030550 012762 000440 000026  MOV      #DMD!MCLK,RKMR1(R2) ;SHIFT DATA ONE HALF BIT TIME
5396 030556 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5397 030564 005037 003614          CLR     PR.BIT      ;;INITIALIZE PRESENT BIT AND
5398 030570 005037 003616          CLR     M1.BIT      ;;PREVIOUS BIT
5399 030574 012700 000341          MOV      #225,RO
5400 030600 004737 046164          2$:  JSR     PC,RDBIT    ;;SIMULATE SYNCH
5401 030604 005300          DEC     RO
5402 030606 001374          BNE     2$
5403 030610 012737 000001 003614  MOV      #1,PR.BIT
5404 030616 004737 046164          JSR     PC,RDBIT
5405 030622 012701 000003          MOV      #3,R1      ;;LOAD NUMBER OF WORDS
5406 030626 012703 066762          MOV      #HEAD1,R3   ;;LOAD ADDRESS OF DATA
5407 030632 012737 000025 003500  MOV      #RDHEAD,E.CS1 ;;LOAD EXPECTED CS1
5408 030640 012304          MOV      (R3)+,R4    ;;GET DATA
5409 030642 012700 000020          MOV      #16,RO     ;;LOAD BIT COUNT
5410 030646 013737 003614 003616 6$:  MOV      PR.BIT,M1.BIT ;;STORE PREVIOUS BIT
5411 030654 006004          ROR     R4          ;;GET NEXT BIT
5412 030656 103403          BCS     7$         ;;CHECK IF 1
5413 030660 005037 003614          CLR     PR.BIT     ;;NO, ZERO
5414 030664 000403          BR     8$         ;;SIMULATE READ DATA
5415
5416 030666 012737 000001 003614 7$:  MOV      #1,PR.BIT  ;;ONE
5417 030674 004737 046164          8$:  JSR     PC,RDBIT    ;;SIMULATE READ DATA
5418 030700 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;;READ COMMAND AND STATUS REG. 1
5419 030706 023737 003500 003440  CMP      E.CS1,T.CS1  ;;CHECK IF CS1 CORRECT
5420 030714 001417          BEQ     9$         ;;YES, SIMULATE NEXT BIT
5421 030716 012737 000003 003624  MOV      #3,WRDCNT   ;;LOAD WORD COUNT
5422 030724 160137 003624          SUB     R1,WRDCNT
5423 030730 012737 000020 003622  MOV      #16,BITCNT  ;;LOAD BIT COUNT
5424 030736 160037 003622          SUB     RO,BITCNT
5425 030742 104150          ERROR  150        ;;CS1 INCORRECT DURING HEADER
5426 030744 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
5427 030752 000522          BR     TST35      ;;GO ON TO NEXT TEST
5428
5429 030754 005300          9$:  DEC     RO          ;;CHECK IF READY FOR NEXT WORD
5430 030756 001333          BNE     6$        ;;NO, GET NEXT BIT
5431 030760 005301          DEC     R1        ;;CHECK IF HEADER FINISHED
5432 030762 001326          BNE     5$        ;;NO, GET NEXT WORD
5433 030764 012700 000004          MOV      #4,RO     ;;LOAD COUNT FOR POSTAMBLE

```

```

5434 030770 013737 003614 003616 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
5435 030776 005037 003614 CLR PR.BIT ;LOAD NEXT BIT
5436 031002 004737 046164 JSR PC,RDBIT ;SIMULATE 1 BIT READ
5437 031006 005300 DEC RO ;CHECK IF TIME FOR READY
5438 031010 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
5439 031012 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
5440 031020 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
5441 031026 012737 000224 003500 MOV #RDY!RDHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5442 031034 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
5443 031042 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5444 031050 001401 BEQ 16$ ;YES, CHECK CS2
5445 031052 104151 ERROR 151 ;CS1 INCORRECT
5446 031054 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5447 031062 001401 BEQ 17$ ;YES, CHECK DATA
5448 031064 104152 ERROR 152 ;CS2 INCORRECT
5449 031066 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
5450 031072 012703 066762 MOV #HEAD1,R3 ;GET ADDRESS OF DATA
5451 031076 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
5452 031102 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
5453 031110 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
5454 031116 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
5455 031124 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
5456 031132 001003 BNE 21$ ;NO, CHECK CS1
5457 031134 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
5458 031142 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
5459 031150 001402 BEQ 22$ ;YES, CHECK CS2
5460 031152 104153 ERROR 153 ;CS1 INCORRECT
5461 031154 000421 BR TST35 ;GO ON TO NEXT TEST
5462
5463 031156 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
5464 031164 001402 BEQ 23$ ;YES, CHECK DATA
5465 031166 104154 ERROR 154 ;CS2 INCORRECT
5466 031170 000413 BR TST35 ;GO ON TO NEXT TEST
5467
5468 031172 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
5469 031200 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
5470 031202 104155 ERROR 155 ;DATA INCORRECT
5471 031204 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
5472 031210 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
5473 031216 001327 BNE 20$ ;NO, GET NEXT WORD
5474

```

.SBTTL **MFM WRITE LOOPBACK TESTS

```

*****
*TEST 35 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
* INDEX PULSE AND 500 DATA BITS. MAKE SURE THAT
* ZEROES ARE WRITTEN. SIMULATE SECTOR PULSE AND MAKE SURE
* WRITE GATE RESETS.
*****
TST35: SCOPE

```

5489 031220 000004

MO8

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T35

02-DEC-77 09:56 PAGE 103
WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER

SEQ 0103

5490	031222	012737	000144	001200		MOV	#100., \$TIMES	;; DO 100. ITERATIONS
5491	031230	013702	001270			MOV	\$BASE, R2	;; LOAD RK611 BASE
5492	031234	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	;; CLEAR RK611
5493	031242	012762	000040	000026		MOV	#DMD, RKMR1(R2)	;; PUT RK611 IN DIAGNOSTIC MODE
5494	031250	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	;; INITIALIZE ROM
5495	031256	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5496	031264	012762	067264	000004		MOV	#WRBUFF, RKBA(R2)	;; ISSUE WRITE HEADER
5497	031272	012762	177777	000002		MOV	#-1, RKWC(R2)	
5498	031300	012762	000027	000000		MOV	#WRHEAD, RKCS1(R2)	
5499	031306	012700	002364			MOV	#<256.+48.+64.+256.+10.>*2, R0	;; ISSUE ENOUGH CLOCKS
5500								;; UNTIL READY FOR INDEX PULSE
5501	031312	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)	
5502	031320	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5503	031326	005300				DEC	R0	
5504	031330	001370				BNE	1\$	
5505	031332	012700	000004			MOV	#4., R0	;; SIMULATE INDEX PULSE
5506	031336	012762	000240	000026		MOV	#MIND!DMD, RKMR1(R2)	
5507	031344	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
5508	031352	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
5509	031360	005300				DEC	R0	
5510	031362	001370				BNE	2\$	
5511	031364	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5512	031372	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT, E.MR1	;; INITIALIZE EXPECTED
5513								;; MAINT. REG
5514	031400	012700	000002			MOV	#2, R0	;; WAIT FOR WRITE GATE
5515	031404	012762	000440	000026	3\$:	MOV	#DMD!MCLK, RKMR1(R2)	
5516	031412	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
5517	031420	005300				DEC	R0	
5518	031422	001370				BNE	3\$	
5519	031424	005037	003626			CLR	SECCNT	;; CLEAR SECTOR COUNT
5520	031430	005037	003612			CLR	P1.BIT	;; INITIALIZE BIT GENERATION
5521	031434	005037	003614			CLR	PR.BIT	
5522	031440	005037	003616			CLR	M1.BIT	
5523	031444	005037	003620			CLR	M2.BIT	
5524	031450	012700	000764			MOV	#500., R0	;; LOAD COUNT FOR 500 BITS
5525	031454	012737	062473	003170		MOV	#EM230, EMW	;; LOAD ERROR MESSAGE
5526	031462	005037	003622			CLR	BITCNT	;; CLEAR BIT COUNT
5527	031466	004737	045214		5\$:	JSR	PC, WRTBIT	;; WRITE ONE BIT
5528	031472	104170				ERROR	170	;; ERROR IN WRITE
5529	031474	005237	003622			INC	BITCNT	;; INCREMENT NUMBER OF BITS WRITTEN
5530	031500	005300				DEC	R0	;; CHECK IF FINISHED
5531	031502	001371				BNE	5\$;; NO, CONTINUE
5532	031504	042737	040000	003524		BIC	#WRTGAT, E.MR1	;; GENERATE EXPECTED MR1
5533	031512	052737	000100	003524		BIS	#MSP, E.MR1	
5534	031520	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	;; RAISE SECTOR PULSE
5535	031526	016237	000026	003464		MOV	RKMR1(R2), T.MR1	;; STORE MAINT. REG. 1
5536	031534	023737	003524	003464		CMP	E.MR1, T.MR1	;; STORE MAINT REG 1
5537	031542	001401				BEQ	10\$;; YES, LOWER SECTOR PULSE
5538	031544	104171				ERROR	171	;; WRITE GATE DID NOT RESET
5539	031546	042737	000100	003524	10\$:	BIC	#MSP, E.MR1	;; GENERATE EXPECTED MR1
5540	031554	052737	040000	003524		BIS	#WRTGAT, E.MR1	
5541	031562	012762	000040	000026		MOV	#DMD, RKMR1(R2)	;; RESET SECTOR PULSE
5542	031570	016237	000026	003464		MOV	RKMR1(R2), T.MR1	;; STORE MAINT REG 1
5543	031576	023737	003524	003464		CMP	E.MR1, T.MR1	;; CHECK MR1 CORRECT
5544	031604	001401				BEQ	TST36	;; YES, GO ON TO NEXT TEST
5545	031606	104172				ERROR	172	;; WRITE GATE DID NOT SET

5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557
5558
5559
5560
5561
5562
5563
5564
5565
5566
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601

031610	000004		
031612	012737	000144	001200
031620	013702	001270	
031624	012762	100000	000000
031632	012762	000040	000026
031640	012762	066762	000004
031646	012762	177775	000002
031654	012762	000027	000000
031662	012700	000312	
031666	012762	000440	000026
031674	012762	000040	000026
031702	005300		
031704	001370		
031706	012700	000004	
031712	012762	000240	000026
031720	012762	000640	000026
031726	012762	000240	000026
031734	005300		
031736	001370		
031740	012762	000040	000026
031746	012700	000010	
031752	012762	000440	000026
031760	012762	000040	000026
031766	005300		
031770	001370		
031772	012762	000140	000026
032000	012762	000040	000026
032006	005037	003626	
032012	012737	063045	003170
032020	012737	062040	003524
032026	005037	003612	
032032	005037	003614	
032036	005037	003616	
032042	005037	003620	
032046	012700	000400	

```

*****
*TEST 36      WRITE LOOPBACK (PART 1)
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE
* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
* CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:
*
*          177777
*          000000
*          177777
*
* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
*
*****
TST36:  SCOPE
        MOV      #100, $TIMES      ;; DO 100. ITERATIONS
        MOV      $BASE, R2        ;; LOAD RK611 BASE
        MOV      #CCLR, RKCS1(R2) ;; CLEAR RK611
        MOV      #DMD, RKMR1(R2)  ;; PUT RK611 IN DIAGNOSTIC MODE
        MOV      #HEAD1, RKBA(R2) ;; ISSUE WRITE HEADER
        MOV      #-3, RKWC(R2)
        MOV      #WRHEAD, RKCS1(R2)
        MOV      #50, #4+2, R0    ;; ISSUE ENOUGH CLOCKS UNTIL
                                ;; READY FOR INDEX PULSE
1$:     MOV      #DMD!MCLK, RKMR1(R2)
        MOV      #DMD, RKMR1(R2)
        DEC     R0
        BNE    1$
        MOV      #4, R0          ;; ISSUE INDEX PULSE
        MOV      #DMD!MIND, RKMR1(R2)
2$:     MOV      #DMD!MIND!MCLK, RKMR1(R2)
        MOV      #DMD!MIND, RKMR1(R2)
        DEC     R0
        BNE    2$
        MOV      #DMD, RKMR1(R2)
        MOV      #8, R0          ;; WAIT FOR WRITE GATE
3$:     MOV      #DMD!MCLK, RKMR1(R2)
        MOV      #DMD, RKMR1(R2)
        DEC     R0
        BNE    3$
        MOV      #DMD!MSP, RKMR1(R2) ;; SIMULATE SECTOR PULSE
        MOV      #DMD, RKMR1(R2)
        CLR     SECCNT          ;; INITIALIZE SECTOR COUNT
        MOV      #EM233, EMW    ;; LOAD ERROR MESSAGE
        MOV      #DMD!MEWD!ECCW!WRTGAT, E.MR1 ;; INITIALIZE EXPECTED
                                ;; MAINT REG 1
        CLR     P1.BIT          ;; INITIALIZE BIT GENERATION
        CLR     PR.BIT
        CLR     M1.BIT
        CLR     M2.BIT
        MOV     #256., R0      ;; SIMULATE SYNCH

```

5602	032052	005037	003622			CLR	BITCNT	; INITIALIZE BIT COUNT
5603	032056	004737	045214	5\$:		JSR	PC,WRTBIT	; WRITE ONE BIT
5604	032062	104170				ERROR	170	; DATA INCORRECT
5605	032064	005237	003622			INC	BITCNT	
5606	032070	005300				DEC	R0	; CHECK IF READY FOR DATA
5607	032072	001371				BNE	5\$; NO, GENERATE NEXT BIT
5608	032074	012737	000001	003612		MOV	#1,P1.BIT	; PUT IN SYNCH BIT
5609	032102	004737	045214			JSR	PC,WRTBIT	
5610	032106	104170				ERROR	170	; DATA INCORRECT
5611	032110	005037	003622			CLR	BITCNT	; INITIALIZE BIT COUNT
5612	032114	012737	063111	003170		MOV	#EM234,EMW	; LOAD ERROR MESSAGE
5613	032122	012703	066762			MOV	#HEAD1,R3	; LOAD ADDRESS OF DATA
5614	032126	012700	000003			MOV	#3,R0	; LOAD NUMBER WORDS IN HEADER
5615	032132	012304		10\$:		MOV	(R3)+,R4	; GET NEXT WORD
5616	032134	012701	000020			MOV	#16,R1	; LOAD BIT COUNT
5617	032140	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	; SHIFT BITS
5618	032146	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5619	032154	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5620	032162	006004				ROR	R4	; SHIFT IN NEXT BIT
5621	032164	103403				BCS	14\$; CHECK IF ONE
5622	032166	005037	003612			CLR	P1.BIT	; ZERO
5623	032172	000403				BR	15\$; CLOCK IN BIT
5624								
5625	032174	012737	000001	003612	14\$:	MOV	#1,P1.BIT	; ONE
5626	032202	004737	045214		15\$:	JSR	PC,WRTBIT	; WRITE BIT
5627	032206	104170				ERROR	170	; BIT INCORRECT
5628	032210	005237	003622			INC	BITCNT	; INCREMENT BIT COUNT
5629	032214	005301				DEC	R1	; CHECK IF WORD FINISHED
5630	032216	001350				BNE	12\$; NO, CONTINUE WITH NEXT BIT
5631	032220	005300				DEC	R0	; CHECK IF HEADER COMPLETE
5632	032222	001343				BNE	10\$; NO, GET NEXT WORD
5633	032224	012701	000020			MOV	#16,R1	; LOAD BIT COUNT FOR NEXT WORD
5634	032230	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	; SHIFT BITS
5635	032236	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5636	032244	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5637	032252	005037	003612			CLR	P1.BIT	
5638	032256	004737	045214			JSR	PC,WRTBIT	; WRITE ZERO
5639	032262	104170				ERROR	170	; BIT INCORRECT
5640	032264	005237	003622			INC	BITCNT	; INCREMENT BIT COUNT
5641	032270	005301				DEC	R1	; CHECK IF FINISHED
5642	032272	001356				BNE	18\$; NO, CLOCK NEXT BIT
5643	032274	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	; SIMULATE INDEX
5644	032302	012700	000004			MOV	#4,R0	
5645	032306	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
5646	032314	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5647	032322	005300				DEC	R0	
5648	032324	001370				BNE	20\$	
5649	032326	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5650	032334	016237	000026	003464		MOV	RKMR1(R2),T.MR1	; GET MAINT REG 1
5651	032342	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	; LOAD EXPECTED MR1
5652	032350	023737	003524	003464		CMP	E.MR1,T.MR1	; CHECK MR1 CORRECT (WRITE GATE RESET)
5653	032356	001401				BEQ	25\$; YES, CHECK IF READY SET
5654	032360	104173				ERROR	173	; MAINT REG 1 INCORRECT
5655	032362	012700	000010		25\$:	MOV	#8,R0	; FINISH COMMAND
5656	032366	012762	000440	000026	26\$:	MOV	#DMD!MCLK,RKMR1(R2)	
5657	032374	012762	000040	000026		MOV	#DMD,RKMR1(R2)	

5658	032402	005300			DEC	RO	
5659	032404	001370			BNE	26\$	
5660	032406	016237	000000	003440	MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
5661	032414	012737	000226	003500	MOV	#RDY,WRHEAD&<TC<GO>>	;E.CS1 ;LOAD EXPECTED CS1
5662	032422	023737	003500	003440	CMP	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
5663	032430	001401			BEG	TST37	;YES, GO ON TO NEXT TEST
5664	032432	104174			ERROR	174	;CS1 INCORRECT

*TEST 37 WRITE LOOPBACK (PART 2)

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

000000
177777
000000

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

*TST37: SCOPE

5685	032434	000004			MOV	#100,\$TIMES	;DO 100. ITERATIONS
5686	032436	012737	000144	001200	MOV	\$BASE,R2	;LOAD RK611 BASE
5687	032444	013702	001270		MOV	#CLR,RKCS1(R2)	;CLEAR RK611
5688	032450	012762	100000	000000	MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
5689	032456	012762	000040	000026	MOV	#HEAD2,RKBA(R2)	;ISSUE WRITE HEADER
5690	032464	012762	066770	000004	MOV	#-3,RKWC(R2)	
5691	032472	012762	177775	000002	MOV	#WRHEAD,RKCS1(R2)	
5692	032500	012762	000027	000000	MOV	#50,#4+2,RO	;ISSUE ENOUGH CLOCKS UNTIL READY FOR INDEX PULSE
5693	032506	012700	000312				
5694							
5695	032512	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)
5696	032520	012762	000040	000026		MOV	#DMD,RKMR1(R2)
5697	032526	005300				DEC	RO
5698	032530	001370				BNE	1\$
5699	032532	012700	000004			MOV	#4,RO ;ISSUE INDEX PULSE
5700	032536	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)
5701	032544	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)
5702	032552	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)
5703	032560	005300				DEC	RO
5704	032562	001370				BNE	2\$
5705	032564	012762	000040	000026		MOV	#DMD,RKMR1(R2)
5706	032572	012700	000010			MOV	#8,RO ;WAIT FOR WRITE GATE
5707	032576	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)
5708	032604	012762	000040	000026		MOV	#DMD,RKMR1(R2)
5709	032612	005300				DEC	RO
5710	032614	001370				BNE	3\$
5711	032616	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5712	032624	012762	000040	000026		MOV	#DMD,RKMR1(R2)
5713	032632	005037	003626			CLR	SECCNT ;INITIALIZE SECTOR COUNT

```

5714 032636 012737 063045 003170 MOV #EM233,EMW ;LOAD ERROR MESSAGE
5715 032644 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5716 ; MAINT REG 1
5717 032652 005037 003612 CLR P1.BIT ;INITIALIZE BIT GENERATION
5718 032658 005037 003614 CLR PR.BIT
5719 032662 005037 003616 CLR M1.BIT
5720 032666 005037 003620 CLR M2.BIT
5721 032672 012700 000400 MOV #256,R0 ;SIMULATE SYNCH
5722 032676 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5723 032702 004737 045214 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5724 032706 104170 ERROR 170 ;DATA INCORRECT
5725 032710 005237 003622 INC BITCNT
5726 032714 005300 DEC R0 ;CHECK IF READY FOR DATA
5727 032716 001371 BNE 5$ ;NO, GENERATE NEXT BIT
5728 032720 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
5729 032726 004737 045214 JSR PC,WRTBIT
5730 032732 104170 ERROR 170 ;DATA INCORRECT
5731 032734 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5732 032740 012737 063111 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5733 032746 012703 066770 MOV #HEAD2,R3 ;LOAD ADDRESS OF DATA
5734 032752 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5735 032756 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5736 032760 012701 000020 MOV #16,R1 ;LOAD BIT COUNT
5737 032764 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5738 032772 013737 003614 003616 MOV PR.BIT,M1.BIT
5739 033000 013737 003612 003614 MOV P1.BIT,PR.BIT
5740 033006 006004 ROR R4 ;SHIFT IN NEXT BIT
5741 033010 103403 BCS 14$ ;CHECK IF ONE
5742 033012 005037 003612 CLR P1.BIT ;ZERO
5743 033016 000403 BR 15$ ;CLOCK IN BIT
5744
5745 033020 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5746 033026 004737 045214 15$: JSR PC,WRTBIT ;WRITE BIT
5747 033032 104170 ERROR 170 ;BIT INCORRECT
5748 033034 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5749 033040 005301 DEC R1 ;CHECK IF WORD FINISHED
5750 033042 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
5751 033044 005300 DEC R0 ;CHECK IF HEADER COMPLETE
5752 033046 001343 BNE 10$ ;NO, GET NEXT WORD
5753 033050 012701 000020 MOV #16,R1 ;LOAD BIT COUNT FOR NEXT WORD
5754 033054 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5755 033062 013737 003614 003616 MOV PR.BIT,M1.BIT
5756 033070 013737 003612 003614 MOV P1.BIT,PR.BIT
5757 033076 005037 003612 CLR P1.BIT
5758 033102 004737 045214 JSR PC,WRTBIT ;WRITE ZERO
5759 033106 104170 ERROR 170 ;BIT INCORRECT
5760 033110 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5761 033114 005301 DEC R1 ;CHECK IF FINISHED
5762 033116 001356 BNE 18$ ;NO, CLOCK NEXT BIT
5763 033120 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5764 033126 012700 000004 MOV #4,R0
5765 033132 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5766 033140 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5767 033146 005300 DEC R0
5768 033150 001370 BNE 20$
5769 033152 012762 000040 000026 MOV #DMD,RKMR1(R2)

```

```

5770 033160 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
5771 033166 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5772 033174 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5773 033202 001401 BEQ 25$ ;YES, CHECK IF READY SET
5774 033204 104173 ERROR 173 ;MAINT REG 1 INCORRECT
5775 033206 012700 000010 25$: MOV #8,R0 ;FINISH COMMAND
5776 033212 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5777 033220 012762 000040 000026 MOV #DMD,RKMR1(R2)
5778 033226 005300 DEC R0
5779 033230 001370 BNE 26$
5780 033232 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5781 033240 012737 000226 003500 MOV #RDY!WRHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5782 033246 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5783 033254 001401 BEQ TST40 ;YES, GO ON TO NEXT TEST
5784 033256 104174 ERROR 174 ;CS1 INCORRECT

```

```

*****
*TEST 40 WRITE LOOPBACK (PART 3)

```

```

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

```

```

* 125252
* 052525
* 125252

```

```

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

```

```

*****
TST40: SCOPE
5804 033260 000004 ST40: MOV #100,$TIMES ;DO 100. ITERATIONS
5805 033262 012737 000144 001200 MOV $BASE,R2 ;LOAD RK611 BASE
5806 033270 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5807 033274 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5808 033302 012762 000040 000026 MOV #HEAD3,RKBA(R2) ;ISSUE WRITE HEADER
5809 033310 012762 066776 000004 MOV #-3,RKWC(R2)
5810 033316 012762 177775 000002 MOV #WRHEAD,RKCS1(R2)
5811 033324 012762 000027 000000 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5812 033332 012700 000312 ;READY FOR INDEX PULSE
5813
5814 033336 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5815 033344 012762 000040 000026 MOV #DMD,RKMR1(R2)
5816 033352 005300 DEC R0
5817 033354 001370 BNE 1$
5818 033356 012700 000004 MOV #4,R0 ;ISSUE INDEX PULSE
5819 033362 012762 000240 000026 2$: MOV #DMD!MIND,RKMR1(R2)
5820 033370 012762 000640 000026 MOV #DMD!MIND!MCLK,RKMR1(R2)
5821 033376 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5822 033404 005300 DEC R0
5823 033406 001370 BNE 2$
5824 033410 012762 000040 000026 MOV #DMD,RKMR1(R2)
5825 033416 012700 000010 MOV #8.,R0 ;WAIT FOR WRITE GATE

```

5826	033422	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)	
5827	033430	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5828	033436	005300				DEC	R0	
5829	033440	001370				BNE	3\$	
5830	033442	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2);SIMULATE SECTOR PULSE	
5831	033450	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5832	033456	005037	003626			CLR	SECCNT	;INITIALIZE SECTOR COUNT
5833	033462	012737	063045	003170		MOV	#EM233,EMW	;LOAD ERROR MESSAGE
5834	033470	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1	;INITIALIZE EXPECTED
5835								;MAINT REG 1
5836	033476	005037	003612			CLR	P1.BIT	;INITIALIZE BIT GENERATION
5837	033502	005037	003614			CLR	PR.BIT	
5838	033506	005037	003616			CLR	M1.BIT	
5839	033512	005037	003620			CLR	M2.BIT	
5840	033516	012700	000400			MOV	#256,R0	;SIMULATE SYNCH
5841	033522	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5842	033526	004737	045214		5\$:	JSR	PC,WRTBIT	;WRITE ONE BIT
5843	033532	104170				ERROR	170	;DATA INCORRECT
5844	033534	005237	003622			INC	BITCNT	
5845	033540	005300				DEC	R0	;CHECK IF READY FOR DATA
5846	033542	001371				BNE	5\$;NO, GENERATE NEXT BIT
5847	033544	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
5848	033552	004737	045214			JSR	PC,WRTBIT	
5849	033556	104170				ERROR	170	;DATA INCORRECT
5850	033560	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5851	033564	012737	063111	003170		MOV	#EM234,EMW	;LOAD ERROR MESSAGE
5852	033572	012703	066776			MOV	#HEAD3,R3	;LOAD ADDRESS OF DATA
5853	033576	012700	000003			MOV	#3,R0	;LOAD NUMBER WORDS IN HEADER
5854	033602	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
5855	033604	012701	000020			MOV	#16,R1	;LOAD BIT COUNT
5856	033610	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5857	033616	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5858	033624	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5859	033632	006004				ROR	R4	;SHIFT IN NEXT BIT
5860	033634	103403				BCS	14\$;CHECK IF ONE
5861	033636	005037	003612			CLR	P1.BIT	;ZERO
5862	033642	000403				BR	15\$;CLOCK IN BIT
5863								
5864	033644	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
5865	033652	004737	045214		15\$:	JSR	PC,WRTBIT	;WRITE BIT
5866	033656	104170				ERROR	170	;BIT INCORRECT
5867	033660	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
5868	033664	005301				DEC	R1	;CHECK IF WORD FINISHED
5869	033666	001350				BNE	12\$;NO, CONTINUE WITH NEXT BIT
5870	033670	005300				DEC	R0	;CHECK IF HEADER COMPLETE
5871	033672	001343				BNE	10\$;NO, GET NEXT WORD
5872	033674	012701	000020			MOV	#16,R1	;LOAD BIT COUNT FOR NEXT WORD
5873	033700	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5874	033706	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5875	033714	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5876	033722	005037	003612			CLR	P1.BIT	
5877	033726	004737	045214			JSR	PC,WRTBIT	;WRITE ZERO
5878	033732	104170				ERROR	170	;BIT INCORRECT
5879	033734	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
5880	033740	005301				DEC	R1	;CHECK IF FINISHED
5881	033742	001356				BNE	18\$;NO, CLOCK NEXT BIT

```

5882 033744 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5883 033752 012700 000004 MOV #4,RO
5884 033756 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5885 033764 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5886 033772 005300 DEC RO
5887 033774 001370 BNE 20$
5888 033776 012762 000040 000026 MOV #DMD,RKMR1(R2)
5889 034004 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
5890 034012 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5891 034020 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5892 034026 001401 BEQ 25$ ;YES, CHECK IF READY SET
5893 034030 104173 ERROR 173 ;MAINT REG 1 INCORRECT
5894 034032 012700 000010 25$: MOV #8,RO ;FINISH COMMAND
5895 034036 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5896 034044 012762 000040 000026 MOV #DMD,RKMR1(R2)
5897 034052 005300 DEC RO
5898 034054 001370 BNE 26$
5899 034056 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5900 034064 012737 000226 003500 MOV #RDY!WRHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5901 034072 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5902 034100 001401 BEQ TST41 ;YES, GO ON TO NEXT TEST
5903 034102 104174 ERROR 174 ;CS1 INCORRECT

```

*TEST 41 WRITE LOOPBACK (PART 4)

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

* 044444
* 022222
* 111111

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

```

5923 034104 000004 TST41: SCOPE
5924 034106 012737 MOV #100,$TIMES ;DO 100. ITERATIONS
5925 034114 013702 MOV $BASE,R2 ;LOAD RK611 BASE
5926 034120 012762 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5927 034126 012762 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5928 034134 012762 MOV #HEAD4,RKBA(R2) ;ISSUE WRITE HEADER
5929 034142 012762 MOV #-3,RKWC(R2)
5930 034150 012762 MOV #WRHEAD,RKCS1(R2)
5931 034156 012700 MOV #50.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
5932
5933 034162 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5934 034170 012762 000040 000026 MOV #DMD,RKMR1(R2)
5935 034176 005300 DEC RO
5936 034200 001370 BNE 1$
5937 034202 012700 000004 MOV #4,RO ;ISSUE INDEX PULSE

```

5938	034206	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5939	034214	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
5940	034222	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5941	034230	005300				DEC	R0	
5942	034232	001370				BNE	2\$	
5943	034234	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5944	034242	012700	000010			MOV	#8,R0 ;WAIT FOR WRITE GATE	
5945	034246	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)	
5946	034254	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5947	034262	005300				DEC	R0	
5948	034264	001370				BNE	3\$	
5949	034266	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE	
5950	034274	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5951	034302	005037	003626			CLR	SECCNT ;INITIALIZE SECTOR COUNT	
5952	034306	012737	063045	003170		MOV	#EM233,EMW ;LOAD ERROR MESSAGE	
5953	034314	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED	
5954							MAINT REG 1	
5955	034322	005037	003612			CLR	P1.BIT ;INITIALIZE BIT GENERATION	
5956	034326	005037	003614			CLR	PR.BIT	
5957	034332	005037	003616			CLR	M1.BIT	
5958	034336	005037	003620			CLR	M2.BIT	
5959	034342	012700	000400			MOV	#256,R0 ;SIMULATE SYNCH	
5960	034346	005037	003622			CLR	BITCNT ;INITIALIZE BIT COUNT	
5961	034352	004737	045214		5\$:	JSR	PC,WRTBIT ;WRITE ONE BIT	
5962	034356	104170				ERROR	170 ;DATA INCORRECT	
5963	034360	005237	003622			INC	BITCNT	
5964	034364	005300				DEC	R0	
5965	034366	001371				BNE	5\$;CHECK IF READY FOR DATA	
5966	034370	012737	000001	003612		MOV	#1,P1.BIT ;NO, GENERATE NEXT BIT	
5967	034376	004737	045214			JSR	PC,WRTBIT ;PUT IN SYNCH BIT	
5968	034402	104170				ERROR	170 ;DATA INCORRECT	
5969	034404	005037	003622			CLR	BITCNT ;INITIALIZE BIT COUNT	
5970	034410	012737	063111	003170		MOV	#EM234,EMW ;LOAD ERROR MESSAGE	
5971	034416	012703	067012			MOV	#HEAD4,R3 ;LOAD ADDRESS OF DATA	
5972	034422	012700	000003			MOV	#3,R0 ;LOAD NUMBER WORDS IN HEADER	
5973	034426	012304			10\$:	MOV	(R3)+,R4 ;GET NEXT WORD	
5974	034430	012701	000020			MOV	#16,R1 ;LOAD BIT COUNT	
5975	034434	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT ;SHIFT BITS	
5976	034442	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5977	034450	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5978	034456	006004				ROR	R4 ;SHIFT IN NEXT BIT	
5979	034460	103403				BCS	14\$;CHECK IF ONE	
5980	034462	005037	003612			CLR	P1.BIT ;ZERO	
5981	034466	000403				BR	15\$;CLOCK IN BIT	
5982								
5983	034470	012737	000001	003612	14\$:	MOV	#1,P1.BIT ;ONE	
5984	034476	004737	045214		15\$:	JSR	PC,WRTBIT ;WRITE BIT	
5985	034502	104170				ERROR	170 ;BIT INCORRECT	
5986	034504	005237	003622			INC	BITCNT ;INCREMENT BIT COUNT	
5987	034510	005301				DEC	R1 ;CHECK IF WORD FINISHED	
5988	034512	001350				BNE	12\$;NO, CONTINUE WITH NEXT BIT	
5989	034514	005300				DEC	R0 ;CHECK IF HEADER COMPLETE	
5990	034516	001343				BNE	10\$;NO, GET NEXT WORD	
5991	034520	012701	000020			MOV	#16,R1 ;LOAD BIT COUNT FOR NEXT WORD	
5992	034524	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT ;SHIFT BITS	
5993	034532	013737	003614	003616		MOV	PR.BIT,M1.BIT	

5994	034540	013737	003612	003614	MOV	P1.BIT,PR.BIT	
5995	034546	005037	003612		CLR	P1.BIT	
5996	034552	004737	045214		JSR	PC,WRTBIT	;WRITE ZERO
5997	034556	104170			ERROR	170	;BIT INCORRECT
5998	034560	005237	003622		INC	BITCNT	;INCREMENT BIT COUNT
5999	034564	005301			DEC	R1	;CHECK IF FINISHED
6000	034566	001356			BNE	18\$;NO, CLOCK NEXT BIT
6001	034570	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	;SIMULATE INDEX
6002	034576	012700	000004		MOV	#4,R0	
6003	034602	012762	000640	000026	20\$:	MOV	#DMD:MIND:MCLK,RKMR1(R2)
6004	034610	012762	000240	000026	MOV	#DMD:MIND,RKMR1(R2)	
6005	034616	005300			DEC	R0	
6006	034620	001370			BNE	20\$	
6007	034622	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
6008	034630	016237	000026	003464	MOV	RKMR1(R2),T.MR1	;GET MAINT REG 1
6009	034636	012737	022040	003524	MOV	#MEWD:ECCW:DMD,E.MR1	;LOAD EXPECTED MR1
6010	034644	023737	003524	003464	MOV	E.MR1,T.MR1	;CHECK MR1 CORRECT (WRITE GATE RESET)
6011	034652	001401			BEQ	25\$;YES, CHECK IF READY SET
6012	034654	104173			ERROR	173	;MAINT REG 1 INCORRECT
6013	034656	012700	000010		25\$:	MOV	#8,R0
6014	034662	012762	000440	000026	26\$:	MOV	#DMD:MCLK,RKMR1(R2)
6015	034670	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
6016	034676	005300			DEC	R0	
6017	034700	001370			BNE	26\$	
6018	034702	016237	000000	003440	MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
6019	034710	012737	000226	003500	MOV	#RDY:WRHEAD<↑C<GO>>,E.CS1	;LOAD EXPECTED CS1
6020	034716	023737	003500	003440	MOV	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
6021	034724	001401			BEQ	TST42	;YES, GO ON TO NEXT TEST
6022	034726	104174			ERROR	174	;CS1 INCORRECT

```

*****
*TEST 42          WRITE LOOPBACK (PART 5)
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
* IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE
* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
* CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:
*
*           052012
*           100520
*           052012
*
* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.
* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
*
*****

```

6042	034730	000004			↑ST42: SCOPE		
6043	034732	012737	000144	001200	MOV	#100,\$TIMES	;DO 100. ITERATIONS
6044	034740	013702	001270		MOV	\$BASE,R2	;LOAD RK611 BASE
6045	034744	012762	100000	000000	MOV	#CLR,RKCS1(R2)	;CLEAR RK611
6046	034752	012762	000040	000026	MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
6047	034760	012762	067020	000004	MOV	#HEADS,RKBA(R2)	;ISSUE WRITE HEADER
6048	034766	012762	177775	000002	MOV	#-3,RKWC(R2)	
6049	034774	012762	000027	000000	MOV	#WRHEAD,RKCS1(R2)	

6050	035002	012700	000312			MOV	#50.*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL
6051								;READY FOR INDEX PULSE
6052	035006	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6053	035014	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6054	035022	005300				DEC	R0	
6055	035024	001370				BNE	1\$	
6056	035026	012700	000004			MOV	#4,R0	;ISSUE INDEX PULSE
6057	035032	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6058	035040	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6059	035046	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6060	035054	005300				DEC	R0	
6061	035056	001370				BNE	2\$	
6062	035060	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6063	035066	012700	000010			MOV	#8,R0	;WAIT FOR WRITE GATE
6064	035072	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6065	035100	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6066	035106	005300				DEC	R0	
6067	035110	001370				BNE	3\$	
6068	035112	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
6069	035120	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6070	035126	005037	003626			CLR	SECCNT	;INITIALIZE SECTOR COUNT
6071	035132	012737	063045	003170		MOV	#EM233,EMW	;LOAD ERROR MESSAGE
6072	035140	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1	;INITIALIZE EXPECTED
6073								;MAINT REG 1
6074	035146	005037	003612			CLR	P1.BIT	;INITIALIZE BIT GENERATION
6075	035152	005037	003614			CLR	PR.BIT	
6076	035156	005037	003616			CLR	M1.BIT	
6077	035162	005037	003620			CLR	M2.BIT	
6078	035166	012700	000400			MOV	#256,R0	;SIMULATE SYNCH
6079	035172	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6080	035176	004737	045214		5\$:	JSR	PC,WRTBIT	;WRITE ONE BIT
6081	035202	104170				ERROR	170	;DATA INCORRECT
6082	035204	005237	003622			INC	BITCNT	
6083	035210	005300				DEC	R0	;CHECK IF READY FOR DATA
6084	035212	001371				BNE	5\$;NO GENERATE NEXT BIT
6085	035214	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
6086	035222	004737	045214			JSR	PC,WRTBIT	
6087	035226	104170				ERROR	170	;DATA INCORRECT
6088	035230	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6089	035234	012737	063111	003170		MOV	#EM234,EMW	;LOAD ERROR MESSAGE
6090	035242	012703	067020			MOV	#HEAD5,R3	;LOAD ADDRESS OF DATA
6091	035246	012700	000003			MOV	#3,R0	;LOAD NUMBER WORDS IN HEADER
6092	035252	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
6093	035254	012701	000020			MOV	#16,R1	;LOAD BIT COUNT
6094	035260	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6095	035266	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6096	035274	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6097	035302	006004				ROR	R4	;SHIFT IN NEXT BIT
6098	035304	103403				BCS	14\$;CHECK IF ONE
6099	035306	005037	003612			CLR	P1.BIT	;ZERO
6100	035312	000403				BR	15\$;CLOCK IN BIT
6101								
6102	035314	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
6103	035322	004737	045214		15\$:	JSR	PC,WRTBIT	;WRITE BIT
6104	035326	104170				ERROR	170	;BIT INCORRECT
6105	035330	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT

6106	035334	005301				DEC	R1		;CHECK IF WORD FINISHED
6107	035336	001350				BNE	12\$;NO, CONTINUE WITH NEXT BIT
6108	035340	005300				DEC	RO		;CHECK IF HEADER COMPLETE
6109	035342	001343				BNE	10\$;NO, GET NEXT WORD
6110	035344	012701	000020			MOV	#16, R1		;LOAD BIT COUNT FOR NEXT WORD
6111	035350	013737	003616	003620	18\$:	MOV	M1.BIT, M2.BIT		;SHIFT BITS
6112	035356	013737	003614	003616		MOV	PR.BIT, M1.BIT		
6113	035364	013737	003612	003614		MOV	P1.BIT, PR.BIT		
6114	035372	005037	003612			CLR	P1.BIT		
6115	035376	004737	045214			JSR	PC WRTBIT		;WRITE ZERO
6116	035402	104170				ERROR	170		;BIT INCORRECT
6117	035404	005237	003622			INC	BITCNT		;INCREMENT BIT COUNT
6118	035410	005301				DEC	R1		;CHECK IF FINISHED
6119	035412	001356				BNE	18\$;NO, CLOCK NEXT BIT
6120	035414	012762	000240	000026		MOV	#DMD:MIND, RKMR1(R2)		;SIMULATE INDEX
6121	035422	012700	000004			MOV	#4, RO		
6122	035426	012762	000640	000026	20\$:	MOV	#DMD:MIND:MCLK, RKMR1(R2)		
6123	035434	012762	000240	000026		MOV	#DMD:MIND, RKMR1(R2)		
6124	035442	005300				DEC	RO		
6125	035444	001370				BNE	20\$		
6126	035446	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
6127	035454	016237	000026	003464		MOV	RKMR1(R2), T.MR1		;GET MAINT REG 1
6128	035462	012737	022040	003524		MOV	#MEWD:ECCW:DMD, E.MR1		;LOAD EXPECTED MR1
6129	035470	023737	003524	003464		MOV	E.MR1, T.MR1		;CHECK MR1 CORRECT (WRITE GATE RESET)
6130	035476	001401				BEQ	25\$;YES, CHECK IF READY SET
6131	035500	104173				ERROR	173		;MAINT REG 1 INCORRECT
6132	035502	012700	000010		25\$:	MOV	#8, RO		;FINISH COMMAND
6133	035506	012762	000440	000026	26\$:	MOV	#DMD:MCLK, RKMR1(R2)		
6134	035514	012762	000040	000026		MOV	#DMD, RKMR1(R2)		
6135	035522	005300				DEC	RO		
6136	035524	001370				BNE	26\$		
6137	035526	016237	000000	003440		MOV	RKCS1(R2), T.CS1		;GET COMMAND AND STATUS REG 1
6138	035534	012737	000226	003500		MOV	#RDY:WRHEAD<↑C<GO>>, E.CS1		;LOAD EXPECTED CS1
6139	035542	023737	003500	003440		MOV	E.CS1, T.CS1		;CHECK IF CS1 CORRECT
6140	035550	001401				BEQ	TST43		;YES, GO ON TO NEXT TEST
6141	035552	104174				ERROR	174		;CS1 INCORRECT

```

*****
:TEST 43      WRITE LOOPBACK (PART 6)
:
:
:   CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
:   IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
:   IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
:   CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE
:   INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
:   CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:
:
:           155555
:           066666
:           155555
:
:   MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
:   CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
:
*****
TST43: SCOPE

```

6161 035554 000004

6162	035556	012737	000144	001200		MOV	#100, \$TIMES	:: DO 100. ITERATIONS
6163	035554	013702	001270			MOV	\$BASE, R2	:: LOAD RK611 BASE
6164	035570	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:: CLEAR RK611
6165	035576	012762	000040	000026		MOV	#DMD, RKMR1(R2)	:: PUT RK611 IN DIAGNOSTIC MODE
6166	035604	012762	067026	000004		MOV	#HEAD6, RKBA(R2)	:: ISSUE WRITE HEADER
6167	035612	012762	177775	000002		MOV	#-3, RKWC(R2)	
6168	035620	012762	000027	000000		MOV	#WRHEAD, RKCS1(R2)	
6169	035626	012700	000312			MOV	#50, *4+2, R0	:: ISSUE ENOUGH CLOCKS UNTIL :: READY FOR INDEX PULSE
6170								
6171	035632	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)	
6172	035640	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6173	035646	005300				DEC	R0	
6174	035650	001370				BNE	1\$	
6175	035652	012700	000004			MOV	#4, R0	:: ISSUE INDEX PULSE
6176	035656	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6177	035664	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
6178	035672	012762	000240	000026		MOV	#DMD!MIND, RKMR1(R2)	
6179	035700	005300				DEC	R0	
6180	035702	001370				BNE	2\$	
6181	035704	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6182	035712	012700	000010			MOV	#8, R0	:: WAIT FOR WRITE GATE
6183	035716	012762	000440	000026	3\$:	MOV	#DMD!MCLK, RKMR1(R2)	
6184	035724	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6185	035732	005300				DEC	R0	
6186	035734	001370				BNE	3\$	
6187	035736	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	:: SIMULATE SECTOR PULSE
6188	035744	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
6189	035752	005037	003626			CLR	SECCNT	:: INITIALIZE SECTOR COUNT
6190	035756	012737	063045	003170		MOV	#EM233, EMW	:: LOAD ERROR MESSAGE
6191	035764	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT.E.MR1	:: INITIALIZE EXPECTED :: MAINT REG 1
6192								
6193	035772	005037	003612			CLR	P1.BIT	:: INITIALIZE BIT GENERATION
6194	035776	005037	003614			CLR	PR.BIT	
6195	036002	005037	003616			CLR	M1.BIT	
6196	036006	005037	003620			CLR	M2.BIT	
6197	036012	012700	000400			MOV	#256, R0	:: SIMULATE SYNCH
6198	036016	005037	003622			CLR	BITCNT	:: INITIALIZE BIT COUNT
6199	036022	004737	045214		5\$:	JSR	PC, WRTBIT	:: WRITE ONE BIT
6200	036026	104170				ERROR	170	:: DATA INCORRECT
6201	036030	005237	003622			INC	BITCNT	
6202	036034	005300				DEC	R0	:: CHECK IF READY FOR DATA
6203	036036	001371				BNE	5\$:: NO, GENERATE NEXT BIT
6204	036040	012737	000001	003612		MOV	#1, P1.BIT	:: PUT IN SYNCH BIT
6205	036046	004737	045214			JSR	PC, WRTBIT	
6206	036052	104170				ERROR	170	:: DATA INCORRECT
6207	036054	005037	003622			CLR	BITCNT	:: INITIALIZE BIT COUNT
6208	036060	012737	063111	003170		MOV	#EM234, EMW	:: LOAD ERROR MESSAGE
6209	036066	012703	067026			MOV	#HEAD6, R3	:: LOAD ADDRESS OF DATA
6210	036072	012700	000003			MOV	#3, R0	:: LOAD NUMBER WORDS IN HEADER
6211	036076	012304			10\$:	MOV	(R3)+, R4	:: GET NEXT WORD
6212	036100	012701	000020			MOV	#16, R1	:: LOAD BIT COUNT
6213	036104	013737	003616	003620	12\$:	MOV	M1.BIT, M2.BIT	:: SHIFT BITS
6214	036112	013737	003614	003616		MOV	PR.BIT, M1.BIT	
6215	036120	013737	003612	003614		MOV	P1.BIT, PR.BIT	
6216	036126	006004				ROR	R4	:: SHIFT IN NEXT BIT
6217	036130	103403				BCS	14\$:: CHECK IF ONE

```

6218 036132 005037 003612 CLR P1.BIT ;ZERO
6219 036136 000403 BR 15$ ;CLOCK IN BIT
6220
6221 036140 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
6222 036146 004737 045214 15$: JSR PC,WRTBIT ;WRITE BIT
6223 036152 104170 ERROR 170 ;BIT INCORRECT
6224 036154 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6225 036160 005301 DEC R1 ;CHECK IF WORD FINISHED
6226 036162 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
6227 036164 005300 DEC R0 ;CHECK IF HEADER COMPLETE
6228 036166 001343 BNE 10$ ;NO, GET NEXT WORD
6229 036170 012701 000020 MOV #16,R1 ;LOAD BIT COUNT FOR NEXT WORD
6230 036174 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6231 036202 013737 003614 003616 MOV PR.BIT,M1.BIT
6232 036210 013737 003612 003614 MOV P1.BIT,PR.BIT
6233 036216 005037 003612 CLR P1.BIT
6234 036222 004737 045214 JSR PC,WRTBIT ;WRITE ZERO
6235 036226 104170 ERROR 170 ;BIT INCORRECT
6236 036230 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6237 036234 005301 DEC R1 ;CHECK IF FINISHED
6238 036236 001356 BNE 18$ ;NO, CLOCK NEXT BIT
6239 036240 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
6240 036246 012700 000004 MOV #4,R0
6241 036252 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6242 036250 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6243 036266 005300 DEC R0
6244 036270 001370 BNE 20$
6245 036272 012762 000040 000026 MOV #DMD,RKMR1(R2)
6246 036300 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6247 036306 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6248 036314 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
6249 036322 001401 BEQ 25$ ;YES, CHECK IF READY SET
6250 036324 104173 ERROR 173 ;MAINT REG 1 INCORRECT
6251 036326 012700 000010 25$: MOV #8,R0 ;FINISH COMMAND
6252 036332 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6253 036340 012762 000040 000026 MOV #DMD,RKMR1(R2)
6254 036346 005300 DEC R0
6255 036350 001370 BNE 26$
6256 036352 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6257 036360 012737 000226 003500 MOV #RDY!WRHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6258 036366 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6259 036374 001401 BEQ TST44 ;YES, GO ON TO NEXT TEST
6260 036376 104174 ERROR 174 ;CS1 INCORRECT

```

```

*****
*TEST 44 WRITE LOOPBACK (PART 7)
*****
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
* CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:
*
* 104210
* 104210
*

```

```

6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273

```

					104210	
6274				:	*	
6275				:	*	
6276				:	*	
6277				:	*	
6278				:	*	
6279				:	*	
6280	036400	000004		:	*****	
6281	036402	012737	000144	↑ST44:	SCOPE	
6282	036410	013702	001270		MOV	#100., \$TIMES ; DO 100. ITERATIONS
6283	036414	012762	100000		MOV	\$BASE, R2 ; LOAD RK611 BASE
6284	036422	012762	000040		MOV	#CCLR, RKCS1(R2) ; CLEAR RK611
6285	036430	012762	067034		MOV	#DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
6286	036436	012762	177775		MOV	#HEAD7, RKBA(R2) ; ISSUE WRITE HEADER
6287	036444	012762	000027		MOV	#-3, RKWC(R2)
6288	036452	012700	000312		MOV	#WRHEAD, RKCS1(R2)
6289					MOV	#50. *4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY FOR INDEX PULSE
6290	036456	012762	000440	1\$:	MOV	#DMD!MCLK, RKMR1(R2)
6291	036464	012762	000040		MOV	#DMD, RKMR1(R2)
6292	036472	005300			DEC	R0
6293	036474	001370			BNE	1\$
6294	036476	012700	000004		MOV	#4, R0 ; ISSUE INDEX PULSE
6295	036502	012762	000240		MOV	#DMD!MIND, RKMR1(R2)
6296	036510	012762	000640	2\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)
6297	036516	012762	000240		MOV	#DMD!MIND, RKMR1(R2)
6298	036524	005300			DEC	R0
6299	036526	001370			BNE	2\$
6300	036530	012762	000040		MOV	#DMD, RKMR1(R2)
6301	036536	012700	000010		MOV	#8, R0 ; WAIT FOR WRITE GATE
6302	036542	012762	000440	3\$:	MOV	#DMD!MCLK, RKMR1(R2)
6303	036550	012762	000040		MOV	#DMD, RKMR1(R2)
6304	036556	005300			DEC	R0
6305	036560	001370			BNE	3\$
6306	036562	012762	000140		MOV	#DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
6307	036570	012762	000040		MOV	#DMD, RKMR1(R2)
6308	036576	005037	003626		CLR	SECCNT ; INITIALIZE SECTOR COUNT
6309	036602	012737	063045		MOV	#EM233, EMW ; LOAD ERROR MESSAGE
6310	036610	012737	062040		MOV	#DMD!MEWD!ECCW!WRTGAT, E.MR1 ; INITIALIZE EXPECTED MAINT REG 1
6311						;
6312	036616	005037	003612		CLR	P1.BIT ; INITIALIZE BIT GENERATION
6313	036622	005037	003614		CLR	PR.BIT
6314	036626	005037	003616		CLR	M1.BIT
6315	036632	005037	003620		CLR	M2.BIT
6316	036636	012700	000400		MOV	#256, R0 ; SIMULATE SYNCH
6317	036642	005037	003622		CLR	BITCNT ; INITIALIZE BIT COUNT
6318	036646	004737	045214		JSR	PC, WRTBIT ; WRITE ONE BIT
6319	036652	104170		5\$:	ERROR	170 ; DATA INCORRECT
6320	036654	005237	003622		INC	BITCNT
6321	036660	005300			DEC	R0 ; CHECK IF READY FOR DATA
6322	036662	001371			BNE	5\$; NO, GENERATE NEXT BIT
6323	036664	012737	000001	003612	MOV	#1, P1.BIT ; PUT IN SYNCH BIT
6324	036672	004737	045214		JSR	PC, WRTBIT
6325	036676	104170			ERROR	170 ; DATA INCORRECT
6326	036700	005037	003622		CLR	BITCNT ; INITIALIZE BIT COUNT
6327	036704	012737	063111	003170	MOV	#EM234, EMW ; LOAD ERROR MESSAGE
6328	036712	012703	067034		MOV	#HEAD7, R3 ; LOAD ADDRESS OF DATA
6329	036716	012700	000003		MOV	#3, R0 ; LOAD NUMBER WORDS IN HEADER

```

6330 036722 012304      10$:  MOV      (R3)+,R4      ;GET NEXT WORD
6331 036724 012701 000020  MOV      #16.,R1      ;LOAD BIT COUNT
6332 036730 013737 003616 003620 12$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
6333 036736 013737 003614 003616  MOV      PR.BIT,M1.BIT
6334 036744 013737 003612 003614  MOV      P1.BIT,PR.BIT
6335 036752 006004      ROR      R4            ;SHIFT IN NEXT BIT
6336 036754 103403      BCS      14$          ;CHECK IF ONE
6337 036756 005037 003612  CLR      P1.BIT      ;ZERO
6338 036762 000403      BR       15$          ;CLOCK IN BIT
6339
6340 036764 012737 000001 003612 14$:  MOV      #1,P1.BIT    ;ONE
6341 036772 004737 045214 15$:  JSR      PC,WRTBIT   ;WRITE BIT
6342 036776 104170      ERROR    170         ;BIT INCORRECT
6343 037000 005237 003622  INC      BITCNT      ;INCREMENT BIT COUNT
6344 037004 005301      DEC      R1          ;CHECK IF WORD FINISHED
6345 037006 001350      BNE     12$         ;NO, CONTINUE WITH NEXT BIT
6346 037010 005300      DEC      R0          ;CHECK IF HEADER COMPLETE
6347 037012 001343      BNE     10$         ;NO, GET NEXT WORD
6348 037014 012701 000020  MOV      #16.,R1      ;LOAD BIT COUNT FOR NEXT WORD
6349 037020 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
6350 037026 013737 003614 003616  MOV      PR.BIT,M1.BIT
6351 037034 013737 003612 003614  MOV      P1.BIT,PR.BIT
6352 037042 005037 003612  CLR      P1.BIT
6353 037046 004737 045214  JSR      PC,WRTBIT   ;WRITE ZERO
6354 037052 104170      ERROR    170         ;BIT INCORRECT
6355 037054 005237 003622  INC      BITCNT      ;INCREMENT BIT COUNT
6356 037060 005301      DEC      R1          ;CHECK IF FINISHED
6357 037062 001356      BNE     18$         ;NO, CLOCK NEXT BIT
6358 037064 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
6359 037072 012700 000004  MOV      #4,R0
6360 037076 012762 000640 000026 20$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
6361 037104 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
6362 037112 005300      DEC      R0
6363 037114 001370      BNE     20$
6364 037116 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6365 037124 016237 000026 003464  MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
6366 037132 012737 022040 003524  MOV      #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6367 037140 023737 003524 003464  CMP      E.MR1,T.MR1  ;CHECK MR1 CORRECT (WRITE GATE RESET)
6368 037146 001401      BEQ     25$         ;YES, CHECK IF READY SET
6369 037150 104173      ERROR    173         ;MAINT REG 1 INCORRECT
6370 037152 012700 000010  MOV      #8.,R0      ;FINISH COMMAND
6371 037156 012762 000440 000026 25$:  MOV      #DMD!MCLK,RKMR1(R2)
6372 037164 012762 000040 000026 26$:  MOV      #DMD,RKMR1(R2)
6373 037172 005300      DEC      R0
6374 037174 001370      BNE     26$
6375 037176 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6376 037204 012737 000226 003500  MOV      #RDY!WRHEAD&<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6377 037212 023737 003500 003440  CMP      E.CS1,T.CS1  ;CHECK IF CS1 CORRECT
6378 037220 001401      BEQ     TST45       ;YES, GO ON TO NEXT TEST
6379 037222 104174      ERROR    174         ;CS1 INCORRECT
6380
6381
6382
6383
6384
6385

```

```

*****
*TEST 45      WRITE TWO HEADERS
*
*      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
*      IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06

```

```

6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406 037224 000004
6407 037226 012737 000144 001200
6408 037234 013702 001270
6409 037240 012762 100000 000000
6410 037246 012762 000040 000026
6411 037254 012762 066762 000004
6412 037262 012703 066762
6413 037266 012762 177772 000002
6414 037274 012762 000027 000000
6415 037302 012700 000312
6416
6417 037306 012762 000440 000026 1$:
6418 037314 012762 000040 000026
6419 037322 005300
6420 037324 001370
6421 037326 012700 000004
6422 037332 012762 000240 000026
6423 037340 012762 000640 000026 2$:
6424 037346 012762 000240 000026
6425 037354 005300
6426 037356 001370
6427 037360 005037 003626
6428 037364 012762 000040 000026
6429 037372 012705 000002
6430 037376 012700 000010
6431 037402 012762 000440 000026 3$:
6432 037410 012762 000040 000026
6433 037416 005300
6434 037420 001370
6435 037422 012762 000140 000026 4$:
6436 037430 012762 000040 000026
6437 037436 012737 063045 003170
6438 037444 012737 062040 003524
6439
6440 037452 005037 003612
6441 037456 005037 003614

```

```

: * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
: * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES, SIMULATE
: * INDEX PULSE, SECTOR PULSE, THREE WORD HEADER
: * CONSISTING OF THE FOLLOWING DATA:
: *
: * 177777
: * 000000
: * 177777
: *
: * FOLLOW THAT BY A SECTOR PULSE AND ONE THREE WORD
: * HEADER CONSISTING OF THE FOLLOWING DATA:
: *
: * 000000
: * 177777
: * 000000
: *
: * SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP.
: * CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMENSATION.
: *
: * *****
: * ST45: SCOPE
: * MOV #100, $TIMES ; DO 100. ITERATIONS
: * MOV $BASE, R2 ; LOAD RK611 BASE
: * MOV #CLR, RKCS1(R2) ; CLEAR RK611
: * MOV #DMD, RKMR1(R2) ; PUT RK611 TO DIAGNOSTIC MODE
: * MOV #HEAD1, RKBA(R2) ; ISSUE WRITE HEADER
: * MOV #HEAD1, R3
: * MOV #-6, R4(R2)
: * MOV #WRHEAD, RKCS1(R2)
: * MOV #50, *4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL
: * ; READY FOR INDEX PULSE
: * 1$: MOV #DMD!MCLK, RKMR1(R2)
: * MOV #DMD, RKMR1(R2)
: * DEC R0
: * BNE 1$
: * MOV #4, R0 ; ISSUE INDEX PULSE
: * MOV #DMD!MIND, RKMR1(R2)
: * 2$: MOV #DMD!MIND!MCLK, RKMR1(R2)
: * MOV #DMD!MIND, RKMR1(R2)
: * DEC R0
: * BNE 2$
: * CLR SECCNT ; CLEAR SECTOR COUNT
: * MOV #DMD, RKMR1(R2)
: * MOV #2, R5 ; LOAD NUMBER OF HEADERS
: * MOV #8, R0 ; WAIT FOR WRITE GATE
: * 3$: MOV #DMD!MCLK, RKMR1(R2)
: * MOV #DMD, RKMR1(R2)
: * DEC R0
: * BNE 3$
: * MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
: * MOV #DMD, RKMR1(R2)
: * MOV #EM233, EMW ; LOAD ERROR MESSAGE
: * MOV #DMD!MEWD!ECCW!WRTGAT, E.MR1 ; INITIALIZE EXPECTED
: * ; MAINT REG 1
: * CLR P1.BIT
: * CLR PR.BIT

```

6442	037462	005037	003616		CLR	M1.BIT		
6443	037466	005037	003620		CLR	M2.BIT		
6444	037472	012700	000400		MOV	#256, R0	; SIMULATE SYNCH	
6445	037476	005037	003622		CLR	BITCNT	; INITIALIZE BIT COUNT	
6446	037502	004737	045214	5\$:	JSR	PC, WRTBIT	; WRITE ONE BIT	
6447	037506	104170			ERROR	170	; DATA INCORRECT	
6448	037510	005237	003622		INC	BITCNT		
6449	037514	005300			DEC	R0	; CHECK IF READY FOR DATA	
6450	037516	001371			BNE	5\$; NO GENERATE NEXT BIT	
6451	037520	012737	000001	003612	MOV	#1, P1.BIT	; PUT IN SYNCH BIT	
6452	037526	004737	045214		JSR	PC, WRTBIT		
6453	037532	104170			ERROR	170	; DATA INCORRECT	
6454	037534	005037	003622		CLR	BITCNT	; INITIALIZE BIT COUNT	
6455	037540	012737	063111	003170	MOV	#EM234, EMW	; LOAD ERROR MESSAGE	
6456	037546	012700	000003		MOV	#3, R0	; LOAD NUMBER OF WORDS IN HEADER	
6457	037552	012304		10\$:	MOV	(R3)+, R4	; GET NEXT WORD	
6458	037554	012701	000020		MOV	#16, R1	; LOAD BIT COUNT	
6459	037560	013737	003616	003620	12\$:	MOV	M1.BIT, M2.BIT	; SHIFT BITS
6460	037566	013737	003614	003616	MOV	PR.BIT, M1.BIT		
6461	037574	013737	003612	003614	MOV	P1.BIT, PR.BIT		
6462	037602	006004			ROR	R4	; SHIFT IN NEXT BIT	
6463	037604	103403			BCS	14\$; CHECK IF ONE	
6464	037606	005037	003612		CLR	P1.BIT	; ZERO	
6465	037612	000403			BR	15\$; CLOCK IN BIT	
6466								
6467	037614	012737	000001	003612	14\$:	MOV	#1, P1.BIT	; ONE
6468	037622	004737	045214	15\$:	JSR	PC, WRTBIT	; WRITE BIT	
6469	037626	104170			ERROR	170	; BIT INCORRECT	
6470	037630	005237	003622		INC	BITCNT	; INCREMENT BIT COUNT	
6471	037634	005301			DEC	R1	; CHECK IF WORD FINISHED	
6472	037636	001350			BNE	12\$; NO CONTINUE	
6473	037640	005300			DEC	R0	; CHECK IF HEADER COMPLETE	
6474	037642	001343			BNE	10\$; NO, GET NEXT WORD	
6475	037644	012701	000020		MOV	#16, R1	; LOAD BIT COUNT FOR NEXT WORD	
6476	037650	013737	003616	003620	18\$:	MOV	M1.BIT, M2.BIT	; SHIFT BITS
6477	037656	013737	003614	003616	MOV	PR.BIT, M1.BIT		
6478	037664	013737	003612	003614	MOV	P1.BIT, PR.BIT		
6479	037672	005037	003612		CLR	P1.BIT		
6480	037676	004737	045214		JSR	PC, WRTBIT	; WRITE ZERO	
6481	037702	104170			ERROR	170	; BIT INCORRECT	
6482	037704	005237	003622		INC	BITCNT	; INCREMENT	
6483	037710	005301			DEC	R1	; CHECK IF READY FOR NEXT HEADER	
6484	037712	001356			BNE	18\$; NO CONTINUE	
6485	037714	005237	003626		INC	SECCNT	; INCREMENT	
6486	037720	005305			DEC	R5	; CHECK IF SECOND HEADER WRITTEN	
6487	037722	001237			BNE	4\$; NO, DO SECOND HEADER	
6488	037724	012762	000240	000026	MOV	#DMD!MIND, RKMR1(R2)	; SIMULATE INDEX PULSE	
6489	037732	012700	000004		MOV	#4, R0		
6490	037736	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK, RKMR1(R2)	
6491	037744	012762	000240	000026	MOV	#DMD!MIND, RKMR1(R2)		
6492	037752	005300			DEC	R0		
6493	037754	001370			BNE	20\$		
6494	037756	012762	000040	000026	MOV	#DMD, RKMR1(R2)		
6495	037764	016237	000026	003464	MOV	RKMR1(R2), T.MR1	; GET MAINT REG 1	
6496	037772	012737	022040	003524	MOV	#MEWD!ECCW!DMD, E.MR1	; LOAD EXPECTED MR1	
6497	040000	023737	003524	003464	CMP	E.MR1, T.MR1	; CHECK MR1 CORRECT (WRITE GATE RESET)	

```

6498 040006 001401
6499 040010 104173
6500 040012 012700 000010
6501 040016 012762 000440 000026
6502 040024 012762 000040 000026
6503 040032 005300
6504 040034 001370
6505 040036 016237 000000 003440
6506 040044 012737 000226 003500
6507 040052 023737 003500 003440
6508 040060 001401
6509 040062 104174

```

```

BEQ 25$ ;YES, CHECK IF READY SET
ERROR 173 ;MAINT REG 1 INCORRECT
MOV #8.,RO ;FINISH COMMAND
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC RO
BNE 26$
MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
MOV #RDY!WRHEAD&<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
BEQ TST46 ;;YES, GO ON TO NEXT TEST
ERROR 174

```

```

*****
*TEST 46 DATA FIELD FILLING ON WRITE HEADER

```

```

* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND
* SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE
* FOLLOWING DATA:

```

```

125252
052525
125252
052525
125252
052525

```

```

* MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
* ECC FIELD ARE WRITTEN CORRECTLY.

```

```

*****
*ST46: SCOPE

```

```

6531 040064 000004
6532 040066 012737 000144 001200
6533 040074 013702 001270
6534 040100 012762 100000 000000
6535 040106 012762 000040 000026
6536 040114 012762 066776 000004
6537 040122 012703 066776
6538 040126 012762 177772 000002
6539 040134 012762 000027 000000
6540 040142 012700 000312
6541
6542 040146 012762 000440 000026 1$:
6543 040154 012762 000040 000026
6544 040162 005300
6545 040164 001370
6546 040166 012700 000004
6547 040172 012762 000240 000026
6548 040200 012762 000640 000026 2$:
6549 040206 012762 000240 000026
6550 040214 005300
6551 040216 001370
6552 040220 012762 000040 000026
6553 040226 012700 000010

```

```

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

```


G10

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046)
T46

02-DEC-77 09:56 PAGE 123
DATA FIELD FILLING ON WRITE HEADER

SEQ 0123

```

6610 040556 012737 000001 003612 MOV #1,P1.BIT ;SET SYNCH BIT
6611 040564 004737 045214 JSR PC,WRTBIT ;WRITE SYNCH BIT
6612 040570 104170 ERROR 170 ;SYNCH BIT INCORRECT
6613 040572 012737 063411 003170 MOV #EM238,EMW ;LOAD ERROR MESSAGE
6614 040600 005037 003622 CLR BITCNT ;CLEAR BIT COUNT
6615 040604 012701 007777 MOV #256,#16,-1,R1 ;LOAD COUNT
6616 040610 013737 003616 003620 20$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6617 040616 013737 003614 003616 MOV PR.BIT,M1.BIT
6618 040624 013737 003612 003614 MOV P1.BIT,PR.BIT
6619 040632 005037 003612 CLR P1.BIT
6620 040636 004737 045214 JSR PC,WRTBIT ;WRITE ZEROS
6621 040642 104170 ERROR 170 ;BIT INCORRECT
6622 040644 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6623 040650 005301 DEC R1 ;CHECK IF READY FOR ECC
6624 040652 001356 BNE 20$ ;NO, CONTINUE
6625 040654 012701 000040 MOV #32.,R1 ;LOAD COUNT
6626 040660 042737 020000 003524 BIC #ECCW,E.MR1 ;RESET ECCW BIT
6627 040666 004737 045214 21$: JSR PC,WRTBIT ;WRITE ECC FIELD
6628 040672 104170 ERROR 170 ;BIT INCORRECT
6629 040674 005237 003622 INC BITCNT ;INCREMENT COUNT
6630 040700 005301 DEC R1 ;CHECK IF READY FOR POST AMBLE
6631 040702 001371 BNE 21$ ;NO CONTINUE
6632 040704 052737 020000 003524 BIS #ECCW,E.MR1 ;SET ECCW
6633 040712 012701 000012 MOV #10.,R1 ;CHECK IF READY FOR NEXT HEADER
6634 040716 004737 045214 22$: JSR PC,WRTBIT ;WRITE POSTAMBLE
6635 040722 104170 ERROR 170 ;BIT INCORRECT
6636 040724 005237 003622 INC BITCNT ;INCREMENT COUNT
6637 040730 005301 DEC R1 ;CHECK IF READY FOR NEXT HEADER
6638 040732 001371 BNE 22$ ;NO, COMPLETE POSTAMBLE
6639 040734 005237 003626 INC SECCNT ;INCREMENT COUNT
6640 040740 005305 DEC R5 ;CHECK IF ALL HEADERS RECEIVED
6641 040742 001402 BEQ 23$ ;YES, SIMULATE INDEX
6642 040744 000137 040262 JMP 4$ ;NO GET NEXT HEADER
6643
6644 040750 012762 000240 000026 23$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6645 040756 012700 000004 MOV #4,RO
6646 040762 012762 000640 000026 25$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6647 040770 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6648 040776 005300 DEC RO
6649 041000 001370 BNE 25$
6650 041002 012762 000040 000026 MOV #DMD,RKMR1(R2)
6651 041010 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6652 041016 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6653 041024 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK IF MR1 CORRECT
6654 ; (WRITE GATE RESET)
6655 BEQ 28$ ;YES, CHECK IF READY SET
6656 041034 104173 ERROR 173 ;MAINTENANCE REG 1 INCORRECT
6657 041036 012700 000010 28$: MOV #8.,RO ;FINISH COMMAND
6658 041042 012762 000440 000026 29$: MOV #DMD!MCLK,RKMR1(R2)
6659 041050 012762 000040 000026 MOV #DMD,RKMR1(R2)
6660 041056 005300 DEC RO
6661 041060 001370 BNE 29$
6662 041062 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6663 041070 012737 000226 003500 MOV #RDY!WRHEAD&<fC<GO>>,E.CS1 ;LOAD EXPECTED CS1
6664 041076 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6665 041104 001401 BEQ ;:YES, GO TO NEXT TEST

```

H10

CZR6CCO RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 124
T46 DATA FIELD FILLING ON WRITE HEADER

SEQ 0124

6666 041106 104174

ERROR 174 ;CS1 INCORRECT

```
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677 041110 000004
6678 041112 012737 000010 001200
6679 041120 013702 001270
6680 041124 012762 100000 000000
6681 041132 012762 000040 000026
6682 041140 012762 067042 000004
6683 041146 012703 067042
6684 041152 012762 177676 000002
6685 041160 012762 000027 000000
6686 041166 012700 000312
6687
6688 041172 012762 000440 000026 1$:
6689 041200 012762 000040 000026
6690 041206 005300
6691 041210 001370
6692 041212 012700 000004
6693 041216 012762 000240 000026
6694 041224 012762 000640 000026 2$:
6695 041232 012762 000240 000026
6696 041240 005300
6697 041242 001370
6698 041244 012762 000040 000026
6699 041252 012700 000010
6700 041256 012762 000440 000026 3$:
6701 041264 012762 000040 000026
6702 041272 005300
6703 041274 001370
6704 041276 005037 003626
6705 041302 012705 000026
6706 041306 012762 000140 000026 4$:
6707 041314 012762 000040 000026
6708 041322 012737 063045 003170
6709 041330 012737 062040 003524
6710
6711 041336 005037 003612
6712 041342 005037 003614
6713 041346 005037 003616
6714 041352 005037 003620
6715 041356 012700 000400
6716 041362 005037 003622
6717 041366 004737 045214
6718 041372 104170
6719 041374 005237 003622
6720 041400 005300
6721 041402 001371
```

```
*****
*TEST 47 WRITE HEADER FOR 26 SECTORS
*
* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIFYING
* 66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTLY.
*****
↑ST47: SCOPE
MOV #10,STIMES ;DO 10 ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #NPRBUF,RKBA(R2) ;ISSUE READ HEADER
MOV #NPRBUF,R3
MOV #-22,*3,RKWC(R2)
MOV #WRHEAD,RKCS1(R2)
MOV #50,*4+2,R0 ;ISSUE CLOCKS UNTIL READY
; FOR INDEX PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #4,R0 ;ISSUE INDEX PULSE
2$: MOV #DMD!MIND,RKMR1(R2)
MOV #DMD!MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
DEC R0
BNE 2$
MOV #8,R0 ;WAIT FOR WRITE GATE
3$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 3$
CLR SECCNT ;CLEAR SECTOR COUNT
MOV #22,R5 ;LOAD NUMBER OF HEADERS
4$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
MOV #EM233,EMW ;LOAD ERROR MESSAGE
MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
; MAINT REG I
CLR P1.BIT
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256,R0 ;SIMULATE SYNCH
5$: CLR BITCNT ;INITIALIZE BIT COUNT
JSR PC,WRTBIT ;WRITE ONE BIT
ERROR 170 ;DATA INCORRECT
INC BITCNT
DEC R0 ;CHECK IF READY FOR DATA
BNE 5$ ;NO, GENERATE NEXT BIT
```

6722	041404	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
6723	041412	004737	045214			JSR	PC,WRTBIT	
6724	041416	104170				ERROR	170	;DATA INCORRECT
6725	041420	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6726	041424	012737	063111	003170		MOV	#EM234,EMW	;LOAD ERROR MESSAGE
6727	041432	012700	000003			MOV	#3,R0	;LOAD NUMBER OF WORDS IN HEADER
6728	041436	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
6729	041440	012701	000020			MOV	#16.,R1	;LOAD BIT COUNT
6730	041444	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6731	041452	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6732	041460	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6733	041466	006004				ROR	R4	;SHIFT IN NEXT BIT
6734	041470	103403				BCS	14\$;CHECK IF ONE
6735	041472	005037	003612			CLR	P1.BIT	;ZERO
6736	041476	000403				BR	15\$;CLOCK IN BIT
6737								
6738	041500	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
6739	041506	004737	045214		15\$:	JSR	PC,WRTBIT	;WRITE BIT
6740	041512	104170				ERROR	170	;BIT INCORRECT
6741	041514	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
6742	041520	005301				DEC	R1	;CHECK IF WORD FINISHED
6743	041522	001350				BNE	12\$;NO, CONTINUE
6744	041524	005300				DEC	R0	;CHECK IF HEADER COMPLETE
6745	041526	001343				BNE	10\$;NO, GET NEXT WORD
6746	041530	012737	063343	003170		MOV	#EM237,EMW	;LOAD ERROR MESSAGE
6747	041536	012701	000477			MOV	#64,+255.,R1	;LOAD COUNT
6748	041542	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6749	041550	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6750	041556	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6751	041564	005037	003612			CLR	P1.BIT	
6752	041570	004737	045214			JSR	PC,WRTBIT	;WRITE ZERO
6753	041574	104170				ERROR	170	;BIT INCORRECT
6754	041576	005301				DEC	R1	;CHECK IF READY FOR DATA
6755	041600	001360				BNE	18\$;NO, CONTINUE
6756	041602	012737	000001	003612		MOV	#1,P1.BIT	;SET SYNCH BIT
6757	041610	004737	045214			JSR	PC,WRTBIT	;WRITE SYNCH BIT
6758	041614	104170				ERROR	170	;SYNCH BIT INCORRECT
6759	041616	012737	063411	003170		MOV	#EM238,EMW	;LOAD ERROR MESSAGE
6760	041624	005037	003622			CLR	BITCNT	;CLEAR BIT COUNT
6761	041630	012701	007777			MOV	#256.*16.-1,R1	;LOAD COUNT
6762	041634	013737	003616	003620	20\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6763	041642	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6764	041650	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6765	041656	005037	003612			CLR	P1.BIT	
6766	041662	004737	045214			JSR	PC,WRTBIT	;WRITE ZEROS
6767	041666	104170				ERROR	170	;BIT INCORRECT
6768	041670	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
6769	041674	005301				DEC	R1	;CHECK IF READY FOR ECC
6770	041676	001356				BNE	20\$;NO, CONTINUE
6771	041700	012701	000040			MOV	#32.,R1	;LOAD COUNT
6772	041704	042737	020000	003524		BIC	#ECCW,E.MR1	;RESET ECCW BIT
6773	041712	004737	045214		21\$:	JSR	PC,WRTBIT	;WRITE ECC FIELD
6774	041716	104170				ERROR	170	;BIT INCORRECT
6775	041720	005237	003622			INC	BITCNT	;INCREMENT COUNT
6776	041724	005301				DEC	R1	;CHECK IF READY FOR POST AMBLE
6777	041726	001371				BNE	21\$;NO CONTINUE

6778	041730	052737	020000	003524		BIS	#ECCW,E.MR1	;;SET ECCW
6779	041736	012701	000012			MOV	#10,R1	;;CHECK IF READY FOR NEXT HEADER
6780	041742	004737	045214		22\$:	JSR	PC,WRTBIT	;;WRITE POSTAMBLE
6781	041746	104170				ERROR	170	;;BIT INCORRECT
6782	041750	005237	003622			INC	BITCNT	;;INCREMENT COUNT
6783	041754	005301				DEC	R1	;;CHECK IF READY FOR NEXT HEADER
6784	041756	001371				BNE	22\$;;NO, COMPLETE POSTAMBLE
6785	041760	005237	003626			INC	SECCNT	;;INCREMENT COUNT
6786	041764	005305				DEC	R5	;;CHECK IF ALL HEADERS RECEIVED
6787	041766	001402				BEQ	23\$;;YES, SIMULATE INDEX
6788	041770	000137	041306			JMP	4\$;;NO GET NEXT HEADER
6789								
6790	041774	012762	000240	000026	23\$:	MOV	#DMD!MIND,RKMR1(R2)	;;SIMULATE INDEX PULSE
6791	042002	012700	000004			MOV	#4,RO	
6792	042006	012762	000640	000026	25\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6793	042014	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6794	042022	005300				DEC	RO	
6795	042024	001370				BNE	25\$	
6796	042026	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6797	042034	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;;GET MAINT REG 1
6798	042042	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	;;LOAD EXPECTED MR1
6799	042050	023737	003524	003464		CMF	E.MR1,T.MR1	;;CHECK IF MR1 CORRECT
6800								;;(WRITE GATE RESET)
6801	042056	001401				BEQ	28\$;;YES, CHECK IF READY SET
6802	042060	104173				ERROR	173	;;MAINTENANCE REG 1 INCORRECT
6803	042062	012700	000010		28\$:	MOV	#8,RO	;;FINISH COMMAND
6804	042066	012762	000440	000026	29\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6805	042074	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6806	042102	005300				DEC	RO	
6807	042104	001370				BNE	29\$	
6808	042106	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;;GET COMMAND AND STATUS REG 1
6809	042114	012737	000226	003500		MOV	#RDY!WRHEAD&<IC<GO>>,E.CS1	;;LOAD EXPECTED CS1
6810	042122	023737	003500	003440		CMF	E.CS1,T.CS1	;;CHECK IF CS1 CORRECT
6811	042130	001401				BEQ	TST50	;;YES, GO TO NEXT TEST
6812	042132	104174				ERROR	174	;;CS1 INCORRECT

 ;TEST 50 WRITE HEADER IN 24 SECTOR FORMAT

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

6826								
6827	042134	000004			TST50:	SCOPE		*****
6828	042136	012737	000144	001200		MOV	#100,\$TIMES	;;DO 100 ITERATIONS
6829	042144	013702	001270			MOV	\$BASE,R2	;;LOAD RK611 BASE
6830	042150	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;;CLEAR RK611
6831	042156	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;;PUT RK611 TO DIAGNOSTIC MODE
6832	042164	012762	066762	000004		MOV	#HEAD1,RKBA(R2)	;;ISSUE WRITE HEADER
6833	042172	012703	066762			MOV	#HEAD1,R3	

6834	042176	012762	177772	000002		MOV	#-6,RKWC(R2)	
6835	042204	012762	010027	000000		MOV	#CFMT!WRHEAD,RKCS1(R2)	
6836	042212	012700	000312			MOV	#50.*4+2,RO	;ISSUE ENOUGH CLOCKS UNTIL ;READY FOR INDEX PULSE
6837								
6838	042216	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6839	042224	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6840	042232	005300				RO		
6841	042234	001370				BNE	1\$	
6842	042236	012700	000004			MOV	#4,RO	;ISSUE INDEX PULSE
6843	042242	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6844	042250	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6845	042256	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6846	042264	005300				RO		
6847	042266	001370				BNE	2\$	
6848	042270	005037	003626			CLR	SECCNT	;CLEAR SECTOR COUNT
6849	042274	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6850	042302	012705	000002			MOV	#2,R5	;LOAD NUMBER OF HEADERS
6851	042306	012700	000010			MOV	#8,RO	;WAIT FOR WRITE GATE
6852	042312	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6853	042320	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6854	042326	005300				RO		
6855	042330	001370				BNE	3\$	
6856	042332	012762	000140	000026	4\$:	MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
6857	042340	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6858	042346	012737	063450	003170		MOV	#EM239,EMW	;LOAD ERROR MESSAGE
6859	042354	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1	;INITIALIZE EXPECTED ;MAINT REG 1
6860								
6861	042362	005037	003612			CLR	P1.BIT	
6862	042366	005037	003614			CLR	PR.BIT	
6863	042372	005037	003616			CLR	M1.BIT	
6864	042376	005037	003620			CLR	M2.BIT	
6865	042402	012700	000400			MOV	#256,RO	;SIMULATE SYNCH
6866	042406	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6867	042412	004737	045214		5\$:	JSR	PC,WRTBIT	;WRITE ONE BIT
6868	042416	104170				ERROR	170	;DATA INCORRECT
6869	042420	005237	003622			INC	BITCNT	
6870	042424	005300				RO		;CHECK IF READY FOR DATA
6871	042426	001371				BNE	5\$;NO GENERATE NEXT BIT
6872	042430	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
6873	042436	004737	045214			JSR	PC,WRTBIT	
6874	042442	104170				ERROR	170	;DATA INCORRECT
6875	042444	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6876	042450	012737	063542	003170		MOV	#EM240,EMW	;LOAD ERROR MESSAGE
6877	042456	012700	000003			MOV	#3,RO	;LOAD NUMBER OF WORDS IN HEADER
6878	042462	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
6879	042464	012701	000020			MOV	#16,R1	;LOAD BIT COUNT
6880	042470	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6881	042476	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6882	042504	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6883	042512	006004				ROR	R4	;SHIFT IN NEXT BIT
6884	042514	103403				BCS	14\$;CHECK IF ONE
6885	042516	005037	003612			CLR	P1.BIT	;ZERO
6886	042522	000403				BR	15\$;CLOCK IN BIT
6887								
6888	042524	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
6889	042532	004737	045214		15\$:	JSR	PC,WRTBIT	;WRITE BIT

```

6890 042536 104170          ERROR 170          ;BIT INCORRECT
6891 042540 005237 003622  INC BITCNT        ;INCREMENT BIT COUNT
6892 042544 005301          DEC R1             ;CHECK IF WORD FINISHED
6893 042546 001350          BNE 12$           ;NO, CONTINUE
6894 042550 005300          DEC R0             ;CHECK IF HEADER COMPLETE
6895 042552 001343          BNE 10$           ;NO, GET NEXT WORD
6896 042554 012701 000020  MOV #16,R1        ;LOAD BIT COUNT FOR NEXT WORD
6897 042560 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6898 042566 013737 003614 003616  MOV PR.BIT,M1.BIT
6899 042574 013737 003612 003614  MOV P1.BIT,PR.BIT
6900 042602 005037 003612  CLR P1.BIT
6901 042606 004737 045214  JSR PC,WRTBIT    ;WRITE ZERO
6902 042612 104170          ERROR 170          ;BIT INCORRECT
6903 042614 005237 003622  INC BITCNT        ;INCREMENT
6904 042620 005301          DEC R1             ;CHECK IF READY FOR NEXT HEADER
6905 042622 001356          BNE 18$           ;NO, CONTINUE
6906 042624 005237 003626  INC SECCNT        ;INCREMENT
6907 042630 005305          DEC R5             ;CHECK IF SECOND HEADER WRITTEN
6908 042632 001237          BNE 4$            ;NO, DO SECOND HEADER
6909 042634 012762 000240 000026  MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6910 042642 012700 000004  MOV #4,R0
6911 042646 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6912 042654 012762 000240 000026  MOV #DMD!MIND,RKMR1(R2)
6913 042662 005300          DEC R0
6914 042664 001370          BNE 20$
6915 042666 012762 000040 000026  MOV #DMD,RKMR1(R2)
6916 042674 016237 000026 003464  MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6917 042702 012737 022040 003524  MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6918 042710 023737 003524 003464  CMP E.MR1,T.MR1   ;CHECK MR1 CORRECT (WRITE GATE RESET)
6919 042716 001401          BEQ 25$           ;YES, CHECK IF READY SET
6920 042720 104173          ERROR 173         ;MAINT REG 1 INCORRECT
6921 042722 012700 000010 25$: MOV #8,R0         ;FINISH COMMAND
6922 042726 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6923 042734 012762 000040 000026  MOV #DMD,RKMR1(R2)
6924 042742 005300          DEC R0
6925 042744 001370          BNE 26$
6926 042746 016237 000000 003440  MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6927 042754 012737 010226 003500  MOV #RDY!CFMT!WRHEAD<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6928 042762 023737 003500 003440  CMP E.CS1,T.CS1   ;CHECK IF CS1 CORRECT
6929 042770 001401          BEQ TST51        ;;YES, GO ON TO NEXT TEST
6930 042772 104174          ERROR 174

```

.SBTTL **TYPE B INSTRUCTION ERRORS

```

*****
;TEST 51          FORMAT ERROR (PART 1)
;
; CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
; CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN
; RK06, IN 26 SECTOR FORMAT, CYLINDER 43, HEAD 0, DRIVE 0.
; CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF
; MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
; DRIVE AVAILABLE AND CONTROLLER ERROR SET.
*****
TST51: SCOPE

```

6945 042774 000004

6946	042776	012737	000144	001200	MOV	#100, \$TIMES	:: DO 100 ITERATIONS
6947	043008	013702	001270		MOV	\$BASE, R2	:: LOAD RK611 BASE
6948	043010	012762	000040	000010	MOV	#SCLR, RKCS2(R2)	:: CLEAR RK611 SUBSYSTEM
6949	043016	012762	000040	000026	MOV	#DMD, RKMR1(R2)	:: PUT RK611 IN MAINTENANCE MODE
6950	043024	012762	000043	000020	MOV	#43, RKDCYL(R2)	:: LOAD CYLINDER ADDRESS REG
6951	043032	012762	000025	000000	MOV	#RDHEAD, RKCS1(R2)	:: ISSUE READ HEADER
6952	043040	012700	000132		MOV	#22, #4+2, R0	:: ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6953	043044	012762	000440	000026	1\$: MOV	#DMD!MCLK, RKMR1(R2)	
6954	043052	012762	000040	000026	MOV	#DMD, RKMR1(R2)	
6955	043060	005300			DEC	R0	
6956	043062	001370			BNE	1\$	
6957	043064	005062	000026		CLR	RKMR1(R2)	:: FINISH COMMAND IN NORMAL MODE
6958	043070	013700	003632		MOV	WAITIM, R0	:: WAIT FOR READY
6959	043074	005762	000000		2\$: TSTB	RKCS1(R2)	
6960	043100	100402			BMI	3\$	
6961	043102	005300			DEC	R0	
6962	043104	001373			BNE	2\$	
6963	043106	016237	000000	003440	3\$: MOV	RKCS1(R2), T.CS1	:: STORE COMMAND AND STATUS REG 1
6964	043114	016237	000010	003450	MOV	RKCS2(R2), T.CS2	:: STORE COMMAND AND STATUS REG 2
6965	043122	016237	000012	003452	MOV	RKDS(R2), T.DS	:: STORE DRIVE STATUS REG
6966	043130	016237	000014	003454	MOV	RKER(R2), T.ER	:: STORE ERROR REG
6967	043136	012737	100224	003500	MOV	#CERR!RDY!RDHEAD, <T.CS1>	:: LOAD EXPECTED CS1
6968	043144	012737	000100	003510	MOV	#IR, E.CS2	:: LOAD EXPECTED CS2
6969	043152	012737	100001	003512	MOV	#SVAL!DRA, E.DS	:: LOAD EXPECTED DRIVE STATUS REG
6970	043160	012737	000020	003514	MOV	#FMTE, E.ER	:: LOAD EXPECTED ERROR REG
6971	043166	023737	003500	003440	CMP	E.CS1, T.CS1	:: CHECK COMMAND AND STATUS REG 1 CORRECT
6972	043174	001401			BEQ	4\$:: YES, CONTINUE
6973	043176	104175			ERROR	175	
6974	043200	023737	003510	003450	4\$: CMP	E.CS2, T.CS2	:: CHECK COMMAND AND STATUS REG 2 CORRECT
6975	043206	001401			BEQ	5\$:: YES, CONTINUE
6976	043210	104176			ERROR	176	
6977	043212	023737	003512	003452	5\$: CMP	E.DS, T.DS	:: CHECK IF DRIVE STATUS REG CORRECT
6978	043220	001401			BEQ	6\$:: YES, CONTINUE
6979	043222	104177			ERROR	177	
6980	043224	023737	003514	003454	6\$: CMP	E.ER, T.ER	:: CHECK IF ERR REG CORRECT
6981	043232	001401			BEQ	7\$:: YES, CONTINUE
6982	043234	104200			ERROR	200	
6983	043236	013737	003440	003540	7\$: MOV	T.CS1, P.CS1	:: STORE PREVIOUS CONTENTS OF
6984	043244	013737	003450	003550	MOV	T.CS2, P.CS2	:: COMMAND AND STATUS REG 1
6985	043252	013737	003452	003552	MOV	T.DS, P.DS	:: COMMAND AND STATUS REG 2
6986	043260	013737	003454	003554	MOV	T.ER, P.ER	:: DRIVE STATUS REG
6987							:: AND ERROR REG
6988	043266	012762	100000	000000	MOV	#CLR, RKCS1(R2)	:: CLEAR RK611
6989	043274	016237	000000	003440	MOV	RKCS1(R2), T.CS1	:: STORE COMMAND AND STATUS REG 1
6990	043302	016237	000010	003450	MOV	RKCS2(R2), T.CS2	:: STORE COMMAND AND STATUS REG 2
6991	043310	016237	000012	003452	MOV	RKDS(R2), T.DS	:: STORE DRIVE STATUS REG
6992	043316	016237	000014	003454	MOV	RKER(R2), T.ER	:: STORE ERROR REG
6993	043324	012737	000200	003500	MOV	#RDY, E.CS1	:: LOAD EXPECTED CS1
6994	043332	012737	000100	003510	MOV	#IR, E.CS2	:: LOAD EXPECTED CS2
6995	043340	005037	003512		CLR	E.DS	:: LOAD EXPECTED DRIVE STATUS REG
6996	043344	005037	003514		CLR	E.ER	:: LOAD EXPECTED ERROR REG
6997	043350	023737	003500	003440	CMP	E.CS1, T.CS1	:: CHECK CS1 CORRECT
6998	043356	001401			BEQ	11\$:: YES, CONTINUE
6999	043360	104211			ERROR	211	:: CS1 INCORRECT
7000	043362	023737	003510	003450	11\$: CMP	E.CS2, T.CS2	:: CHECK CS2 CORRECT
7001	043370	001401			BEQ	12\$:: YES, CONTINUE

```

7002 043372 104212          ERROR 212          ;CS2 INCORRECT
7003 043374 023737 003512 003452 12$:  CMP      E.DS,T.DS    ;CHECK DRIVE STATUS CORRECT
7004 043402 001401          BEQ      13$          ;YES, CONTINUE
7005 043404 104213          ERROR 213          ;DRIVE STATUS REG INCORRECT
7006 043406 023737 003514 003454 13$:  CMP      E.ER,T.ER   ;CHECK IF ERROR REG CORRECT
7007 043414 001401          BEQ      14$          ;YES, GO ON TO NEXT TEST
7008 043416 104214          ERROR 214          ;ERROR REG INCORRECT
7009 043420          14$:
7010
7011
7012  ;*****
7013  ;*TEST 52          FORMAT ERROR (PART 2)
7014  ;*
7015  ;*          CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR.  PUT THE
7016  ;*          CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN
7017  ;*          RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0.
7018  ;*          CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6.  TURN OFF
7019  ;*          MAINTENANCE MODE.  MAKE SURE FORMAT ERROR,
7020  ;*          DRIVE AVAILABLE AND CONTROLLER ERROR SET.
7021  ;*****
7022  043420 000004          TST52: SCOPE
7023  043422 012737 000144 001200  MOV      #100, $TIMES ;DO 100. ITERATIONS
7024  043430 013702 001270          MOV      $BASE,R2    ;LOAD RK611 BASE
7025  043434 012762 000040 000010  MOV      #SCLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
7026  043442 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
7027  043450 012762 000003 000020  MOV      #3,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG
7028  043456 012762 010025 000000  MOV      #RDHEAD:CFMT,RKCS1(R2) ;ISSUE READ HEADER
7029  043464 012700 000132          MOV      #22,*4+2,R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6
7030  043470 012762 000440 000026 1$:  MOV      #DMD:MCLK,RKMR1(R2)
7031  043476 012762 000040 000026  MOV      #DMD,RKMR1(R2)
7032  043504 005300          DEC      R0
7033  043506 001370          BNE     1$
7034  043510 005062 000026          CLR     RKMR1(R2) ;FINISH COMMAND IN NORMAL MODE
7035  043514 013700 003632          MOV     WAITIM,R0 ;WAIT FOR READY
7036  043520 105762 000000          2$:  TSTB   RKCS1(R2)
7037  043524 100402          BMI     3$
7038  043526 005300          DEC     R0
7039  043530 001373          BNE     2$
7040  043532 016237 000000 003440 3$:  MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
7041  043540 016237 000010 003450  MOV     RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
7042  043546 016237 000012 003452  MOV     RKDS(R2),T.DS ;STORE DRIVE STATUS REG
7043  043554 016237 000014 003454  MOV     RKER(R2),T.ER ;STORE ERROR REG
7044  043562 012737 110224 003500  MOV     #CERR:RDY:RDHEAD:CFMT<↑C<GO>>,E.CS1 ;LOAD EXPECTED CS1
7045  043570 012737 000100 003510  MOV     #IR,E.CS2 ;LOAD EXPECTED CS2
7046  043576 012737 100001 003512  MOV     #SVAL:DRA,E.DS ;LOAD EXPECTED DRIVE STATUS REG
7047  043604 012737 000030 003514  MOV     #FMTE:DRPAR,E.ER ;LOAD EXPECTED ERROR REG
7048  043612 023737 003500 003440  CMP     E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
7049  043620 001401          BEQ     4$          ;YES, CONTINUE
7050  043622 104201          ERROR 201
7051  043624 023737 003510 003450 4$:  CMP     E.CS2,T.CS2 ;CHECK COMMAND AND STATU REG 2 CORRECT
7052  043632 001401          BEQ     5$          ;YES, CONTINUE
7053  043634 104202          ERROR 202
7054  043636 023737 003512 003452 5$:  CMP     E.DS,T.DS ;CHECK IF DRIVE STRATUS REG CORRECT
7055  043644 001401          BEQ     6$          ;YES, CONTINUE
7056  043646 104203          ERROR 203
7057  043650 023737 003514 003454 6$:  CMP     E.ER,T.ER ;CHECK IF ERR REG CORRECT

```

```

7058 043656 001401 BEQ 7$ ;YES, CONTINUE
7059 043660 104204 ERROR 204
7060 043662 013737 003440 003540 7$: MOV T.CS1,P.CS1 ;STORE PREVIOUS CONTENTS OF
7061 043670 013737 003450 003550 MOV T.CS2,P.CS2 ;COMMAND AND STATUS REG 1
7062 043676 013737 003452 003552 MOV T.DS,P.DS ;COMMAND AND STATUS REG 2
7063 043704 013737 003454 003554 MOV T.ER,P.ER ;DRIVE STATUS REG
7064 ;AND ERROR REG
7065 043712 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
7066 043720 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
7067 043726 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
7068 043734 016237 000012 003452 MOV RKDS(R2),T.DS ;STORE DRIVE STATUS REG
7069 043742 016237 000014 003454 MOV RKER(R2),T.ER ;STORE ERROR REG
7070 043750 012737 000200 003500 MOV #RDY,E.CS1 ;LOAD EXPECTED CS1
7071 043756 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
7072 043764 005037 003512 CLR E.DS ;LOAD EXPECTED DRIVE STATUS REG
7073 043770 005037 003514 CLR E.ER ;LOAD EXPECTED ERROR REG
7074 043774 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
7075 044002 001401 BEQ 11$ ;YES, CONTINUE
7076 044004 104211 ERROR 211 ;CS1 INCORRECT
7077 044006 023737 003510 003450 11$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
7078 044014 001401 BEQ 12$ ;YES, CONTINUE
7079 044016 104212 ERROR 212 ;CS2 INCORRECT
7080 044020 023737 003512 003452 12$: CMP E.DS,T.DS ;CHECK DRIVE STATUS CORRECT
7081 044026 001401 BEQ 13$ ;YES, CONTINUE
7082 044030 104213 ERROR 213 ;DRIVE STATUS REG INCORRECT
7083 044032 023737 003514 003454 13$: CMP E.ER,T.ER ;CHECK IF ERROR REG CORRECT
7084 044040 001401 BEQ 14$ ;YES, GO ON TO NEXT TEST
7085 044042 104214 ERROR 214 ;ERROR REG INCORRECT
7086 044044 14$:
7087
7088 ;*****
7089 ;*TEST 53 FAULT SETTING CONTROLLER ERROR
7090 ;*
7091 ;* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
7092 ;* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO
7093 ;* AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE
7094 ;* 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6.
7095 ;* TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE
7096 ;* AVAILABLE AND CONTROLLER ERROR SET.
7097 ;*
7098 ;*****
7099 ;*ST53: SCOPE
7100 044044 000004 MOV #100,$TIMES ;DO 100. ITERATIONS
7101 044046 012737 000144 001200 MOV $BASE,R2 ;LOAD RK611 BASE
7102 044054 013702 001270 MOV #SCLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
7103 044060 012762 000040 000010 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
7104 044066 012762 000040 000026 MOV #3,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG
7105 044074 012762 000003 000020 MOV #-3,RKWC(R2) ;LOAD WORD COUNT
7106 044102 012762 177775 000002 MOV #WRBUFF,RKBA(R2) ;LOAD BUS ADDRESS
7107 044110 012762 067264 000004 MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
7108 044116 012762 000027 000000 MOV #22,*4+2,R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6
7109 044124 012700 000132 MOV #DMD!MCLK,RKMR1(R2)
7110 044130 012762 000440 000026 1$: MOV #DMD,RKMR1(R2)
7111 044136 012762 000040 000026 DEC R0
7112 044144 005300 BNE 1$
7113 044150 005062 000026 CLR RKMR1(R2) ;FINISH COMMAND IN NORMAL MODE

```



```

7167
7168
7169
7170
7171
7172
7173
7174
7175
7176 044504
7177 044504 000004
7178 044506 005037 001102
7179 044512 005037 001200
7180 044516 005237 001222
7181 044522 042737 100000 001222
7182 044530 005327
7183 044532 000001
7184 044534 003063
7185 044536 012737
7186 044540 000001
7187 044542 044532
7188 044544 104401 044552
7189 044550 000407
7190
7191 044570
7192 044570 013746 001222
7193
7194 044574 104405
7195 044576 104401 044604
7196 044602 000421
7197
7198 044646
7199 044646 013746 001112
7200
7201 044652 104405
7202 044654 104401 001211
7203 044660 005037 001112
7204 044664 013700 000042
7205 044670 001405
7206 044672 000005
7207 044674 004710
7208 044676 000240
7209 044700 000240
7210 044702 000240
7211 044704
7212 044704 000137
7213 044706 004654
7214 044710 377 377 000
7215 044714
7216
7217
7218
7219 044714 013703 003640
7220 044720 012713 000004
7221 044724 012737 000157 067300
7222 044732 012713 000001

```

```

.SBTTL END OF PASS ROUTINE
*****
;INCREMENT THE PASS NUMBER ($PASS)
;TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
;WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
;IF THERE IS A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO NEWPAS
SEOP:
  SCOPE
  CLR $STNM ;ZERO THE TEST NUMBER
  CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
  INC $PASS ;INCREMENT THE PASS NUMBER
  BIC #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
  DEC (PC)+ ;LOOP?
SEOPCT: .WORD 1
  BGT $DOAGN ;YES
  MOV (PC)+,2(PC)+ ;RESTORE COUNTER
SENDCT: .WORD 1
  $EOPCT
  TYPE ,65$ ;TYPE ASCIZ STRING
  BR ,64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ <12><15>/END PASS #/
;64$:
  MOV $PASS,-(SP) ;SAVE $PASS FOR TYPEOUT
  ;TYPE PASS NUMBER
  TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
  TYPE ,67$ ;TYPE ASCIZ STRING
  BR ,66$ ;GET OVER THE ASCIZ
;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
;66$:
  MOV $ERTTL,-(SP) ;SAVE $ERTTL FOR TYPEOUT
  ;TOTAL NUMBER OF ERRORS
  TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
  TYPE , $CRLF ;TYPE CARRIAGE RETURN, LINE FEED
  CLR $ERTTL ;CLEAR ERROR TOTAL
  MOV ,2#42,RO ;GET MONITOR ADDRESS
  BEQ $DOAGN ;BRANCH IF NO MONITOR
  RESET ;CLEAR THE WORLD
SENDAD: JSR PC,(RO) ;GO TO MONITOR
  NOP ;SAVE ROOM
  NOP ;FOR
  NOP ;ACT11
$DOAGN:
  JMP ,2(PC)+ ;RETURN
$RTNAD: .WORD NEWPAS
$ENULL: .BYTE -1,-1,0 ;NULL CHARACTER STRING
  .EVEN
.SBTTL GENERATE BAD PARITY IN MEMORY
WRTPAR: MOV CSRPTR,R3 ;GET ADDRESS OF CSR TO BE USED
  MOV #WR.PAR,(R3) ;ALLOW SETTING OF BAD PARITY
  MOV #157,BADPAR ;WRITE BAD PARITY
  MOV #PAR.EN,(R3) ;ALLOW PARITY DETECTION

```

```

7223 044736 000207          RTS      PC          ;RETURN
7224
7225          .SBTTL  CHECK FOR MEMORY CHECK ENABLE
7226
7227 044740 012737 045074 000004 PARCHK: MOV      #20$,ERRVEC      ;SET VECTOR FOR MEMORY PARITY CHECK
7228 044746 012737 000340 000006      MOV      #PR7,ERRVEC+2
7229          ;      MOV      #0,BADPAR      ;LOAD GOOD PARITY
7230 044754 005037 003630      CLR      MEMPAR      ;CLEAR FLAG
7231 044760 005037 003636      CLR      PARPRE      ;CLEAR PARITY PRESENT FLAG
7232 044764 012703 172100      MOV      #MEMBAS,R3      ;LOAD REGISTER TO DETERMINE IF
7233          ;      MEMORY CHECK ENABLE AVAILABLE
7234 044770 042713 000007      BIC      #7,(R3)      ;DISABLE IN ORDER NOT GET INTERR
7235 044774 012737 000000 067300      MOV      #0,BADPAR      ;WRITE GOOD PARITY
7236 045002 012704 000001      MOV      #1,R4      ;INITIALIZE MASK
7237 045006 012713 000001 16$:      MOV      #PAR.EN,(R3)      ;ENABLE MEMORY CHECK
7238 045012 005713      TST      (R3)
7239          ;      BIS      R4,MEMPAR      ;SET PARITY PRESENT BIT
7240          ;      TST      PARPRE      ;TEST IF PARITY PRESENT ALREADY SET
7241          ;      BNE      10$      ;YES - SKIP
7242 045014 012737 045100 000114      MOV      #30$,MEMVEC      ;SET UP PARITY TRAP VECTOR
7243 045022 012737 000340 000116      MOV      #PR7,MEMVEC+2
7244 045030 012713 000004      MOV      #WR.PAR,(R3)      ;ALLOW SETTING OF BAD PARITY
7245 045034 012737 177777 067300      MOV      #-1,BADPAR      ;TRY SET BAD PARITY IN BADPAR
7246 045042 012713 000001      MOV      #PAR.EN,(R3)      ;SET TO TRAP ON PARITY ERROR
7247 045046 005737 067300      TST      BADPAR      ;CAUSE TRAP IF BAD PARITY
7248
7249 045052 005037 003636      CLR      PARPRE      ;COME TO THIS POINT IF NO TRAP
7250 045056 000424      BR       35$      ;TO BYPASS T22,24 IF ANY MEM NOT TRX PARITY
7251          ;      CHECK BIT
7252 045060 062703 000002 10$:      ADD      #2,R3      ;BUMP TO NEXT CSR
7253 045064 000241      CLC
7254 045066 006104      ROL      R4      ;ROTATE MASK
7255 045070 001346      BNE      16$      ;SKIP IF MASK 0
7256 045072 000416      BR       35$      ;ELSE GO TO EXIT
7257 045074 022626 20$:      CMP      (SP)+,(SP)+      ;CLEAN STACK
7258 045076 000770      BR
7259
7260 045100 022626 30$:      CMP      (SP)+,(SP)+      ;CLEAR STACK
7261 045102 005013      CLR      (R3)      ;CLEAR CSR REGISTER
7262 045104 005037 067300      CLR      BADPAR      ;CLEAN UP BAD PARITY
7263 045110 012713 000001      MOV      #PAR.EN,(R3)      ;SET TO DETECT PARITY ERROR
7264 045114 010337 003640      MOV      R3,CSRPTR      ;STORE CSR TO BE USED IN TESTS
7265 045120 012737 177777 003636      MOV      #-1,PARPRE      ;SET PARITY PRESENT FLAG
7266 045126 000754      BR       10$      ;TEST NEXT CSR
7267
7268 045130 012737 000006 000004 35$:      MOV      #ERRVEC+2,ERRVEC      ;RESTORE TRAP CATCHER
7269 045136 005037 000006      CLR      ERRVEC+2
7270 045142 005737 003630      TST      MEMPAR      ;TEST IF ANY PARITY AVAILABLE
7271 045146 001006      BNE      37$      ;YES - SKIP
7272 045150 012737 000116 000114      MOV      #MEMVEC+2,MEMVEC      ;ELSE RESTORE TRAP CATCHER
7273 045156 005037 000116      CLR      MEMVEC+2
7274 045162 000413      BR       40$
7275 045164 012737 046274 000114 37$:      MOV      #MEMERR,MEMVEC      ;SKIP
7276 045172 012737 000340 000116      MOV      #PR7,MEMVEC+2      ;ELSE SET FOR PARITY TRAPS
7277 045200 012746 000000      MOV      #PRO,-(SP)      ;SET PRIORITY 0
7278 045204 012746 045212      MOV      #40$,-(SP)

```

```

7279 045210 000002
7280 045212 000207
7281
7282
7283
7284 045214 052737 002400 003524 WRTBIT: BIS #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT. REG. 1
7285 045222 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2) ;PROVIDE 1ST UPWARD TRANSITION
7286 045230 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1
7287 045236 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK IF MAINT REG 1 CORRECT
7288 045244 001416 BEQ 3$ ;YES, PROVIDE DOWNWARD TRANSITION
7289 045246 012737 045266 001202 MOV #1$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
7290 045254 012737 066403 003172 MOV #EMW1,EMW+2 ;LOAD ERROR MESSAGE
7291 045262 011646 MOV (SP),-(SP) ;SAVE RETURN
7292 045264 000207 RTS PC ;MR1 INCORRECT ON UPWARD TRANSITION
7293
7294 045266 032777 001000 133644 1$: BIT #SW9, $SWR ;CHECK IF LOOP ON ERROR
7295 045274 001402 BEQ 3$ ;NO CONTINUE
7296 045276 000137 046150 JMP 63$ ;YES, LOOP ON ERROR
7297
7298 045302 042737 014400 003524 3$: BIC #MCLK!PCA!PCD,E.MR1 ;INITIALIZE MAINTENANCE REG. 1
7299 045310 052737 042000 003524 BIS #MEWD!WRTGAT,E.MR1
7300 045316 005737 003614 TST PR.BIT ;CHECK IF ONE
7301 045322 001152 BNE 20$ ;YES, SIMULATE ONE
7302 045324 005737 003616 TST M1.BIT ;CHECK IF PREVIOUS ONE
7303 045330 001023 BNE 10$ ;YES, NO TRANSITION
7304 045332 042737 002000 003524 BIC #MEWD,E.MR1 ;INDICATE TRANSITION
7305 045340 005737 003612 TST P1.BIT ;CHECK IF NEXT BIT = 1
7306 045344 001007 BNE 5$ ;YES, CHECK FOR PRECOMP ADVANCE
7307 045346 005737 003620 TST M2.BIT ;CHECK FOR PRECOMP. ADVANCE
7308 045352 001412 BEQ 10$ ;NO CLOCK IN ZERO
7309 045354 052737 010000 003524 BIS #PCD,E.MR1 ;SET PRECOMP. DELAY
7310 045362 000406 BR 10$ ;CLOCK IN ZERO
7311
7312 045364 005737 003620 5$: TST M2.BIT ;CHECK FOR PRECOMP. ADVANCE
7313 045370 001003 BNE 10$ ;CLOCK IN ZERO
7314 045372 052737 004000 003524 BIS #PCA,E.MR1 ;SET PRECOMP. ADVANCE
7315 045400 012762 000040 000026 10$: MOV #DMD,RKMR1(R2) ;CLOCK IN DATA BIT
7316 045406 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MR1
7317 045414 023737 003464 003524 CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
7318 045422 001416 BEQ 12$ ;YES, CONTINUE
7319 045424 012737 045444 001202 MOV #11$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
7320 045432 012737 066472 003172 MOV #EMW2,EMW+2 ;LOAD ERROR MESSAGE
7321 045440 011646 MOV (SP),-(SP) ;SAVE RETURN
7322 045442 000207 RTS PC ;MR1 INCORRECT
7323
7324 045444 032777 001000 133466 11$: BIT #SW9, $SWR ;CHECK IF LOOP ON ERROR
7325 045452 001402 BEQ 12$ ;NO CONTINUE
7326 045454 000137 046150 JMP 63$ ;YES, LOOP ON ERROR
7327
7328 045460 052737 002400 003524 12$: BIS #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT REG 1
7329 045466 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSITION
7330 045474 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1
7331 045502 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK IF MAINT REG. 1 CORRECT
7332 045510 001416 BEQ 15$ ;YES, CONTINUE
7333 045512 012737 045532 001202 MOV #13$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
7334 045520 012737 066563 003172 MOV #EMW3,EMW+2 ;LOAD ERROR MESSAGE

```

```

7335 045526 011646          MOV      (SP),-(SP)      ;SAVE RETURN
7336 045530 000207          RTS      PC              ;MRI INCORRECT
7337
7338 045532 032777 001000 133400 13$:  BIT      #SW9,2SWR      ;CHECK IF LOOP ON ERROR
7339 045540 001402          BEQ      15$            ;NO CONTINUE
7340 045542 000137 046150          JMP      63$            ;YES, LOOP ON ERROR
7341
7342 045546 052737 002000 003524 15$:  BIS      #MEWD,E.MR1    ;RESET TRANSITION INDICATION
7343 045554 042737 000400 003524          BIC      #MCLK,E.MR1
7344 045562 012762 000040 000026          MOV      #DMD,RKMR1(R2) ;SUPPLY LAST PART OF DATA
7345 045570 016237 000026 003464          MOV      RKMR1(R2),T.MR1 ;STORE MR1
7346 045576 023737 003464 003524          CMP      T.MR1,E.MR1    ;CHECK IF MR1 CORRECT
7347 045604 001414          BEQ      18$            ;YES, RETURN
7348 045606 012737 045626 001202          MOV      #17$, $ESCAPE  ;LOAD ESCAPE FOR LOOP ON ERROR
7349 045614 012737 066652 003172          MOV      #EMW4,EMW+2    ;LOAD ERROR MESSAGE
7350 045622 011646          MOV      (SP),-(SP)    ;SAVE RETURN
7351 045624 000207          RTS      PC              ;MRI INCORRECT
7352
7353 045626 032777 001000 133304 17$:  BIT      #SW9,2SWR      ;CHECK IF LOOP ON ERROR
7354 045634 001145          BNE      63$            ;YES, LOOP ON ERROR
7355 045636 005037 001202 18$:  CLR      $ESCAPE        ;CLEAR ESCAPE
7356 045642 062716 000002          ADD      #2,(SP)        ;ADJUST RETURN
7357 045646 000207          RTS      PC              ;RETURN
7358
7359 045650 005737 003612 20$:  TST      P1.BIT         ;CHECK IN NEXT BIT A ONE
7360 045654 001007          BNE      30$            ;YES, CHECK IF PRECOMP DELAY
7361 045656 005737 003616          TST      M1.BIT         ;CHECK FOR PRECOMP ADVANCE
7362 045662 001415          BEQ      40$            ;NO CLOCK IN DATA BIT
7363 045664 052737 004000 003524          BIS      #PCA,E.MR1    ;SET PRECOMP. ADVANCE
7364 045672 000411          BR       40$            ;CHECK MR1
7365
7366 045674 042737 000400 003524 30$:  BIC      #MCLK,E.MR1    ;RESET MAINT CLOCK IN EXPECTED MR1
7367 045702 005737 003616          TST      M1.BIT         ;CHECK FOR PRECOMP DELAY
7368 045706 001003          BNE      40$            ;NO CHECK MR1
7369 045710 052737 010000 003524          BIS      #PCD,E.MR1    ;SET SET PRECOMP DELAY
7370 045716 012762 000040 000026 40$:  MOV      #DMD,RKMR1(R2) ;CLOCK IN DATA BIT
7371 045724 016237 000026 003464          MOV      RKMR1(R2),T.MR1 ;STORE MR1
7372 045732 023737 003464 003524          CMP      T.MR1,E.MR1    ;CHECK MR1 CORRECT
7373 045740 001414          BEQ      42$            ;YES, CLOCK IN REST OF BIT
7374 045742 012737 045762 001202          MOV      #41$, $ESCAPE  ;LOAD ESCAPE FOR LOOP ON ERROR
7375 045750 012737 066472 003172          MOV      #EMW2,EMW+2    ;LOAD ERROR MESSAGE
7376 045756 011646          MOV      (SP),-(SP)    ;SAVE RETURN
7377 045760 000207          RTS      PC              ;MRI INCORRECT
7378
7379 045762 032777 001000 133150 41$:  BIT      #SW9,2SWR      ;CHECK IF LOOP ON ERROR
7380 045770 001067          BNE      63$            ;YES, LOOP ON ERROR
7381 045772 052737 000400 003524 42$:  BIS      #MCLK,E.MR1    ;CREATE EXPECTED MAINT. REG. 1
7382 046000 012762 000440 000026          MOV      #DMD:MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSITION
7383 046006 016237 000026 003464          MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG 1
7384 046014 023737 003524 003464          CMP      E.MR1,T.MR1    ;CHECK IF MAINT REG 1 CORRECT
7385 046022 001414          BEQ      45$            ;YES, CONTINUE
7386 046024 012737 046044 001202          MOV      #43$, $ESCAPE  ;LOAD ESCAPE
7387 046032 012737 066563 003172          MOV      #EMW3,EMW+2    ;LOAD ERROR MESSAGE
7388 046040 011646          MOV      (SP),-(SP)    ;SAVE RETURN
7389 046042 000207          RTS      PC              ;MRI INCORRECT
7390

```

SIMULATE ONE BIT OF WRITE DATA IN MAINTENANCE MODE

```

7391 046044 032777 001000 133066 43$: BIT #SW9, QSWR ;CHECK IF LOOP ON ERROR
7392 046052 001036 BNE 63$ ;YES, LOOP ON ERROR
7393 046054 042737 002400 003524 45$: BIC #MWD, MCLK, E, MR1 ;SET TRANSITION
7394 046062 012762 000040 000026 MOV #DMD, RKMR1(R2) ;CLOCK TRANSITION
7395 046070 016237 000026 003464 MOV RKMR1(R2), T.MR1 ;STORE MR1
7396 046076 023737 003464 003524 CMP T.MR1, E.MR1 ;CHECK IF MR1 CORRECT
7397 046104 001414 BEQ 50$ ;YES, RETURN
7398 046106 012737 046126 001202 MOV #47$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
7399 046114 012737 066652 003172 MOV #EMW4, EMW+2 ;LOAD ERROR MESSAGE
7400 046122 011646 MOV (SP), -(SP) ;SAVE RETURN
7401 046124 000207 RTS PC ;MR1 INCORRECT
7402
7403 046126 032777 001000 133004 47$: BIT #SW9, QSWR ;CHECK IF LOOP ON ERROR
7404 046134 001005 BNE 63$ ;YES, LOOP ON ERROR
7405 046136 005037 001202 50$: CLR $ESCAPE ;CLEAR ESCAPE
7406 046142 062716 000002 ADD #2, (SP) ;ADJUST RETURN
7407 046146 000207 RTS PC ;RETURN
7408
7409 046150 005037 001202 63$: CLR $ESCAPE ;CLEAR ESCAPE
7410 046154 012706 001100 MOV #STACK, SP ;FORCE STACK
7411 046160 000177 132724 JMP Q$LPERR ;LOOP ON ERROR
7412
7413 .SBTTL SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE
7414
7415 046164 005737 003614 RDBIT: TST PR.BIT ;CHECK IF ONE
7416 046170 001024 BNE 10$ ;YES, SIMULATE ONE
7417 046172 005737 003616 TST M1.BIT ;CHECK IF PREVIOUS ONE
7418 046176 001404 BEQ 4$ ;NO, INSERT TRANSITION
7419 046200 012762 000440 000026 MOV #DMD, MCLK, RKMR1(R2) ;YES, DO NOT INSERT TRANSITION
7420 046206 000403 BR 5$ ;CLOCK IN ZERO
7421
7422 046210 012762 001440 000026 4$: MOV #DMD, MCLK, MERD, RKMR1(R2) ;INSERT TRANSITION
7423 046216 012762 000040 000026 5$: MOV #DMD, RKMR1(R2) ;CLOCK IN ZERO
7424 046224 012762 000440 000026 MOV #DMD, MCLK, RKMR1(R2)
7425 046232 012762 000040 000026 MOV #DMD, RKMR1(R2)
7426 046240 000207 RTS PC ;RETURN
7427
7428 046242 012762 000440 000026 10$: MOV #DMD, MCLK, RKMR1(R2) ;CLOCK IN ONE
7429 046250 012762 000040 000026 MOV #DMD, RKMR1(R2)
7430 046256 012762 001440 000026 MOV #DMD, MCLK, MERD, RKMR1(R2)
7431 046264 012762 000040 000026 MOV #DMD, RKMR1(R2)
7432 046272 000207 RTS PC ;RETURN
7433
7434 .SBTTL MEMORY CHECK ENABLE TRAP
7435
7436 046274 012737 046310 001202 MEMERR: MOV #10$, $ESCAPE ;LOAD ESCAPE
7437 046302 011637 003604 MOV (SP), TRAPPC ;STORE PC
7438 046306 104147 ERROR 147 ;REPORT MEM PARITY ERROR
7439 046310 005037 001202 10$: CLR $ESCAPE ;CLEAR ESCAPE
7440 046314 032777 001000 132616 BIT #SW9, QSWR ;CHECK IF LOOP ON ERROR
7441 046322 001001 BNE 15$ ;YES, FORCE STACK AND TRY AGAIN
7442 046324 000002 RTI ;NO, RETURN
7443
7444 046326 012706 001100 15$: MOV #STACK, SP ;INITIALIZE STACK
7445 046332 000177 132552 JMP Q$LPERR ;LOOP ON ERROR
7446

```

```

7447 .SBTTL ROUTINE TO SIZE MEMORY
7448
7449 ;*****
7450 *CALL:
7451 * JSR PC,$SIZE
7452 * RETURN
7453 *$LSTAD WILL CONTAIN:
7454 * WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
7455 * WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
7456 *$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
7457 *$KT11 IS THE MEMORY MANAGEMENT KEY
7458 *$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
7459 * MUST BE SETUP BEFORE THE CALL
7460 *$BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
7461 *
7462 * DETERMINED BY ROUTINE
7463 $SIZE: MOV RO,-(SP) ;;SAVE RO ON THE STACK
7464 MOV R1,-(SP) ;;SAVE R1 ON THE STACK
7465 MOV R2,-(SP) ;;SAVE R2 ON THE STACK
7466 MOV R3,-(SP) ;;SAVE R3 ON THE STACK
7467 MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
7468 MOV @#ERRVEC+2,-(SP)
7469 MOV SP,RO ;;SAVE THE STACK POINTER
7470 ;;SET THE ERRVEC PS TO THE PRESENT PS
7471 TRAP ;;PUSH OLD PSW AND PC ON STACK
7472 MOV (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
7473 MOV #3776,R1 ;;SETUP ADDRESS
7474 TSTB (PC)+ ;;USE MEMORY MANAGEMENT?
7475 $KT11: .WORD 200 ;;SET TO USE MEMORY MANAGEMENT
7476 BPL SCORE ;;BR IF NO
7477 MOV #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
7478 TST @#SR0 ;;KT11 ARE YOU THERE?
7479 BIS #100000,$KT11 ;;YES--SET KT11 KEY
7480 CLR -(SP) ;;INITIALIZE FOR "PAR" LOADING
7481 MOV #KIPAR0,R2 ;;ADDRESS OF FIRST "PAR"
7482 MOV #1DB,R3 ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
7483 MOV #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
7484 MOV (SP),(R2)+ ;;LOAD "PAR"
7485 ADD #200,(SP) ;;UPDATE FOR NEXT "PAR"
7486 SOB R3,1$ ;;LOOP UNTIL ALL EIGHT ARE LOADED
7487 MOV #177600,-(R2) ;;SETUP KIPAR7 FOR I/O
7488 CLR -(R2) ;;SETUP KIPAR6 FOR TESTING
7489 MOV #2$,@#ERRVEC ;;CATCH TIMEOUT IF NO SR3
7490 MOV #20,@#SR3 ;;ENABLE 22 BIT MODE
7491 BR 3$ ;;THIS PDP-11 HAS A SR3 REGISTER
7492 CMP (SP)+,(SP)+ ;;CLEAN OFF THE STACK--NO SR3
7493 INC @#SR0 ;;TURN ON MEMORY MANAGEMENT
7494 MOV #$SKTOUT,@#ERRVEC ;;SET FOR TIME OUT
7495 TST @#143776 ;;TRAP ON NON-EX-MEM
7496 ADD #40,(R2) ;;MAKE A 1K STEP
7497 CMP @#KIPAR7,(R2) ;;LAST ONE?
7498 BHI 4$ ;;NO--TRY IT
7499 SKTOUT: MOV (R2),R2 ;;GET LAST BANK+1
7500 CLR @#SR0 ;;TURN OFF MEMORY MANAGEMENT
7501 BR $SIZEX
7502 SKTNEX: BIC #100000,$KT11 ;;KT11 NON-EXISTENT

```

```

7503 046544 012737 046574 000004 SCORE: MOV #SCROUT, @#ERRVEC ;; SET FOR TIMEOUT
7504 046552 005002 CLR R2 ;; SET UP BANK
7505 046554 062701 004000 1$: ADD #4000, R1 ;; INCREMENT BY 1K
7506 046560 062702 000040 ADD #40, R2 ;; 1K STEP
7507 046564 005711 TST (R1) ;; TRAP ON TIME OUT
7508 046566 022701 177776 CMP #177776, R1 ;; LAST ONE
7509 046572 001370 BNE 1$ ;; NO--TRY AGAIN
7510 046574 162701 004000 SCROUT: SUB #4000, R1
7511 046600 162702 000040 S$SIZE: SUB #40, R2 ;; DROP BACK
7512 046604 010006 MOV RO, SP ;; RESTORE THE STACK
7513 046606 012637 000006 MOV (SP)+, @#ERRVEC+2 ;; RESTORE ERROR VECTOR
7514 046612 012637 000004 MOV (SP)+, @#ERRVEC
7515 046616 010137 046640 MOV R1, $LSTAD ;; LAST ADDRESS
7516 046622 010237 046642 MOV R2, $LSTBK ;; LAST BANK
7517 046626 012603 MOV (SP)+, R3 ;; RESTORE R3
7518 046630 012602 MOV (SP)+, R2 ;; RESTORE R2
7519 046632 012601 MOV (SP)+, R1 ;; RESTORE R1
7520 046634 012600 MOV (SP)+, RO ;; RESTORE RO
7521 046636 000207 RTS PC
7522 046640 000000 $LSTAD: .WORD 0 ;; CONTAINS THE LAST ADDRESS
7523 046642 000000 $LSTBK: .WORD 0 ;; CONTAINS THE LAST BANK
7524 .SBTTL SCOPE HANDLER ROUTINE
7525
7526 ;*****
7527 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7528 ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7529 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7530 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7531 ;*SW14=1 LOOP ON TEST
7532 ;*SW11=1 INHIBIT ITERATIONS
7533 ;*SW09=1 LOOP ON ERROR
7534 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
7535 ;*CALL
7536 ;* SCOPE ;; SCOPE=IOT
7537
7538 $SCOPE:
7539 046644 104407 CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
7540 046646 032777 040000 132264 1$: BIT #BIT14, @SWR ;; LOOP ON PRESENT TEST?
7541 046654 001131 BNE $OVER ;; YES IF SW14=1
7542 ;*****START OF CODE FOR THE XOR TESTER*****
7543 046656 000416 $XTSTR: BR 6$ ;; IF RUNNING ON THE "XOR" TESTER CHANGE
7544 ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
7545 046660 013746 000004 MOV @#ERRVEC, -(SP) ;; SAVE THE CONTENTS OF THE ERROR VECTOR
7546 046664 012737 046704 000004 MOV #5$, @#ERRVEC ;; SET FOR TIMEOUT
7547 046672 005737 177060 TST @#177060 ;; TIME OUT ON XOR?
7548 046676 012637 000004 MOV (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
7549 046702 000500 BR $SVLAD ;; GO TO THE NEXT TEST
7550 046704 022626 5$: CMP (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
7551 046706 012637 000004 MOV (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
7552 046712 000440 BR 7$ ;; LOOP ON THE PRESENT TEST
7553 046714 6$: ;*****END OF CODE FOR THE XOR TESTER*****
7554 046714 032777 000400 132216 BIT #BIT08, @SWR ;; LOOP ON SPEC. TEST?
7555 046722 001421 BEQ 2$ ;; BR IF NO
7556 046724 005046 CLR -(SP) ;; CLEAR A TEMP. LOCATION
7557 046726 117716 132206 MOVB @SWR, (SP) ;; PICKUP THE DESIRED TEST NUMBER
7558 046732 001414 BEQ 8$ ;; BRANCH IF BAD TEST NUMBER IN SWR

```

7559	046734	022716	000053					CMP	#53,(SP)	::CHECK THE NUMBER IN THE SWR
7560	046740	002411						BLT	8\$::BRANCH IF TEST NUMBER IS OUT OF RANGE
7561	046742	011637	001102					MOV	(SP), \$STNM	::UPDATE THE TEST NUMBER
7562	046746	005316						DEC	(SP)	::BACKUP BY ONE
7563	046750	006316						ASL	(SP)	::SCALE THE TEST NUMBER AS AN INDEX
7564	046752	062716	047156					ADD	\$\$SWOBTBL,(SP)	::FORM THE ADDRESS OF TEST POINTER
7565	046756	013637	001106					MOV	2(SF)+,\$LPADR	::SET LOOP ADDRESS TO DESIRED TEST
7566	046762	000466						BR	\$OVER	::GO LOOP ON THE TEST
7567	046764	005726				8\$:		TST	(SP)+	::CLEAN THE BAD TEST NUMBER OFF OF THE STACK
7568	046766	105737	001103			2\$:		TSTB	\$ERFLG	::HAS AN ERROR OCCURRED?
7569	046772	001421						BEQ	3\$::BR IF NO
7570	046774	123737	001115	001103				CMPB	\$ERMAX,\$ERFLG	::MAX. ERRORS FOR THIS TEST OCCURRED?
7571	047002	101015						BHI	3\$::BR IF NO
7572	047004	032777	001000	132126				BIT	#BIT09,\$SWR	::LOOP ON ERROR?
7573	047012	001404						BEQ	4\$::BR IF NO
7574	047014	013737	001110	001106		7\$:		MOV	\$LPERR,\$LPADR	::SET LOOP ADDRESS TO LAST SCOPE
7575	047022	000446						BR	\$OVER	
7576	047024	105037	001103			4\$:		CLRB	\$ERFLG	::ZERO THE ERROR FLAG
7577	047030	005037	001200					CLR	\$TIMES	::CLEAR THE NUMBER OF ITERATIONS TO MAKE
7578	047034	000415						BR	1\$::ESCAPE TO THE NEXT TEST
7579	047036	032777	004000	132074		3\$:		BIT	#BIT11,\$SWR	::INHIBIT ITERATIONS?
7580	047044	001011						BNE	1\$::BR IF YES
7581	047046	005737	001222					TST	\$PASS	::IF FIRST PASS OF PROGRAM
7582	047052	001406						BEQ	1\$::INHIBIT ITERATIONS
7583	047054	005237	001104					INC	\$ICNT	::INCREMENT ITERATION COUNT
7584	047060	023737	001200	001104				CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE
7585	047066	002024						BGE	\$OVER	::BR IF MORE ITERATION REQUIRED
7586	047070	012737	000001	001104		1\$:		MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
7587	047076	013737	047154	001200				MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
7588	047104	105237	001102					MOV	\$STNM,\$STNM	::COUNT TEST NUMBERS
7589	047110	113737	001102	001220		\$SVLAD:		MOVB	\$STNM,\$STNM	::SET TEST NUMBER IN APT MAILBOX
7590	047116	011637	001106					MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
7591	047122	011637	001110					MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
7592	047126	005037	001202					CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
7593	047132	112737	000001	001115				MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7594	047140	013777	001102	131774		\$OVER:		MOV	\$STNM,\$DISPLAY	::DISPLAY TEST NUMBER
7595	047146	013716	001106					MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
7596	047152	000002						RTI		::FIXES PS
7597	047154	003720								::MAX. NUMBER OF ITERATIONS
7598	047156					\$MXCNT:	2000.			
7599	047156	004674				\$SWOBTBL:				
7600	047160	005112						.WORD	TST1+2	::STARTING ADDRESS OF TEST 1
7601	047162	005330						.WORD	TST2+2	::STARTING ADDRESS OF TEST 2
7602	047164	005544						.WORD	TST3+2	::STARTING ADDRESS OF TEST 3
7603	047166	005760						.WORD	TST4+2	::STARTING ADDRESS OF TEST 4
7604	047170	006442						.WORD	TST5+2	::STARTING ADDRESS OF TEST 5
7605	047172	007070						.WORD	TST6+2	::STARTING ADDRESS OF TEST 6
7606	047174	007456						.WORD	TST7+2	::STARTING ADDRESS OF TEST 7
7607	047176	010602						.WORD	TST10+2	::STARTING ADDRESS OF TEST 10
7608	047200	011270						.WORD	TST11+2	::STARTING ADDRESS OF TEST 11
7609	047202	012154						.WORD	TST12+2	::STARTING ADDRESS OF TEST 12
7610	047204	013042						.WORD	TST13+2	::STARTING ADDRESS OF TEST 13
7611	047206	014246						.WORD	TST14+2	::STARTING ADDRESS OF TEST 14
7612	047210	015124						.WORD	TST15+2	::STARTING ADDRESS OF TEST 15
7613	047212	015562						.WORD	TST16+2	::STARTING ADDRESS OF TEST 16
7614	047214	016676						.WORD	TST17+2	::STARTING ADDRESS OF TEST 17
								.WORD	TST20+2	::STARTING ADDRESS OF TEST 20

```
7615 047216 020012 .WORD TST21+2 ;: STARTING ADDRESS OF TEST 21
7616 047220 021126 .WORD TST22+2 ;: STARTING ADDRESS OF TEST 22
7617 047222 022374 .WORD TST23+2 ;: STARTING ADDRESS OF TEST 23
7618 047224 023264 .WORD TST24+2 ;: STARTING ADDRESS OF TEST 24
7619 047226 024120 .WORD TST25+2 ;: STARTING ADDRESS OF TEST 25
7620 047230 024652 .WORD TST26+2 ;: STARTING ADDRESS OF TEST 26
7621 047232 025404 .WORD TST27+2 ;: STARTING ADDRESS OF TEST 27
7622 047234 026136 .WORD TST30+2 ;: STARTING ADDRESS OF TEST 30
7623 047236 026670 .WORD TST31+2 ;: STARTING ADDRESS OF TEST 31
7624 047240 027422 .WORD TST32+2 ;: STARTING ADDRESS OF TEST 32
7625 047242 030154 .WORD TST33+2 ;: STARTING ADDRESS OF TEST 33
7626 047244 030454 .WORD TST34+2 ;: STARTING ADDRESS OF TEST 34
7627 047246 031222 .WORD TST35+2 ;: STARTING ADDRESS OF TEST 35
7628 047250 031612 .WORD TST36+2 ;: STARTING ADDRESS OF TEST 36
7629 047252 032436 .WORD TST37+2 ;: STARTING ADDRESS OF TEST 37
7630 047254 033262 .WORD TST40+2 ;: STARTING ADDRESS OF TEST 40
7631 047256 034106 .WORD TST41+2 ;: STARTING ADDRESS OF TEST 41
7632 047260 034732 .WORD TST42+2 ;: STARTING ADDRESS OF TEST 42
7633 047262 035556 .WORD TST43+2 ;: STARTING ADDRESS OF TEST 43
7634 047264 036402 .WORD TST44+2 ;: STARTING ADDRESS OF TEST 44
7635 047266 037226 .WORD TST45+2 ;: STARTING ADDRESS OF TEST 45
7636 047270 040066 .WORD TST46+2 ;: STARTING ADDRESS OF TEST 46
7637 047272 041112 .WORD TST47+2 ;: STARTING ADDRESS OF TEST 47
7638 047274 042136 .WORD TST50+2 ;: STARTING ADDRESS OF TEST 50
7639 047276 042776 .WORD TST51+2 ;: STARTING ADDRESS OF TEST 51
7640 047300 043422 .WORD TST52+2 ;: STARTING ADDRESS OF TEST 52
7641 047302 044046 .WORD TST53+2 ;: STARTING ADDRESS OF TEST 53
7642 ;: *****
7643 ;: .SBTTL LOOP ON INTERNAL ERROR
7644
7645 047304 032777 001000 131626 SCOP1$: BIT #SW9, @SWR ;: CHECK IF LOOP ON ERROR
7646 047312 001405 BEQ 5$ ;: NO RETURN
7647 047314 105737 001103 TSTB $ERFLG ;: CHECK IF ERROR OCCURED
7648 047320 001402 BEQ 5$ ;: NO, RETURN
7649 047322 013716 001110 MOV $LPERR, (SP) ;: GO BACK TO BEGINNING OF LOOP
7650 047326 000002 5$: RTI ;: RETURN
7651 ;: .SBTTL APT COMMUNICATIONS ROUTINE
7652
7653 ;: *****
7654 047330 112737 000001 047574 $ATY1: MOVB #1, $FFLG ;: TO REPORT FATAL ERROR
7655 047336 112737 000001 047572 $ATY3: MOVB #1, $MFLG ;: TO TYPE A MESSAGE
7656 047344 000403 BR $ATYC
7657 047346 112737 000001 047574 $ATY4: MOVB #1, $FFLG ;: TO ONLY REPORT FATAL ERROR
7658 047354 $ATYC:
7659 047354 010046 MOV R0, -(SP) ;: PUSH R0 ON STACK
7660 047356 010146 MOV R1, -(SP) ;: PUSH R1 ON STACK
7661 047360 105737 047572 TSTB $MFLG ;: SHOULD TYPE A MESSAGE?
7662 047364 001450 BEQ 5$ ;: IF NOT: BR
7663 047366 122737 000001 001234 CMPB #APTENV, $ENV ;: OPERATING UNDER APT?
7664 047374 001031 BNE 3$ ;: IF NOT: BR
7665 047376 132737 000100 001235 BITB #APTSPOOL, $ENVM ;: SHOULD SPOOL MESSAGES?
7666 047404 001425 BEQ 3$ ;: IF NOT: BR
7667 047406 017600 000004 MOV @4(SP), R0 ;: GET MESSAGE ADDR.
7668 047412 062766 000002 000004 ADD #2, 4(SP) ;: BUMP RETURN ADDR.
7669 047420 005737 001214 1$: TST $MSGTYPE ;: SEE IF DONE W/ LAST XMISSION?
7670 047424 001375 BNE 1$ ;: IF NOT: WAIT
```

```

7671 047426 010037 001230
7672 047432 105720
7673 047434 001376
7674 047436 163700 001230
7675 047442 006200
7676 047444 010037 001232
7677 047450 012737 000004 001214
7678 047456 000413
7679 047460 017637 000004 047504 3$:
7680 047466 062766 000002 000004
7681 047474 013746 177776
7682 047500 004737 050336
7683 047504 000000
7684 047506
7685 047506 105737 047574
7686 047512 001416
7687 047514 005737 001234
7688 047520 001413
7689 047522 005737 001214
7690 047526 001375
7691 047530 017637 000004 001216
7692 047536 062766 000002 000004
7693 047544 005237 001214
7694 047550 105037 047574
7695 047554 105037 047573
7696 047560 105037 047572
7697 047564 012601
7698 047566 012600
7699 047570 000207
7700 047572 000
7701 047573 000
7702 047574 000
7703 047576
7704 000200
7705 000001
7706 000100
7707 000040
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722 047576
7723 047576 104407
7724 047600 105237 001103
7725 047604 001775
7726 047606 013777 001102 131326

MOV RO,$MSGAD ;;PUT ADDR IN MAILBOX
TSTB (RO)+ ;;FIND END OF MESSAGE
BNE 2$
SUB $MSGAD,RO ;;SUB START OF MESSAGE
ASR RO ;;GET MESSAGE LNTH IN WORDS
MOV RO,$MSGGLT ;;PUT LENGTH IN MAILBOX
MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
BR 5$
MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
ADD #2,4(SP) ;;BUMP RETURN ADDRESS
MOV 177776,-(SP) ;;PUSH 177776 ON STACK
JSR PC,$TYPE ;;CALL TYPE MACRO
.WORD 0
5$:
10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
BEQ 12$ ;;IF NOT: BR
TST $ENV ;;RUNNING UNDER APT?
BEQ 12$ ;;IF NOT: BR
TST $MSGTYPE ;;FINISHED LAST MESSAGE?
BNE 11$ ;;IF NOT: WAIT
MOV @4(SP),$FATAL ;;GET ERROR #
ADD #2,4(SP) ;;BUMP RETURN ADDR.
INC $MSGTYPE ;;TELL APT TO TAKE ERROR
12$: CLRB $FFLG ;;CLEAR FATAL FLAG
CLRB $LFLG ;;CLEAR LOG FLAG
CLRB $MFLG ;;CLEAR MESSAGE FLAG
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTS PC ;;RETURN
$MFLG: .BYTE 0 ;;MESSG. FLAG
$LFLG: .BYTE 0 ;;LOG FLAG
$FFLG: .BYTE 0 ;;FATAL FLAG
.EVEN
APTSIZE=200
APTENV=001
APTSPOOL=100
APTCSUP=040
.SBTTL ERROR HANDLER ROUTINE
*****
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
*AND GO TO TYPERR ON ERROR
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW15=1 HALT ON ERROR
*SW13=1 INHIBIT ERROR TYPEOUTS
*SW10=1 BELL ON ERROR
*SW09=1 LOOP ON ERROR
*CALL
* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
$ERROR:
7$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;;SET THE ERROR FLAG
BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG

```

```

7727 047614 032777 002000 131316          BIT      #BIT10,ASWR      ;; BELL ON ERROR?
7728 047622 001402                    BEQ      1$           ;; NO - SKIP
7729 047624 104401 001204                    TYPE    $SBELL        ;; RING BELL
7730 047630 005237 001112          1$:      INC      $ERTTL    ;; COUNT THE NUMBER OF ERRORS
7731 047634 011637 001116          MOV      (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
7732 047640 162737 000002 001116          SUB      #2, $ERRPC
7733 047646 117737 131244 001114          MOVB    $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
7734 047654 032777 020000 131256          BIT      #BIT13,ASWR  ;; SKIP TYPEOUT IF SET
7735 047662 001004                    BNE     20$         ;; SKIP TYPEOUTS
7736 047664 004737 047776          JSR     PC, TYPERR  ;; GO TO USER ERROR ROUTINE
7737 047670 104401 001211          TYPE    , $CRLF
7738 047674                    20$:
7739 047674 122737 000001 001234          CMPB    #APTENV, $ENV ;; RUNNING IN APT MODE
7740 047702 001007                    BNE     2$           ;; NO, SKIP APT ERROR REPORT
7741 047704 113737 001114 047716          MOVB    $ITEMB, 21$  ;; SET ITEM NUMBER AS ERROR NUMBER
7742 047712 004737 047346          JSR     PC, $ATY4   ;; REPORT FATAL ERROR TO APT
7743 047716          000                    21$:      .BYTE    0
7744 047717          000                    .BYTE    0
7745 047720 000777                    BR      22$         ;; APT ERROR LOOP
7746 047722 005777 131212          2$:      TST     $SWR        ;; HALT ON ERROR
7747 047726 100002                    BPL     3$           ;; SKIP IF CONTINUE
7748 047730 000000          HALT
7749 047732 104407          CKSWR  ;; HALT ON ERROR!
7750 047734 032777 001000 131176 3$:      BIT      #BIT09,ASWR  ;; TEST FOR CHANGE IN SOFT-SWR
7751 047742 001402                    BEQ     4$           ;; LOOP ON ERROR SWITCH SET?
7752 047744 013716 001110                    MOV     $LPERR, (SP) ;; BR IF NO
7753 047750 005737 001202          4$:      TST     $ESCAPE    ;; FUDGE RETURN FOR LOOPING
7754 047754 001402                    BEQ     5$           ;; CHECK FOR AN ESCAPE ADDRESS
7755 047756 013716 001202          MOV     $ESCAPE, (SP) ;; BR IF NONE
7756 047762          5$:
7757 047762 022737 044674 000042          CMP     #SENDAD, @#42 ;; FUDGE RETURN ADDRESS FOR ESCAPE
7758 047770 001001                    BNE     6$           ;; ACT-11 AUTO-ACCEPT?
7759 047772 000000                    HALT    ;; BRANCH IF NO
7760 047774          6$:
7761 047774 000002          RTI      ;; YES
7762
7763          ;; *****
7764          .SBTTL TYPE ERROR ROUTINE
7765          ;*ENTRY JSR PC, TYPERR
7766          ;*RETURN RTS PC
7767          ;*
7768          ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
7769          ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
7770          ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
7771          ;*THE ERROR.
7772          ;; *****
7773 047776 104413          TYPERR: SAVREG
7774 050000 113700 001114          MOVB    $ITEMB, RO  ;; ENTER ERROR NUMBER
7775 050004 042700 177400          BIC     #177400, RO  ;; CLEAR UNUSED BITS
7776 050010 005300          DEC     RO          ;; FORM INDEX FOR ERROR TABLE
7777 050012 006300          ASL     RO
7778 050014 006300          ASL     RO
7779 050016 006300          ASL     RO
7780 050020 062700 001300 1$:      ADD     #ERRTB, RO   ;; FORM ADDRESS OF ERROR ENTRY
7781 050024 012037 050040          MOV     (RO)+, 2$  ;; GET EM POINTER
7782 050030 001404          BEQ     3$         ;; BRANCH IF THERE ISN'T ONE

```

7783	050032	104401	001211		TYPE	, \$CRLF	; TYPE CARRIAGE RETURN LINE FEED
7784	050036	104401			TYPE		; TYPE ERROR MESSAGE (EM)
7785	050040	000000		2\$:	.WORD	0	; EM POINTER GOES HERE
7786	050042	012037	050056	3\$:	MOV	(R0)+, 4\$; GET DH POINTER
7787	050046	001404			BEQ	5\$; BRANCH IF THERE ISN'T ONE
7788	050050	104401	001211		TYPE	, \$CRLF	; TYPE CR-LF
7789	050054	104401			TYPE		; TYPE DATA HEADER
7790	050056	000000		4\$:	.WORD	0	; DH POINTER GOES HERE
7791	050060	012001		5\$:	MOV	(R0)+, R1	; GET DT POINTER
7792	050062	001445			BEQ	20\$; BRANCH IF THERE ARE NONE
7793	050064	005004			CLR	R4	; RESET INDENT SWITCH
7794	050066	012000			MOV	(R0)+, R0	; GET DF POINTER
7795	050070	012002			MOV	(R0)+, R2	; STORE NUMBER OF DH'S
7796	050072	104401	001211		TYPE	, \$CRLF	
7797	050076	112003		10\$:	MOVB	(R0)+, R3	; GET & STORE NUMBER OF DATA WORDS
7798	050100	105720			TSTB	(R0)+	; BUMP PAST FORMAT WORD
7799	050102	005703			TST	R3	; TEST IF ANY DATA FOR THIS HEADER
7800	050104	001416			BEQ	14\$; NO - SKIP DATA PRINT
7801	050106	005704			TST	R4	; CHECK IF INDENT WORDS
7802	050110	001004			BNE	12\$; YES, GO INDENT
7803	050112	013146		11\$:	MOV	2(R1)+, -(SP)	; PUT FIRST DATA WORD ON STACK
7804	050114	104402			TYPOC		; TYPE IT
7805	050116	005303			DEC	R3	; MORE DATA WORDS
7806	050120	001403			BEQ	13\$; NO-BRANCH
7807	050122	104401	054723	12\$:	TYPE	, SPACE2	; TYPE SEPARATORS
7808	050126	000771			BR	11\$; LOOP
7809	050130	104401	001211	13\$:	TYPE	, \$CRLF	; TYPE <CR><LF>
7810	050134	005710			TST	(R0)	; CHECK IF NEXT HEADER AVAILABLE
7811	050136	001401			BEQ	14\$; NO, DO NOT CHANGE INDENT
7812	050140	005104			COM	R4	; CHANGE INDENT
7813	050142	005302		14\$:	DEC	R2	; MORE DH'S?
7814	050144	003414			BLE	20\$; NO-BRANCH
7815	050146	012037	050166	15\$:	MOV	(R0)+, 18\$; GET NEXT DH POINTER
7816	050152	001751			BEQ	10\$; IF NO HEADER GET DATA
7817	050154	005704			TST	R4	; INDENT?
7818	050156	001402			BEQ	17\$; NO-BRANCH
7819	050160	104401	054723		TYPE	, SPACE2	; INDENT
7820	050164	104401		17\$:	TYPE		; TYPE DH
7821	050166	000000		18\$:	.WORD	0	; DH POINTER GOES HERE
7822	050170	104401	001211		TYPE	, \$CRLF	
7823	050174	000740			BR	10\$; LOOP
7824	050176	104414		20\$:	RESREG		
7825	050200	005237	003610		INC	ERRCNT	; INCREMENT ERROR COUNT
7826	050204	032777	010000 130726		BIT	#SW12, 2SWR	; CHECK IF ABORT AFTER 20 ERRORS
7827	050212	001421			BEQ	25\$; NO, RETURN
7828	050214	022737	000024 003610		CMP	#20., ERRCNT	; CHECK IF EROR THRESHOLD EXCEEDED
7829	050222	103015			BHIS	25\$; NO, RETURN
7830	050224	104401	054726		TYPE	, ABORT	; TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
7831							; ERROR THRESHOLD EXCEEDED"
7832	050230	005737	000042		TST	42	; CHECK IF IN CHAIN MODE
7833	050234	001407			BEQ	30\$; NO, HALT
7834	050236	012737	000001 044532		MOV	#1, \$EOPCT	; FORCE END OF PASS COUNT TO ONE FOR ABORT
7835	050244	012706	001100		MOV	#STACK, SP	; INITIALIZE STACK
7836	050250	000137	044504		JMP	\$EOP	; BRING IN NEXT PROGRAM IN CHAIN
7837	050254	000000		30\$:	HALT		
7838	050256	000207		25\$:	RTS	PC	

```

7839
7840
7841 050260 013702 001270
7842 050264 012762 100000
7843 050272 012706 001100
7844 050276 104401 054657
7845 050302 005737 000042
7846 050306 001410
7847 050310 105037 001103
7848 050314 005037 001202
7849 050320 005037 044532
7850 050324 000137 044504
7851
7852 050330 000000
7853 050332 000137 003666
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872 050336 105737 001157
7873 050342 100002
7874 050344 000000
7875 050346 000430
7876 050350 010046
7877 050352 017600 000002
7878 050356 122737 000001 001234
7879 050364 001011
7880 050366 132737 000100 001235
7881 050374 001405
7882 050376 010037 050406
7883 050402 004737 047336
7884 050406 000000
7885 050410 132737 000040 001235
7886 050416 001003
7887 050420 112046
7888 050422 001005
7889 050424 005726
7890 050426 012600
7891 050430 062716 000002
7892 050434 000002
7893 050436 122716 000011
7894 050442 001430

```

```

.SBTTL CONTROLLED PROGRAM HALT
CTRHLT: MOV $BASE,R2
MOV #CCLR,RKCS1(R2)
MOV #STACK,SP
TYPE OPROO7
TST 42
BEQ 5$
CLRB $ERFLG
CLR $ESCAPE
CLR $EOPCT
JMP $EOP

```

```

;SET '611 BASE
;CLEAR CONTROLLER
;CLEAR STACK
;TYPE HALT MESSAGE
;TEST IF MONITOR PRESENT
;NO - SKIP
;CLEAR ERROR FLAG
;CLEAR ESCAPE
;SET PASS COUNT TO 0
;GO TO END OF PASS

```

```

5$: HALT
JMP START1

```

```

;HALT PROGRAM
;DO RESTART IF CONTINUE

```

```

.SBTTL TYPE ROUTINE

```

```

;*****
;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

```

```

*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR
*OR
* TYPE
* MESADR
*

```

```

;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING

```

```

$TYPE: TSTB $TFPLG
BPL 1$
HALT
BR 3$
1$: MOV RO,-(SP)
MOV @2(SP),RO
CMPB #APTENV,$ENV
BNE 62$
BITB #APTPOOL,$ENVM
BEQ 62$
MOV RO,61$
JSR PC,$ATY3
61$: .WORD 0
62$: BITB #APTCSUP,$ENVM
BNE 60$
2$: MOVB (RO)+,-(SP)
BNE 4$
TST (SP)+
60$: MOV (SP)+,RO
3$: ADD #2,(SP)
RTI
4$: CMPB #HT,(SP)
BEQ 8$

```

```

;IS THERE A TERMINAL?
;BR IF YES
;HALT HERE IF NO TERMINAL
;LEAVE
;SAVE RO
;GET ADDRESS OF ASCIZ STRING
;RUNNING IN APT MODE
;NO GO CHECK FOR APT CONSOLE
;SPOOL MESSAGE TO APT
;NO GO CHECK FOR CONSOLE
;SETUP MESSAGE ADDRESS FOR APT
;SPOOL MESSAGE TO APT
;MESSAGE ADDRESS
;APT CONSOLE SUPPRESSED
;YES SKIP TYPE OUT
;PUSH CHARACTER TO BE TYPED ONTO STACK
;BR IF IT ISN'T THE TERMINATOR
;IF TERMINATOR POP IT OFF THE STACK
;RESTORE RO
;ADJUST RETURN PC
;RETURN
;BRANCH IF <HT>

```

```

7895 050444 122716 000200          CMPB   #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
7896 050450 001006                   BNE    5$              ;;
7897 050452 005726                   TST    (SP)+           ;;POP <CR><LF> EQUIV
7898 050454 104401                   TYPE   ;;TYPE A CR AND LF
7899 050456 001211                   $CRLF
7900 050460 105037 050614          CLRB   $CHARCNT       ;;CLEAR CHARACTER COUNT
7901 050464 000755                   BR     2$              ;;GET NEXT CHARACTER
7902 050466 004737 050550          5$:   JSR    PC,$TYPEC  ;;GO TYPE THIS CHARACTER
7903 050472 123726 001156          6$:   CMPB   $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
7904 050476 001350                   BNE    2$              ;;IF NO GO GET NEXT CHAR.
7905 050500 013746 001154          MOV    $NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
7906                                     AND    THE NULL CHAR.
7907 050504 105366 000001          7$:   DEC    1(SP)      ;;DOES A NULL NEED TO BE TYPED?
7908 050510 002770                   BLT    6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
7909 050512 004737 050550          JSR    PC,$TYPEC  ;;GO TYPE A NULL
7910 050516 105337 050614          DEC    $CHARCNT       ;;DO NOT COUNT AS A COUNT
7911 050522 000770                   BR     7$              ;;LOOP
7912
7913                                     ;HORIZONTAL TAB PROCESSOR
7914
7915 050524 112716 000040          8$:   MOV    #' (SP)    ;;REPLACE TAB WITH SPACE
7916 050530 004737 050550          9$:   JSR    PC,$TYPEC  ;;TYPE A SPACE
7917 050534 132737 000007 050614  BIT    #7,$CHARCNT    ;;BRANCH IF NOT AT
7918 050542 001372                   BNE    9$              ;;TAB STOP
7919 050544 005726                   TST    (SP)+           ;;POP SPACE OFF STACK
7920 050546 000724                   BR     2$              ;;GET NEXT CHARACTER
7921 050550 105777 130374          $TYPEC: TST    $STPS     ;;WAIT UNTIL PRINTER IS READY
7922 050554 100375                   BPL    $TYPEC
7923 050556 116677 000002 130366  MOV    2(SP),2$TPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7924 050564 122766 000015 000002  CMPB   #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
7925 050572 001003                   BNE    1$              ;;BRANCH IF NO
7926 050574 105037 050614          CLRB   $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
7927 050600 000406                   BR     $TYPEX          ;;EXIT
7928 050602 122766 000012 000002  1$:   CMPB   #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
7929 050610 001402                   BEQ    $TYPEX          ;;BRANCH IF YES
7930 050612 105227                   INCB   (PC)+           ;;COUNT THE CHARACTER
7931 050614 000000          $CHARCNT: .WORD    0   ;;CHARACTER COUNT STORAGE
7932 050616 000207          $TYPEX:  RTS    PC
7933
7934                                     .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
7935
7936                                     ;*****
7937                                     ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
7938                                     ;OCTAL (ASCII) NUMBER AND TYPE IT.
7939                                     ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7940                                     ;CALL:
7941                                     ;      MOV    NUM,-(SP)    ;;NUMBER TO BE TYPED
7942                                     ;      TYPOS  ;;CALL FOR TYPEOUT
7943                                     ;      .BYTE  N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
7944                                     ;      .BYTE  M      ;;M=1 OR 0
7945                                     ;      ;;1=TYPE LEADING ZEROS
7946                                     ;      ;;0=SUPPRESS LEADING ZEROS
7947                                     ;
7948                                     ;$STYON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7949                                     ;$TYPOS OR $TYPC
7950                                     ;CALL:

```

```

7951      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7952      *      TYPON      ;;CALL FOR TYPEOUT
7953      *
7954      *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7955      *CALL:
7956      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7957      *      TYPOC      ;;CALL FOR TYPEOUT
7958
7959 050620 017646 000000      $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
7960 050624 116637 000001 051043  MOVB      1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
7961 050632 112637 051045      MOVB      (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
7962 050636 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
7963 050642 000406      BR      $TYPON
7964 050644 112737 000001 051043  $TYPOC: MOVB      #1,$OFILL      ;;SET THE ZERO FILL SWITCH
7965 050652 112737 000006 051045      MOVB      #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
7966 050660 112737 000005 051042  $TYPON: MOVB      #5,$OCNT      ;;SET THE ITERATION COUNT
7967 050666 010346      MOV      R3,-(SP)      ;;SAVE R3
7968 050670 010446      MOV      R4,-(SP)      ;;SAVE R4
7969 050672 010546      MOV      R5,-(SP)      ;;SAVE R5
7970 050674 113704 051045      MOVB      $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
7971 050700 005404      NEG      R4
7972 050702 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
7973 050706 110437 051044      MOVB      R4,$OMODE      ;;SAVE IT FOR USE
7974 050712 113704 051043      MOVB      $OFILL,R4      ;;GET THE ZERO FILL SWITCH
7975 050716 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
7976 050722 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
7977 050724 006105      1$: ROL      R5      ;;ROTATE MSB INTO "C"
7978 050726 000404      BR      3$      ;;GO DO MSB
7979 050730 006105      2$: ROL      R5      ;;FORM THIS DIGIT
7980 050732 006105      ROL      R5
7981 050734 006105      ROL      R5
7982 050736 010503      MOV      R5,R3
7983 050740 006103      3$: ROL      R3      ;;GET LSB OF THIS DIGIT
7984 050742 105337 051044      DECB      $OMODE      ;;TYPE THIS DIGIT?
7985 050746 100016      BPL      7$      ;;BR IF NO
7986 050750 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
7987 050754 001002      BNE      4$      ;;TEST FOR 0
7988 050756 005704      TST      R4      ;;SUPPRESS THIS 0?
7989 050760 001403      BEQ      5$      ;;BR IF YES
7990 050762 005204      4$: INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
7991 050764 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
7992 050770 052703 000040      5$: BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
7993 050774 110337 051040      MOVB      R3,#$      ;;SAVE FOR TYPING
7994 051000 104401 051040      TYPE      #8$      ;;GO TYPE THIS DIGIT
7995 051004 105337 051042      7$: DECB      $OCNT      ;;COUNT BY 1
7996 051010 003347      BGT      2$      ;;BR IF MORE TO DO
7997 051012 002402      BLT      6$      ;;BR IF DONE
7998 051014 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
7999 051016 000744      BR      2$      ;;GO DO THE LAST DIGIT
8000 051020 012605      6$: MOV      (SP)+,R5      ;;RESTORE R5
8001 051022 012604      MOV      (SP)+,R4      ;;RESTORE R4
8002 051024 012603      MOV      (SP)+,R3      ;;RESTORE R3
8003 051026 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
8004 051034 012616      MOV      (SP)+,(SP)
8005 051036 000002      RTI      ;;RETURN
8006 051040 000      8$: .BYTE      0      ;;STORAGE FOR ASCII DIGIT
    
```

```

8007 051041 000
8008 051042 000
8009 051043 000
8010 051044 000000
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023 051046
8024 051046 010046
8025 051050 010146
8026 051052 010246
8027 051054 010346
8028 051056 010546
8029 051060 012746 020200
8030 051064 016605 000020
8031 051070 100004
8032 051072 005405
8033 051074 112766 000055 000001
8034 051102 005000 1$:
8035 051104 012703 051262
8036 051110 112723 000040
8037 051114 005002 2$:
8038 051116 016001 051252
8039 051122 160105 3$:
8040 051124 002402
8041 051126 005202
8042 051130 000774
8043 051132 060105 4$:
8044 051134 005702
8045 051136 001002
8046 051140 105716
8047 051142 100407
8048 051144 106316 5$:
8049 051146 103003
8050 051150 116663 000001 177777
8051 051156 052702 000060 6$:
8052 051162 052702 000040 7$:
8053 051166 110223
8054 051170 005720
8055 051172 020027 000010
8056 051176 002746
8057 051200 003002
8058 051202 010502
8059 051204 000764
8060 051206 105726 8$:
8061 051210 100003
8062 051212 116663 177777 177776

```

```

      .BYTE 0          ;; TERMINATOR FOR TYPE ROUTINE
SOCNT: .BYTE 0        ;; OCTAL DIGIT COUNTER
$OFILL: .BYTE 0       ;; ZERO FILL SWITCH
$OMODE: .WORD 0        ;; NUMBER OF DIGITS TO TYPE
.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

$TYPDS:
MOV      R0,-(SP)          ;; PUSH R0 ON STACK
MOV      R1,-(SP)          ;; PUSH R1 ON STACK
MOV      R2,-(SP)          ;; PUSH R2 ON STACK
MOV      R3,-(SP)          ;; PUSH R3 ON STACK
MOV      R5,-(SP)          ;; PUSH R5 ON STACK
MOV      #20200,-(SP)      ;; SET BLANK SWITCH AND SIGN
MOV      20(SP),R5         ;; GET THE INPUT NUMBER
BPL      1$                ;; BR IF INPUT IS POS.
NEG      R5                ;; MAKE THE BINARY NUMBER POS.
MOVB    #'-,1(SP)         ;; MAKE THE ASCII NUMBER NEG.
CLR      R0                ;; ZERO THE CONSTANTS INDEX
MOV      #SDBLK,R3        ;; SETUP THE OUTPUT POINTER
MOVB    #' ,(R3)+         ;; SET THE FIRST CHARACTER TO A BLANK
2$:     CLR      R2        ;; CLEAR THE BCD NUMBER
MOV      $DTBL(R0),R1     ;; GET THE CONSTANT
3$:     SUB      R1,R5     ;; FORM THIS BCD DIGIT
BLT     4$                ;; BR IF DONE
INC     R2                ;; INCREASE THE BCD DIGIT BY 1
BR      3$
4$:     ADD      R1,R5     ;; ADD BACK THE CONSTANT
TST     R2                ;; CHECK IF BCD DIGIT=0
BNE     5$                ;; FALL THROUGH IF 0
TSTB   (SP)              ;; STILL DOING LEADING 0'S?
BMI     7$                ;; BR IF YES
5$:     ASLB    (SP)       ;; MSD?
BCC     6$                ;; BR IF NO
MOVB    1(SP),-1(R3)      ;; YES--SET THE SIGN
BIS     #'0,R2            ;; MAKE THE BCD DIGIT ASCII
7$:     BIS     #' ,R2     ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB    R2,(R3)+         ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST     (R0)+            ;; JUST INCREMENTING
CMP     R0,#10           ;; CHECK THE TABLE INDEX
BLT     2$                ;; GO DO THE NEXT DIGIT
BGT     8$                ;; GO TO EXIT
MOV     R5,R2            ;; GET THE LSD
BR      6$                ;; GO CHANGE TO ASCII
8$:     TSTB   (SP)+       ;; WAS THE LSD THE FIRST NON-ZERO?
BPL     9$                ;; BR IF NO
MOVB    -1(SP),-2(R3)    ;; YES--SET THE SIGN FOR TYPING

```

```

8063 051220 105013          9$:   CLRB   (R3)           ;; SET THE TERMINATOR
8064 051222 012605          MOV   (SP)+,R5         ;; POP STACK INTO R5
8065 051224 012603          MOV   (SP)+,R3         ;; POP STACK INTO R3
8066 051226 012602          MOV   (SP)+,R2         ;; POP STACK INTO R2
8067 051230 012601          MOV   (SP)+,R1         ;; POP STACK INTO R1
8068 051232 012600          MOV   (SP)+,R0         ;; POP STACK INTO R0
8069 051234 104401 051262  TYPE   $DBLK          ;; NOW TYPE THE NUMBER
8070 051240 016666 000002 000004  MOV   2(SP),4(SP)      ;; ADJUST THE STACK
8071 051246 012616          MOV   (SP)+,(SP)
8072 051250 000002          RTI                    ;; RETURN TO USER
8073 051252 023420          $DTBL: 10000.
8074 051254 001750          1000.
8075 051256 000144          100.
8076 051260 000012          10.
8077 051262 000004          $DBLK: .BLKW 4
8078                                     .SBTTL TTY INPUT ROUTINE
8079
8080                                     ;:*****
8081                                     .ENABL  LSB
8082 051272 000000          $TKCNT: .WORD 0       ;; NUMBER OF ITEMS IN QUEUE
8083 051274 000000          $TKQIN: .WORD 0       ;; INPUT POINTER
8084 051276 000000          $TKQOUT: .WORD 0      ;; OUTPUT POINTER
8085 051300 000001          $TKQSRV: .BLKB 1     ;; TTY KEYBOARD QUEUE
8086                                     $TKQEND=.
8087                                     .EVEN
8088
8089                                     ;*TK INITIALIZE ROUTINE
8090                                     ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
8091                                     ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
8092
8093                                     ;*CALL:
8094                                     ;*   JSR   PC,$TKINT
8095                                     ;*   RETURN
8096
8097 051302 005037 051272          $TKINT: CLR   $TKCNT      ;; CLEAR COUNT OF ITEMS IN QUEUE
8098 051306 012737 051300 051274  MOV   #$TKQSRV,$TKQIN  ;; MOVE THE STARTING ADDRESS OF THE
8099 051314 013737 051274 051276  MOV   $TKQIN,$TKQOUT   ;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
8100 051322 012737 051352 000060  MOV   #$TKSRV,$TKVEC  ;; INITIALIZE THE KEYBOARD VECTOR
8101 051330 012737 000200 000062  MOV   #200,$TKVEC+2   ;; "BR" LEVEL 4
8102 051336 005777 127604          TST   $TKB            ;; CLEAR DONE FLAG
8103 051342 012777 000100 127574  MOV   #100,$TKS       ;; ENABLE TTY KEYBOARD INTERRUPT
8104 051350 000207          RTS   PC              ;; RETURN TO CALLER
8105
8106                                     ;*TK SERVICE ROUTINE
8107                                     ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
8108                                     ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
8109                                     ;*IT IN THE QUEUE.
8110                                     ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
8111                                     ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
8112
8113 051352 117746 127570          $TKSRV: MOVB  $TKB,-(SP) ;; PICKUP THE CHARACTER
8114 051356 042716 177600          BIC   #↑C177,(SP)    ;; STRIP THE JUNK
8115 051362 021627 000003          CMP   (SP),#3        ;; IS IT A CONTROL C?
8116 051366 001007          BNE   1$              ;; BRANCH IF NO
8117 051370 104401 052466          TYPE  $CNTLC          ;; TYPE A CONTROL-C (↑C)
8118 051374 004737 051302          JSR   PC,$TKINT      ;; INIT THE KEYBOARD

```

```

8119 051400 005726          TST      (SP)+          ;; CLEAN UP STACK
8120 051402 000137 050260    JMP      CTRHLT        ;; CONTROL C RESTART
8121 051406 021627 000007    1$:    CMP      (SP),#7     ;; IS IT A CONTROL G?
8122 051412 001004          BNE     2$            ;; BRANCH IF NO
8123 051414 022737 000176 001140    CMP      #SWREG,SWR    ;; IS SOFT-SWR SELECTED?
8124 051422 001500          BEQ     6$            ;; GO TO SWR CHANGE
8125
8126 051424          2$:
8127 051424 022737 000001 051272    CMP      #1,$STKCNT   ;; IS THE QUEUE FULL?
8128 051432 001004          BNE     3$            ;; BRANCH IF NO
8129 051434 104401 001204    TYPE    $BELL         ;; RING THE TTY BELL
8130 051440 005726          TST     (SP)+         ;; CLEAN CHARACTER OFF OF STACK
8131 051442 000451          BR      5$            ;; EXIT
8132 051444 021627 000023    3$:    CMP      (SP),#23     ;; IS IT A CONTROL-S?
8133 051450 001021          BNE     32$           ;; BRANCH IF NO
8134 051452 005077 127466    CLR     @STKS         ;; DISABLE TTY KEYBOARD INTERRUPTS
8135 051456 005726          TST     (SP)+         ;; CLEAN CHAR OFF STACK
8136 051460 105777 127460    31$:   TSTB   @STKS         ;; WAIT FOR A CHAR
8137 051464 100375          BPL     31$          ;; LOOP UNTIL ITS THERE
8138 051466 117746 127454    MOVB   @STKB,-(SP)   ;; GET THE CHARACTER
8139 051472 042716 177600    BIC    #177,(SP)    ;; MAKE IT 7-BIT ASCII
8140 051476 022627 000021    CMP     (SP),#21    ;; IS IT A CONTROL-Q?
8141 051502 001366          BNE     31$          ;; BRANCH IF NO
8142 051504 012777 000100 127432    MOV     #100,@STKS   ;; REENABLE TTY KEYBOARD INTERRUPTS
8143 051512 000002          RTI                    ;; RETURN
8144 051514 005237 051272    32$:   INC     $STKCNT      ;; COUNT THIS CHARACTER
8145 051520 021627 000140    CMP     (SP),#140   ;; IS IT UPPER CASE?
8146 051524 002405          BLT     4$            ;; BRANCH IF YES
8147 051526 021627 000175    CMP     (SP),#175   ;; IS IT A SPECIAL CHAR?
8148 051532 003002          BGT     4$            ;; BRANCH IF YES
8149 051534 042716 000040    BIC    #40,(SP)    ;; MAKE IT UPPER CASE
8150 051540 112677 177530    4$:   MOVB   (SP)+,@STKQIN ;; AND PUT IT IN QUEUE
8151 051544 005237 051274    INC     $STKQIN     ;; UPDATE THE POINTER
8152 051550 023727 051274 051301    CMP     $STKQIN,$STKQEND ;; GO OFF THE END?
8153 051556 001003          BNE     5$            ;; BRANCH IF NO
8154 051560 012737 051300 051274    MOV     #$STKQSRST,$STKQIN ;; RESET THE POINTER
8155 051566 000002          RTI                    ;; RETURN

```

8156 *****

```

8157 ;; *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
8158 ;; *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
8159 ;; *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
8160 ;; *CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```

```

8161
8162 051570 022737 000176 001140 $CKSWR: CMP     #SWREG,SWR    ;; IS THE SOFT-SWR SELECTED
8163 051576 001124          BNE     15$          ;; EXIT IF NOT
8164 051600 105777 127340    TSTB   @STKS         ;; IS A CHAR WAITING?
8165 051604 100121          BPL     15$          ;; IF NOT, EXIT
8166 051606 117746 127334    MOVB   @STKB,-(SP)   ;; YES
8167 051612 042716 177600    BIC    #177,(SP)    ;; MAKE IT 7-BIT ASCII
8168 051616 021627 000007    CMP     (SP),#7     ;; IS IT A CONTROL-G?
8169 051622 001300          BNE     2$            ;; IF NOT, PUT IT IN THE TTY QUEUE
8170          AND    EXIT

```

```

8171 *****
8172 ;; *CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
8173 ;; *ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
8174

```

```

8175      ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
8176 051624 123727 001134 000001 6$: CMPB $AUTOB,#1      ;; ARE WE RUNNING IN AUTO-MODE?
8177 051632 001674      BEQ 2$              ;; BRANCH IF YES
8178 051634 005726      TST (SP)+          ;; CLEAR CONTROL-G OFF STACK
8179 051636 004737 051302 JSR PC,$TKINT      ;; FLUSH THE TTY INPUT QUEUE
8180 051642 005077 127276 CLR $STKS          ;; DISABLE TTY KEYBOARD INTERRUPTS
8181 051646 112737 000001 001135 MOVB #1,$INTAG    ;; SET INTERRUPT MODE INDICATOR
8182
8183 051654 104401 052500      TYPE , $CNTLG      ;; ECHO THE CONTROL-G (↑G)
8184 051660 104401 052505 $GTSWR: TYPE , $MSWR  ;; TYPE CURRENT CONTENTS
8185 051664 013746 000176      MOV SWREG,-(SP)    ;; SAVE SWREG FOR TYPEOUT
8186 051670 104402      TYPEOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
8187 051672 104401 052516      TYPE , $MNEW      ;; PROMPT FOR NEW SWR
8188 051676 005046      19$: CLR -(SP)        ;; CLEAR COUNTER
8189 051700 005046      CLR -(SP)        ;; THE NEW SWR
8190 051702 105777 127236      7$: TSTB $STKS      ;; CHAR THERE?
8191 051706 100375      BPL 7$              ;; IF NOT TRY AGAIN
8192
8193 051710 117746 127232      MOVB $STKB,-(SP)   ;; PICK UP CHAR
8194 051714 042716 177600      BIC #↑C177,(SP)  ;; MAKE IT 7-BIT ASCII
8195
8196 051720 021627 000003      CMP (SP),#3        ;; IS IT A CONTROL-C?
8197 051724 001015      BNE 9$            ;; BRANCH IF NOT
8198 051726 104401 052466      TYPE , $CNTLC     ;; YES, ECHO CONTROL-C (↑C)
8199 051732 062706 000006      ADD #6,SP         ;; CLEAN UP STACK
8200 051736 123727 001135 000001 CMPB $INTAG,#1    ;; REENABLE TTY KEYBOARD INTERRUPTS?
8201 051744 001003      BNE 8$            ;; BRANCH IF NO
8202 051746 012777 000100 127170 MOV #100,$STKS    ;; ALLOW TTY KEYBOARD INTERRUPTS
8203 051754 000137 050260      8$: JMP CTRHLT    ;; CONTROL-C RESTART
8204
8205
8206 051760 021627 000025      9$: CMP (SP),#25     ;; IS IT A CONTROL-U?
8207 051764 001005      BNE 10$          ;; BRANCH IF NOT
8208 051766 104401 052473      TYPE , $CNTLU    ;; YES, ECHO CONTROL-U (↑U)
8209 051772 062706 000006      20$: ADD #6,SP    ;; IGNORE PREVIOUS INPUT
8210 051776 000737      BR 19$              ;; LET'S TRY IT AGAIN
8211
8212
8213 052000 021627 000015      10$: CMP (SP),#15   ;; IS IT A <CR>?
8214 052004 001022      BNE 16$          ;; BRANCH IF NO
8215 052006 005766 000004      TST 4(SP)        ;; YES, IS IT THE FIRST CHAR?
8216 052012 001403      BEQ 11$            ;; BRANCH IF YES
8217 052014 016677 000002 127116 MOV 2(SP),$SWR    ;; SAVE NEW SWR
8218 052022 062706 000006      11$: ADD #6,SP    ;; CLEAR UP STACK
8219 052026 104401 001211      14$: TYPE , $CRLF  ;; ECHO <CR> AND <LF>
8220 052032 123727 001135 000001 CMPB $INTAG,#1    ;; RE-ENABLE TTY KBD INTERRUPTS?
8221 052040 001003      BNE 15$            ;; BRANCH IF NOT
8222 052042 012777 000100 127074 MOV #100,$STKS    ;; RE-ENABLE TTY KBD INTERRUPTS
8223 052050 000002      15$: RTI          ;; RETURN
8224 052052 004737 050550      16$: JSR PC,$TYPEC  ;; ECHO CHAR
8225 052056 021627 000060      CMP (SP),#60     ;; CHAR < 0?
8226 052062 002420      BLT 18$          ;; BRANCH IF YES
8227 052064 021627 000067      CMP (SP),#67     ;; CHAR > 7?
8228 052070 003015      BGT 18$          ;; BRANCH IF YES
8229 052072 042726 000060      BIC #60,(SP)+   ;; STRIP-OFF ASCII
8230 052076 005766 000002      TST 2(SP)        ;; IS THIS THE FIRST CHAR

```

```

8231 052102 001403          BEQ      17$          ;; BRANCH IF YES
8232 052104 006316          ASL      (SP)         ;; NO, SHIFT PRESENT
8233 052106 006316          ASL      (SP)         ;; CHAR OVER TO MAKE
8234 052110 006316          ASL      (SP)         ;; ROOM FOR NEW ONE.
8235 052112 005266 000002 17$: INC      2(SP)         ;; KEEP COUNT OF CHAR
8236 052116 056616 177776  BIS      -2(SP), (SP) ;; SET IN NEW CHAR
8237 052122 000667          BR       7$           ;; GET THE NEXT ONE
8238 052124 104401 001210 18$: TYPE   $QUES      ;; TYPE ?<CR><LF>
8239 052130 000720          BR       20$         ;; SIMULATE CONTROL-U
8240          .DSABL  LSB
8241
8242
8243          ;; *****
8244          ;; THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8245          ;; CALL:
8246          ;; RDCHR          ;; GET A CHARACTER FROM THE QUEUE
8247          ;; RETURN HERE  ;; CHARACTER IS ON THE STACK
8248          ;;              ;; WITH PARITY BIT STRIPPED OFF
8249          ;;
8250
8251 052132 011646          $RDCHR: MOV     (SP), -(SP) ;; PUSH DOWN THE PC AND
8252 052134 016666 000004 000002 MOV     4(SP), 2(SP) ;; THE PS
8253 052142 005066 000004 CLR     4(SP)         ;; GET READY FOR A CHARACTER
8254 052146 005046          CLR     -(SP)        ;; PUT NEW PS ON STACK
8255 052150 012746 052156 MOV     #64$, -(SP)  ;; PUT NEW PC ON STACK
8256 052154 000002          RTI          ;; POP NEW PC AND PS
8257 052156          64$:
8258 052156 005737 051272 1$: TST     $TKCNT      ;; WAIT ON A CHARACTER
8259 052162 001775          BEQ     1$           ;;
8260 052164 005337 051272 DEC     $TKCNT      ;; DECREMENT THE COUNTER
8261 052170 117766 177102 000004 MOVB   2($TKQOUT, 4(SP) ;; GET ONE CHARACTER
8262 052176 005237 051276 INC     $TKQOUT      ;; UPDATE THE POINTER
8263 052202 023727 051276 051301 CMP    $TKQOUT, #($TKQEND) ;; DID IT GO OFF OF THE END?
8264 052210 001003          BNE    2$           ;; BRANCH IF NO
8265 052212 012737 051300 051276 MOV    #($TKQSRT, $TKQOUT) ;; RESET THE POINTER
8266 052220 000002          RTI          ;; RETURN
8267          ;; *****
8268          ;; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
8269          ;; CALL:
8270          ;; RDLIN          ;; INPUT A STRING FROM THE TTY
8271          ;; RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8272          ;;              ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
8273          ;;
8274 052222 010346          $RDLIN: MOV    R3, -(SP) ;; SAVE R3
8275 052224 005046          CLR    -(SP)        ;; CLEAR THE RUBOUT KEY
8276 052226 012703 052456 MOV    #($TTYIN, R3) ;; GET ADDRESS
8277 052232 022703 052456 2$: CMP    #($TTYIN+8., R3) ;; BUFFER FULL?
8278 052236 101456          BLOS   4$           ;; BR IF YES
8279 052240 104410          RDCHR  ;; GO READ ONE CHARACTER FROM THE TTY
8280 052242 112613          MOVB  (SP)+, (R3)    ;; GET CHARACTER
8281 052244 122713 000177 10$: CMPB  #177, (R3)   ;; IS IT A RUBOUT
8282 052250 001022          BNE   5$           ;; BR IF NO
8283 052252 005716          TST   (SP)         ;; IS THIS THE FIRST RUBOUT?
8284 052254 001007          BNE   6$           ;; BR IF NO
8285 052256 112737 000134 052454 MOVB  #'\, 9$      ;; TYPE A BACK SLASH
8286 052264 104401 052454 TYPE   , 9$

```

```

8287 052270 012716 177777      MOV      #-1,(SP)      ;; SET THE RUBOUT KEY
8288 052274 005303      DEC      R3          ;; BACKUP BY ONE
8289 052276 020327 052456      CMP      R3,#$TTYIN  ;; STACK EMPTY?
8290 052302 103434      BLO     4$          ;; BR IF YES
8291 052304 111337 052454      MOVVB   (R3),9$     ;; SETUP TO TYPEOUT THE DELETED CHAR.
8292 052310 104401 052454      TYPE    9$         ;; GO TYPE
8293 052314 000746      BR      2$         ;; GO READ ANOTHER CHAR.
8294 052316 005716      TST     (SP)       ;; RUBOUT KEY SET?
8295 052320 001406      BEQ     7$         ;; BR IF NO
8296 052322 112737 000134 052454      MOVVB   #' \,9$    ;; TYPE A BACK SLASH
8297 052330 104401 052454      TYPE    9$         ;;
8298 052334 005016      CLR     (SP)       ;; CLEAR THE RUBOUT KEY
8299 052336 122713 000025      CMPB   #25,(R3)   ;; IS CHARACTER A CTRL U?
8300 052342 001003      BNE     8$         ;; BR IF NO
8301 052344 104401 052473      TYPE    $CNTLU     ;; TYPE A CONTROL "U"
8302 052350 000726      BR      1$         ;; GO START OVER
8303 052352 122713 000022      CMPB   #22,(R3)   ;; IS CHARACTER A "↑R"?
8304 052356 001011      BNE     3$         ;; BRANCH IF NO
8305 052360 105013      CLRB   (R3)       ;; CLEAR THE CHARACTER
8306 052362 104401 001211      TYPE    , $CRLF    ;; TYPE A "CR" & "LF"
8307 052366 104401 052456      TYPE    , $TTYIN   ;; TYPE THE INPUT STRING
8308 052372 000717      BR      2$         ;; GO PICKUP ANOTHER CHARACTER
8309 052374 104401 001210      TYPE    $QUES     ;; TYPE A '?'
8310 052400 000712      BR      1$         ;; CLEAR THE BUFFER AND LOOP
8311 052402 111337 052454      MOVVB   (R3),9$   ;; ECHO THE CHARACTER
8312 052406 104401 052454      TYPE    9$         ;;
8313 052412 122723 000015      CMPB   #15,(R3)+  ;; CHECK FOR RETURN
8314 052416 001305      BNE     2$         ;; LOOP IF NOT RETURN
8315 052420 105063 177777      CLRB   -1(R3)     ;; CLEAR RETURN (THE 15)
8316 052424 104401 001212      TYPE    $LF       ;; TYPE A LINE FEED
8317 052430 005726      TST     (SP)+     ;; CLEAN RUBOUT KEY FROM THE STACK
8318 052432 012603      MOV     (SP)+,R3  ;; RESTORE R3
8319 052434 011646      MOV     (SP)-,(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
8320 052436 016666 000004 000002      MOV     4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
8321 052444 012766 052456 000004      MOV     $TTYIN,4(SP)
8322 052452 000002      RTI              ;; RETURN
8323 052454 000          9$: .BYTE 0      ;; STORAGE FOR ASCII CHAR. TO TYPE
8324 052455 000          .BYTE 0      ;; TERMINATOR
8325 052456 000010      $TTYIN: .BLKB 8  ;; RESERVE 8 BYTES FOR TTY INPUT
8326 052466 041536 005015 000      $CNTLC: .ASCIZ /↑C/<15><12> ;; CONTROL "C"
8327 052473 136 006525 000012      $CNTLU: .ASCIZ /↑U/<15><12> ;; CONTROL "U"
8328 052500 043536 005015 000      $CNTLG: .ASCIZ /↑G/<15><12> ;; CONTROL "G"
8329 052505 015 051412 051127      $MSWR: .ASCIZ <15><12>/SWR = /
8330 052512 036440 000040      $MNEW: .ASCIZ / NEW = /
8331 052516 020040 042516 020127      .EVEN
8332 052524 020075 000
8333 052530
8334 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
8335
8336 ;*****
8337 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
8338 ;*CHANGE IT TO BINARY.
8339 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
8340 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
8341 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
8342 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.

```

```

8343 ;*CALL:
8344 ;* RDOCT ;: READ AN OCTAL NUMBER
8345 ;* RETURN HERE ;: LOW ORDER BITS ARE ON TOP OF THE STACK
8346 ;* ;: HIGH ORDER BITS ARE IN $HIOCT
8347
8348 052530 011646 $RDOCT: MOV (SP),-(SP) ;: PROVIDE SPACE FOR THE
8349 052532 016666 000004 000002 MOV 4(SF),2(SP) ;: INPUT NUMBER
8350 052540 010046 MOV RO,-(SP) ;: PUSH RO ON STACK
8351 052542 010146 MOV R1,-(SP) ;: PUSH R1 ON STACK
8352 052544 010246 MOV R2,-(SP) ;: PUSH R2 ON STACK
8353 052546 104411 1$: RDLIN ;: READ AN ASCII LINE
8354 052550 012600 MOV (SP)+,RO ;: GET ADDRESS OF 1ST CHARACTER
8355 052552 010037 052656 MOV RO,$$ ;: AND SAVE IT
8356 052556 005001 CLR R1 ;: CLEAR DATA WORD
8357 052560 005002 CLR R2
8358 052562 112046 2$: MOV (RO)+,-(SP) ;: PICKUP THIS CHARACTER
8359 052564 001420 BEQ 3$ ;: IF ZERO GET OUT
8360 052566 122716 000060 CMPB #'0,(SP) ;: MAKE SURE THIS CHARACTER
8361 052572 003026 BGT 4$ ;: IS AN OCTAL DIGIT
8362 052574 122716 000067 CMPB #'7,(SP)
8363 052600 002423 BLT 4$
8364 052602 006301 ASL R1 ;: *2
8365 052604 006102 ROL R2
8366 052606 006301 ASL R1 ;: *4
8367 052610 006102 ROL R2
8368 052612 006301 ASL R1 ;: *8
8369 052614 006102 ROL R2
8370 052616 042716 177770 BIC #'C7,(SP) ;: STRIP THE ASCII JUNK
8371 052622 062601 ADD (SP)+,R1 ;: ADD IN THIS DIGIT
8372 052624 000756 BR 2$ ;: LOOP
8373 052626 005726 3$: TST (SP)+ ;: CLEAN TERMINATOR FROM STACK
8374 052630 010166 000012 MOV R1,12(SP) ;: SAVE THE RESULT
8375 052634 010237 052666 MOV R2,$HIOCT
8376 052640 012602 MOV (SP)+,R2 ;: POP STACK INTO R2
8377 052642 012601 MOV (SP)+,R1 ;: POP STACK INTO R1
8378 052644 012600 MOV (SP)+,RO ;: POP STACK INTO RO
8379 052646 000002 RTI ;: RETURN
8380 052650 005726 4$: TST (SP)+ ;: CLEAN PARTIAL FROM STACK
8381 052652 105010 CLR (RO) ;: SET A TERMINATOR
8382 052654 104401 TYPE ;: TYPE UP THRU THE BAD CHAR.
8383 052656 000000 5$: .WORD 0
8384 052660 104401 001210 TYPE $QUES ;: "?" "CR" & "LF"
8385 052664 000730 BR 1$ ;: TRY AGAIN
8386 052666 000000 $HIOCT: .WORD 0 ;: HIGH ORDER BITS GO HERE
8387 .SBTTL SAVE AND RESTORE RO-R5 ROUTINES
8388
8389 ;: *****
8390 ;: *SAVE RO-R5
8391 ;: *CALL:
8392 ;: * SAVREG
8393 ;: *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
8394 ;: *
8395 ;: *TOP---(+16)
8396 ;: * +2---(+18)
8397 ;: * +4---R5
8398 ;: * +6---R4

```

```

8399
8400
8401
8402
8403
8404 052670
8405 052670 010046
8406 052672 010146
8407 052674 010246
8408 052676 010346
8409 052700 010446
8410 052702 010546
8411 052704 016646 000022
8412 052710 016646 000022
8413 052714 016646 000022
8414 052720 016646 000022
8415 052724 000002
8416
8417
8418
8419
8420 052726
8421 052726 012666 000022
8422 052732 012666 000022
8423 052736 012666 000022
8424 052742 012666 000022
8425 052746 012605
8426 052750 012604
8427 052752 012603
8428 052754 012602
8429 052756 012601
8430 052760 012600
8431 052762 000002
8432
8433
8434
8435
8436
8437
8438 052764 017737 126150 003634
8439 052772 012737 053012 000024
8440 053000 012737 000340 000026
8441 053006 000000
8442 053010 000776
8443
8444
8445
8446
8447 053012 005037 053106
8448 053016 012737 000144 053110
8449 053024 005237 053106
8450 053030 001375
8451 053032 005337 053110
8452 053036 001372
8453 053040 012737 052764 000024
8454 053046 012737 000340 000026

```

```

;* +8---R3
;* +10---R2
;* +12---R1
;* +14---R0

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

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

.SBTTL POWER DOWN AND UP ROUTINE
; ;*****

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

:POWER UP ROUTINE
$PWRUP: CLR $PWRCT ;LOAD WAIT COUNT
MOV #100, $PWRCT+2
1$: INC $PWRCT ;WAIT FOR TELETYPE
BNE 1$
DEC $PWRCT+2
BNE 1$
MOV #SPWRDN, PWRVEC ;SET UP FOR POWER DOWN VECTOR
MOV #PR7, PWRVEC+2

```

8455 053054 012706 001100
8456 053060 104401 053112
8457 053064 004737 044740
8458 053070 013777 003634 126042
8459 053076 013702 001270
8460 053102 000177 126000
8461
8462 053106 000000 000000
8463 053112 047520 042527 000122
8464
8465
8466
8467
8468
8469
8470
8471
8472
8473 053120 010046
8474 053122 016600 000002
8475 053126 005740
8476 053130 111000
8477 053132 006300
8478 053134 016000 053154
8479 053140 000200
8480
8481
8482
8483
8484 053142 011646
8485 053144 016666 000004 000002
8486 053152 000002
8487
8488
8489
8490
8491
8492
8493
8494
8495 053154 053142
8496 053156 050336
8497 053160 050644
8498 053162 050620
8499 053164 050660
8500 053166 051046
8501
8502 053170 051660
8503
8504 053172 051570
8505 053174 052132
8506 053176 052222
8507 053200 052530
8508 053202 052670
8509 053204 052726
8510 053206 047304

```

MOV #STACK,SP ;FORCE STACK
TYPE $POWER ;TYPE POWER
JSR PC,PARCHK ;REINITIALIZE MEMORY CHECK ENABLE
MOV SAVSWR,ASWR ;RESTORE SWITCH REGISTER
MOV $BASE,R2 ;REINITIALISE R2 FOR '611 BASE
JMP @SLPADR ;GO BACK TO LAST TEST

$PWRCT: .WORD 0,0 ;TELETYPE TIME OUT
$POWER: .ASCIZ /POWER/
        EVEN
        .SBTTL TRAP DECODER

;*****
;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
;GO TO THAT ROUTINE.

$TRAP: MOV RO,-(SP) ;;SAVE RO
        MOV 2(SP),RO ;;GET TRAP ADDRESS
        TST -(RO) ;;BACKUP BY 2
        MOVB (RO),RO ;;GET RIGHT BYTE OF TRAP
        ASL RO ;;POSITION FOR INDEXING
        MOV $TRPAD(RO),RO ;;INDEX TO TABLE
        RTS RO ;;GO TO ROUTINE

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

$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
        MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
        RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE

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

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

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

        $CKSWR ;;CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;;CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
        $RDOCT ;;CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
        $SAVREG ;;CALL=SAVREG TRAP+13(104413) SAVE RO-R5 ROUTINE
        $RESREG ;;CALL=RESREG TRAP+14(104414) RESTORE RO-R5 ROUTINE
        $SCOP1$ ;;CALL=SCOP1 TRAP+15(104415) INTERNAL LOOP ON ERROR

```

```

      .SBTTL DATA TABLE FOR PRINT OUT
8511
8512
8513 053210 001220 003604 DT000: .WORD $TESTN,TRAPPC
8514 053214 001220 001116 003500 DT001: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3
8515 053222 003440 003526 003466
8516 053230 003530 003470
8517 053234 001220 001116 003500 DT015: .WORD $TESTN,$ERRPC,E.CS1,T.CS1
8518 053242 003440
8519 053244 001220 001116 003524 DT017: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
8520 053252 003464 003614 003616
8521 053260 003622
8522 053262 001220 001116 003500 DT022: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC
8523 053270 003440 003504 003444
8524 053276 003502 003442
8525 053302 001220 001116 003524 DT025: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,BITCNT
8526 053310 003464 003622
8527 053314 001220 001116 003500 DT031: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC,E.MR1,T.MR1
8528 053322 003440 003504 003444
8529 053330 003502 003442 003524
8530 053336 003464
8531 053340 001220 001116 003500 DT035: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.BA,T.BA
8532 053346 003440 003510 003450
8533 053354 003504 003444
8534 053360 003502 003442
8535 053364 001220 001116 003522 DT041: .WORD E.WC,T.WC
8536 053372 003462 $TESTN,$ERRPC,E.DB,T.DB
8537 053374 001220 001116 003500 DT042: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8538 053402 003440 003510 003450
8539 053410 001220 001116 003522 DT054: .WORD $TESTN,$ERRPC,E.DB,T.DB,WRDCNT
8540 053416 003462 003624
8541 053422 001220 001116 003500 DT055: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,WRDCNT
8542 053430 003440 003510 003450
8543 053436 003624
8544 053440 001220 001116 003500 DT072: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
8545 053446 003440 003510 003450
8546 053454 003514 003454
8547 053460 003504 003444 003502 .WORD E.BA,T.BA,E.WC,T.WC
8548 053466 003442
8549 053470 001220 001116 003500 DT077: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8550 053476 003440 003510 003450
8551 053504 001220 001116 003500 DT150: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,WRDCNT,BITCNT
8552 053512 003440 003624 003622
8553 053520 001220 001116 003500 DT164: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS1,BITCNT
8554 053526 003440 003510 003440
8555 053534 003622
8556 053536 001220 001116 003524 DT170: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT,SECCNT
8557 053544 003464 003612 003614
8558 053552 003616 003620 003622
8559 053560 003626
8560 053562 001220 001116 003524 DT171: .WORD $TESTN,$ERRPC,E.MR1,T.MR1
8561 053570 003464
8562 053572 001220 001116 003500 DT175: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER
8563 053600 003440 003510 003450
8564 053606 003512 003452 003514
8565 053614 003454
8566 053616 001220 001116 003500 DT211: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER

```

8567	053624	003440	003510	003450
8568	053632	003512	003452	003514
8569	053640	003454		
8570	053642	003540	003550	003552
8571	053650	003554		

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

8572
8573
8574 053652 000001 000
8575 053654 002 000
8576 053656 000005 000
8577 053660 000 000
8578 053662 055037 000
8579 053664 000 000
8580 053666 055055 000
8581 053670 002 000
8582 053672 055121 000
8583 053674 000 000
8584 053676 055200 000
8585 053700 006 000
8586 053702 000005 000
8587 053704 000 000
8588 053706 055037 000
8589 053710 000 000
8590 053712 055055 000
8591 053714 002 000
8592 053716 055257 000
8593 053720 000 000
8594 053722 055276 000
8595 053724 002 000
8596 053726 000005 000
8597 053730 000 000
8598 053732 055037 000
8599 053734 000 000
8600 053736 055055 000
8601 053740 002 000
8602 053742 055314 000
8603 053744 000 000
8604 053746 055360 000
8605 053750 005 000
8606 053752 000005 000
8607 053754 000 000
8608 053756 055037 000
8609 053760 000 000
8610 053762 055055 000
8611 053764 002 000
8612 053766 055426 000
8613 053770 000 000
8614 053772 055505 000
8615 053774 006 000
8616 053776 000005 000
8617 054000 000 000
8618 054002 055037 000
8619 054004 000 000
8620 054006 055055 000
8621 054010 002 000
8622 054012 055562 000
8623 054014 000 000
8624 054016 055606 000
8625 054020 003 000
8626 054022 000005 000
8627 054024 000 000

.SBTTL DATA FORMAT FOR PRINT OUT

DF000: .WORD 1
.BYTE 2,0
DF001: .WORD 5,0 ;ERRORS 1-14
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH001A
.BYTE 0,0
.WORD DH001B
.BYTE 6,0
DF015: .WORD 5,0 ;ERRORS 15-16
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH015A
.BYTE 0,0
.WORD DH015B
.BYTE 5,0
DF017: .WORD 5,0 ;ERROR 17-21
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH017A
.BYTE 0,0
.WORD DH017B
.BYTE 5,0
DF022: .WORD 5,0 ;ERROR 22-24
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH022A
.BYTE 0,0
.WORD DH022B
.BYTE 6,0
DF025: .WORD 5,0 ;ERROR 25
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH025A
.BYTE 0,0
.WORD DH025B
.BYTE 3,0
DF031: .WORD 5,0 ;ERROR 31-34
.BYTE 0,0

8628	054026	055037		.WORD	DH000A	
8629	054030	000	000	.BYTE	0,0	
8630	054032	055055		.WORD	DH000B	
8631	054034	002	000	.BYTE	2,0	
8632	054036	055634		.WORD	DH031A	
8633	054040	000	000	.BYTE	0,0	
8634	054042	055733		.WORD	DH031B	
8635	054044	010	000	.BYTE	8.,0	
8636	054046	000005		.WORD	5	;ERRORS 35-40
8637	054050	000	000	.BYTE	0,0	
8638	054052	055037		.WORD	DH000A	
8639	054054	000	000	.BYTE	0,0	
8640	054056	055055		.WORD	DH000B	
8641	054060	002	000	.BYTE	2,0	
8642	054062	056031		.WORD	DH035A	
8643	054064	000	000	.BYTE	0,0	
8644	054066	056130		.WORD	DH035B	
8645	054070	010	000	.BYTE	8.,0	
8646	054072	000005		.WORD	5	;ERROR 41
8647	054074	000	000	.BYTE	0,0	
8648	054076	055037		.WORD	DH000A	
8649	054100	000	000	.BYTE	0,0	
8650	054102	055055		.WORD	DH000B	
8651	054104	002	000	.BYTE	2,0	
8652	054106	056225		.WORD	DH041A	
8653	054110	000	000	.BYTE	0,0	
8654	054112	056242		.WORD	DH041B	
8655	054114	002	000	.BYTE	2.,0	
8656	054116	000005		.WORD	5	;ERRORS 42-43
8657	054120	000	000	.BYTE	0,0	
8658	054122	055037		.WORD	DH000A	
8659	054124	000	000	.BYTE	0,0	
8660	054126	055055		.WORD	DH000B	
8661	054130	002	000	.BYTE	2,0	
8662	054132	056257		.WORD	DH042A	
8663	054134	000	000	.BYTE	0,0	
8664	054136	056316		.WORD	DH042B	
8665	054140	004	000	.BYTE	4,0	
8666	054142	000005		.WORD	5	;ERROR 54
8667	054144	000	000	.BYTE	0,0	
8668	054146	055037		.WORD	DH000A	
8669	054150	000	000	.BYTE	0,0	
8670	054152	055055		.WORD	DH000B	
8671	054154	002	000	.BYTE	2,0	
8672	054156	056354		.WORD	DH054A	
8673	054160	000	000	.BYTE	0,0	
8674	054162	056401		.WORD	DH054B	
8675	054164	003	000	.BYTE	3,0	
8676	054166	000005		.WORD	5	;ERROR 55
8677	054170	000	000	.BYTE	0,0	
8678	054172	055037		.WORD	DH000A	
8679	054174	000	000	.BYTE	0,0	
8680	054176	055055		.WORD	DH000B	
8681	054200	002	000	.BYTE	2,0	
8682	054202	056427		.WORD	DH055A	
8683	054204	000	000	.BYTE	0,0	

8684	054206	056474		.WORD	DH055B	
8685	054210	005	000	.BYTE	5,0	
8686	054212	000007		DF072: .WORD	7	
8687	054214	000	000	.BYTE	0,0	
8688	054216	055037		.WORD	DH000A	
8689	054220	000	000	.BYTE	0,0	
8690	054222	055055		.WORD	DH0C0B	
8691	054224	002	000	.BYTE	2,0	
8692	054226	056542		.WORD	DH072A	
8693	054230	000	000	.BYTE	0,0	
8694	054232	056601		.WORD	DH072B	
8695	054234	004	000	.BYTE	4,0	
8696	054236	056637		.WORD	DH072C	
8697	054240	000	000	.BYTE	0,0	
8698	054242	056716		.WORD	DH072D	
8699	054244	006	000	.BYTE	6,0	
8700	054246	000007		DF077: .WORD	7	
8701	054250	000	000	.BYTE	0,0	
8702	054252	055037		.WORD	DH000A	
8703	054254	000	000	.BYTE	0,0	
8704	054256	055055		.WORD	DH000B	
8705	054260	002	000	.BYTE	2,0	
8706	054262	056542		.WORD	DH072A	
8707	054264	000	000	.BYTE	0,0	
8708	054266	056601		.WORD	DH072B	
8709	054270	004	000	.BYTE	4,0	
8710	054272	000005		DF150: .WORD	5	;ERROR 150
8711	054274	000	000	.BYTE	0,0	
8712	054276	055037		.WORD	DH000A	
8713	054300	000	000	.BYTE	0,0	
8714	054302	055055		.WORD	DH000B	
8715	054304	002	000	.BYTE	2,0	
8716	054306	056773		.WORD	DH150A	
8717	054310	000	000	.BYTE	0,0	
8718	054312	057027		.WORD	DH150B	
8719	054314	004	000	.BYTE	4,0	
8720	054316	000005		DF164: .WORD	5	;ERROR 164
8721	054320	000	000	.BYTE	0,0	
8722	054322	055037		.WORD	DH000A	
8723	054324	000	000	.BYTE	0,0	
8724	054326	055055		.WORD	DH000B	
8725	054330	002	000	.BYTE	2,0	
8726	054332	057065		.WORD	DH164A	
8727	054334	000	000	.BYTE	0,0	
8728	054336	057131		.WORD	DH164B	
8729	054340	005	000	.BYTE	5,0	
8730	054342	000005		DF170: .WORD	5	;ERROR 170
8731	054344	000	000	.BYTE	0,0	
8732	054346	055037		.WORD	DH000A	
8733	054350	000	000	.BYTE	0,0	
8734	054352	055055		.WORD	DH000B	
8735	054354	002	000	.BYTE	2,0	
8736	054356	057177		.WORD	DH170A	
8737	054360	000	000	.BYTE	0,0	
8738	054362	057276		.WORD	DH170B	
8739	054364	010	000	.BYTE	8.,0	

8740	054366	000005		DF171:	.WORD	5		;ERRORS 171-174
8741	054370	000	000		.BYTE	0,0		
8742	054372	055037			.WORD	DH000A		
8743	054374	000	000		.BYTE	0,0		
8744	054376	055055			.WORD	DH000B		
8745	054400	002	000		.BYTE	2,0		
8746	054402	057374			.WORD	DH171A		
8747	054404	000	000		.BYTE	0,0		
8748	054406	057413			.WORD	DH171B		
8749	054410	002	000		.BYTE	2,0		
8750	054412	000005		DF175:	.WORD	5		;ERRORS 175-210
8751	054414	000	000		.BYTE	0,0		
8752	054416	055037			.WORD	DH000A		
8753	054420	000	000		.BYTE	0,0		
8754	054422	055055			.WORD	DH000B		
8755	054424	002	000		.BYTE	2,0		
8756	054426	057431			.WORD	DH175A		
8757	054430	000	000		.BYTE	0,0		
8758	054432	057530			.WORD	DH175B		
8759	054434	010	000		.BYTE	8.,0		
8760	054436	000007		DF211:	.WORD	7		;ERRORS 211-214
8761	054440	000	000		.BYTE	0,0		
8762	054442	055037			.WORD	DH000A		
8763	054444	000	000		.BYTE	0,0		
8764	054446	055055			.WORD	DH000B		
8765	054450	002	000		.BYTE	2,0		
8766	054452	057431			.WORD	DH175A		
8767	054454	000	000		.BYTE	0,0		
8768	054456	057530			.WORD	DH175B		
8769	054460	010	000		.BYTE	8.,0		
8770	054462	057625			.WORD	DH211A		
8771	054464	000	000		.BYTE	0,0		
8772	054466	057653			.WORD	DH211B		
8773	054470	004	000		.BYTE	4,0		

.SBTTL ASCII MESSAGES

8774					
8775					
8776	054472	005015	045522	030466	OPR001: .ASCIZ <15><12>/RK611 VECTOR ADDRESS (/
8777	054500	020061	042526	052103	
8778	054506	051117	040440	042104	
8779	054514	042522	051523	024040	
8780	054522	000040			
8781	054524	024440	036440	000040	OPR002: .ASCIZ /) = /
8782	054532	045522	030466	020061	OPR003: .ASCIZ /RK611 VECTOR ADDRESS (/
8783	054540	042526	052103	051117	
8784	054546	040440	042104	042522	
8785	054554	051523	024040	000040	
8786	054562	045522	030466	020061	OPR004: .ASCIZ /RK611 PRIORITY (/
8787	054570	051120	047511	044522	
8788	054576	054524	024040	000040	
8789	054604	005015	047062	020104	OPR006: .ASCIZ <15><12>/2ND PASS RUN TIME IS APPROX 13 MINUTES/<15><12>
8790	054612	040520	051523	051040	
8791	054620	047125	052040	046511	
8792	054626	020105	051511	040440	
8793	054634	050120	047522	020130	
8794	054642	031461	046440	047111	
8795	054650	052125	051505	005015	
8796	054656	000			
8797	054657	015	025012	025052	OPR007: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
8798	054664	025052	020040	050040	
8799	054672	047522	051107	046501	
8800	054700	044040	046101	042524	
8801	054706	020104	020040	025052	
8802	054714	025052	025052	005015	
8803	054722	000			
8804	054723	040	000040		
8805	054726	005015	051120	043517	SPACE2: .ASCIZ / /
8806	054734	040522	020115	041101	ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
8807	054742	051117	042524	020104	
8808	054750	042502	040503	051525	
8809	054756	020105	051105	047522	
8810	054764	020122	044124	042522	
8811	054772	044123	046117	020104	
8812	055000	054105	042503	042105	
8813	055006	042105	005015	000	
8814	055013	015	052012	051505	TSTBY1: .ASCIZ <15><12>/TEST /
8815	055020	020124	000		
8816	055023	040	054502	040520	TSTBY2: .ASCIZ / BYPASSED/<15><12>
8817	055030	051523	042105	005015	
8818	055036	000			

.SBTTL DATA HEADERS				
8819				
8820				
8821	055037	124	051505	020124
8822	055044	020040	042440	051122
8823	055052	051117	000	
8824	055055	116	046525	020040
8825	055062	020040	050040	000103
8826	055070	042524	052123	020040
8827	055076	020040	051124	050101
8828	055104	005015		
8829	055106	052516	020115	020040
8830	055114	020040	041520	000
8831	055121	105	050130	041505
8832	055126	020124	040440	052503
8833	055134	040524	020114	042440
8834	055142	050130	041505	020124
8835	055150	040440	052103	040525
8836	055156	020114	042440	050130
8837	055164	041505	020124	040440
8838	055172	052103	040525	000114
8839	055200	045522	051503	020061
8840	055206	020040	045522	051503
8841	055214	020061	020040	042515
8842	055222	051523	040440	020040
8843	055230	042515	051523	040440
8844	055236	020040	042515	051523
8845	055244	041040	020040	042515
8846	055252	051523	041040	000
8847	055257	105	050130	041505
8848	055264	020124	040440	052103
8849	055272	040525	000114	
8850	055276	045522	051503	020061
8851	055304	020040	045522	051503
8852	055312	000061		
8853	055314	054105	042520	052103
8854	055322	020040	041501	052524
8855	055330	046101	020040	051120
8856	055336	051505	047105	020124
8857	055344	051120	053105	052517
8858	055352	020123	044502	000124
8859	055360	045522	051115	020061
8860	055366	020040	045522	051115
8861	055374	020061	020040	044502
8862	055402	020124	020040	020040
8863	055410	044502	020124	020040
8864	055416	020040	047503	047125
8865	055424	000124		
8866	055426	054105	042520	052103
8867	055434	020040	041501	052524
8868	055442	046101	020040	054105
8869	055450	042520	052103	020040
8870	055456	041501	052524	046101
8871	055464	020040	054105	042520
8872	055472	052103	020040	041501
8873	055500	052524	046101	000
8874	055505	122	041513	030523

```

.DH000A: .ASCIZ /TEST ERROR/
.DH000B: .ASCIZ /NUM PC/
.DH000C: .ASCII /TEST TRAP/<15><12>
.DH001A: .ASCIZ /EXPECT ACUTAL EXPECT ACTUAL EXPECT ACTUAL/
.DH001B: .ASCIZ /RKCS1 RKCS1 MESS A MESS A MESS B MESS B/
.DH015A: .ASCIZ /EXPECT ACTUAL/
.DH015B: .ASCIZ /RKCS1 RKCS1/
.DH017A: .ASCIZ /EXPECT ACTUAL PRESENT PREVIOUS BIT/
.DH017B: .ASCIZ /RKMR1 RKMR1 BIT BIT COUNT/
.DH022A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
.DH022B: .ASCIZ /RKCS1 RKCS1 RKBA RKBA RKWC RKWC/

```



```

9085 .SBTTL ERROR MESSAGES
9086
9087 057710 047125 054105 042520 EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
9088 057716 052103 042105 046440
9089 057724 046505 051117 020131
9090 057732 040520 044522 054524
9091 057740 042440 040516 046102
9092 057746 020105 051124 050101
9093 057754 000
9094 057755 101 052124 046505 EM200: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER/
9095 057762 052120 047111 020107
9096 057770 047524 041440 042510
9097 057776 045503 051440 042505
9098 060004 020113 042515 051523
9099 060012 043501 020105 051106
9100 060020 046517 051040 040505
9101 060026 020104 042510 042101
9102 060034 051105 000
9103 060037 101 052124 046505 EM201: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER/
9104 060044 052120 047111 020107
9105 060052 047524 041440 042510
9106 060060 045503 051440 042505
9107 060066 020113 042515 051523
9108 060074 043501 020105 051106
9109 060102 046517 053440 044522
9110 060110 042524 044040 040505
9111 060116 042504 000122
9112 060122 052101 042524 050115 EM202: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM READ HEADER/
9113 060130 044524 043516 052040
9114 060136 020117 044103 041505
9115 060144 020113 046103 040505
9116 060152 020122 051104 053111
9117 060160 020105 042515 051523
9118 060166 043501 020105 051106
9119 060174 046517 051040 040505
9120 060202 020104 042510 042101
9121 060210 051105 000
9122 060213 101 052124 046505 EM203: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM WRITE HEADER/
9123 060220 052120 047111 020107
9124 060226 047524 041440 042510
9125 060234 045503 041440 042514
9126 060242 051101 042040 044522
9127 060250 042526 046440 051505
9128 060256 040523 042507 043040
9129 060264 047522 020115 051127
9130 060272 052111 020105 042510
9131 060300 042101 051105 000
9132 060305 101 052124 046505 EM204: .ASCIZ /ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT/
9133 060312 052120 047111 020107
9134 060320 020101 042522 042101
9135 060326 044040 040505 042504
9136 060334 020122 047524 041440
9137 060342 042510 045503 051440
9138 060350 041505 047524 020122
9139 060356 052520 051514 020105
9140 060364 042504 042524 052103
    
```

9141	060372	000				
9142	060373	101	052124	046505	EM205:	.ASCIZ /ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT/
9143	060400	052120	047111	020107		
9144	060406	020101	051127	052111		
9145	060414	020105	042510	042101		
9146	060422	051105	052040	020117		
9147	060430	044103	041505	020113		
9148	060436	047111	042504	020130		
9149	060444	052520	051514	020105		
9150	060452	042504	042524	052103		
9151	060460	000				
9152	060461	101	052124	046505	EM206:	.ASCIZ /ATTEMPTING AN NPR READ OF ONE WORD/
9153	060466	052120	047111	020107		
9154	060474	047101	047040	051120		
9155	060502	051040	040505	020104		
9156	060510	043117	047440	042516		
9157	060516	053440	051117	000104		
9158	060524	052101	042524	050115	EM207:	.ASCIZ /ATTEMPTING AN NPR READ/
9159	060532	044524	043516	040440		
9160	060540	020116	050116	020122		
9161	060546	042522	042101	000		
9162	060553	101	052124	046505	EM208:	.ASCIZ /ATTEMPTING NPR READ CHECKING ZERO DETECT/
9163	060560	052120	047111	020107		
9164	060566	050116	020122	042522		
9165	060574	042101	041440	042510		
9166	060602	045503	047111	020107		
9167	060610	042532	047522	042040		
9168	060616	052105	041505	000124		
9169	060624	052101	042524	050115	EM209:	.ASCIZ /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
9170	060632	044524	043516	047040		
9171	060640	051120	051040	040505		
9172	060646	020104	044527	044124		
9173	060654	041040	051525	040440		
9174	060662	042104	042522	051523		
9175	060670	044440	041516	042522		
9176	060676	042515	052116	044440		
9177	060704	044116	041111	052111		
9178	060712	000				
9179	060713	101	052124	046505	EM210:	.ASCII /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
9180	060720	052120	047111	020107		
9181	060726	050116	020122	042522		
9182	060734	042101	053440	052111		
9183	060742	020110	052502	020123		
9184	060750	042101	051104	051505		
9185	060756	020123	047111	051103		
9186	060764	046505	047105	020124		
9187	060772	047111	044510	044502		
9188	061000	124				
9189	061001	040	044103	041505		.ASCIZ / CHECKING ZERO DETECT/
9190	061006	044513	043516	055040		
9191	061014	051105	020117	042504		
9192	061022	042524	052103	000		
9193	061027	101	052124	046505	EM211:	.ASCIZ /ATTEMPTING TO FORCE NON-EXISTENT MEMORY/
9194	061034	052120	047111	020107		
9195	061042	047524	043040	051117		
9196	061050	042503	047040	047117		

9197	061056	042455	044530	052123	
9198	061064	047105	020124	042515	
9199	061072	047515	054522	000	
9200	061077	101	052124	046505	EM212: .ASCIZ /ATTEMPTING TO CLEAR NON-EXISTENT MEMORY/
9201	061104	052120	047111	020107	
9202	061112	047524	041440	042514	
9203	061120	051101	047040	047117	
9204	061126	042455	044530	052123	
9205	061134	047105	020124	042515	
9206	061142	047515	054522	000	
9207	061147	124	051505	044524	EM213: .ASCIZ /TESTING EXTENDED MEMORY ADDRESSING BITS/
9208	061154	043516	042440	052130	
9209	061162	047105	042504	020104	
9210	061170	042515	047515	054522	
9211	061176	040440	042104	042522	
9212	061204	051523	047111	020107	
9213	061212	044502	051524	000	
9214	061217	101	052124	046505	EM214: .ASCIZ /ATTEMPTING TO FORCE UNIBUS PARITY ERROR/
9215	061224	052120	047111	020107	
9216	061232	047524	043040	051117	
9217	061240	042503	052440	044516	
9218	061246	052502	020123	040520	
9219	061254	044522	054524	042440	
9220	061262	051122	051117	000	
9221	061267	101	052124	046505	EM215: .ASCIZ /ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY/
9222	061274	052120	047111	020107	
9223	061302	050116	020122	042522	
9224	061310	042101	047440	020106	
9225	061316	047514	040503	044524	
9226	061324	047117	050040	044522	
9227	061332	051117	052040	020117	
9228	061340	040502	020104	040520	
9229	061346	044522	054524	000	
9230	061353	101	052124	046505	EM216: .ASCIZ /ATTEMPTING TO CLEAR UNIBUS PARITY ERROR/
9231	061360	052120	047111	020107	
9232	061366	047524	041440	042514	
9233	061374	051101	052440	044516	
9234	061402	052502	020123	040520	
9235	061410	044522	054524	042440	
9236	061416	051122	051117	000	
9237	061423	101	052124	046505	EM217: .ASCIZ /ATTEMPTING 18 BIT NPR READ/
9238	061430	052120	047111	020107	
9239	061436	034061	041040	052111	
9240	061444	047040	051120	051040	
9241	061452	040505	000104		
9242	061456	052101	042524	050115	EM218: .ASCIZ /ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT/
9243	061464	044524	043516	030440	
9244	061472	020070	044502	020124	
9245	061500	050116	020122	042522	
9246	061506	042101	041440	042510	
9247	061514	045503	047111	020107	
9248	061522	042532	047522	042040	
9249	061530	052105	041505	000124	
9250	061536	052101	042524	050115	EM219: .ASCIZ /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
9251	061544	044524	043516	030440	
9252	061552	020070	044502	020124	

9253	061560	050116	020122	042522
9254	061566	042101	053440	052111
9255	061574	020110	044502	020124
9256	061602	033061	024040	040520
9257	061610	020051	042523	000124
9258	061616	052101	042524	050115
9259	061624	044524	043516	030440
9260	061632	020070	044502	020124
9261	061640	050116	020122	042522
9262	061646	042101	053440	052111
9263	061654	020110	044502	020124
9264	061662	033061	024040	040520
9265	061670	020051	042523	124
9266	061675	101	052124	046505
9267	061702	052120	047111	020107
9268	061710	044523	052515	040514
9269	061716	044524	047117	047440
9270	061724	020106	040504	040524
9271	061732	044440	020116	042522
9272	061740	042101	044040	040505
9273	061746	042504	000122	
9274	061752	052101	042524	050115
9275	061760	044524	043516	051040
9276	061766	040505	020104	042510
9277	061774	042101	051105	044440
9278	062002	020116	040515	047111
9279	062010	040524	042516	041516
9280	062016	020105	047515	042504
9281	062024	000		
9282	062025	101	052124	046505
9283	062032	052120	047111	020107
9284	062040	040504	040524	041040
9285	062046	043125	042506	020122
9286	062054	042522	042101	040440
9287	062062	052106	051105	051040
9288	062070	040505	020104	042510
9289	062076	042101	051105	000
9290	062103	101	052124	046505
9291	062110	052120	047111	020107
9292	062116	044523	052515	040514
9293	062124	044524	047117	047440
9294	062132	020106	040504	040524
9295	062140	044440	020116	042522
9296	062146	042101	044040	040505
9297	062154	042504	020122	030450
9298	062162	020070	044502	020124
9299	062170	047506	046522	052101
9300	062176	000051		
9301	062200	052101	042524	050115
9302	062206	044524	043516	051040
9303	062214	040505	020104	042510
9304	062222	042101	051105	024040
9305	062230	034061	041040	052111
9306	062236	043040	051117	040515
9307	062244	024524	044440	020116
9308	062252	040515	047111	020124

EM220: .ASCII /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/

EM221: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER/

EM222: .ASCIZ /ATTEMPTING READ HEADER IN MAINTANENCE MODE/

EM223: .ASCIZ /ATTEMPTING DATA BUFFER READ AFTER READ HEADER/

EM224: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER (18 BIT FORMAT)/

EM225: .ASCIZ /ATTEMPTING READ HEADER (18 BIT FORMAT) IN MAINT MODE/

9309	062260	047515	042504	000	
9310	062265	101	052124	046505	EM226: .ASCII /ATTEMPTING DATA BUFFER READ AFTER/<12><15>
9311	062272	052120	047111	020107	
9312	062300	040504	040524	041040	
9313	062306	043125	042506	020122	
9314	062314	042522	042101	040440	
9315	062322	052106	051105	006412	
9316	062330	042522	042101	044040	.ASCIZ /READ HEADER IN 18 BIT FORMAT/
9317	062336	040505	042504	020122	
9318	062344	047111	030440	020070	
9319	062352	044502	020124	047506	
9320	062360	046522	052101	000	
9321	062365	101	052124	046505	EM227: .ASCII /ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER/<15><12>
9322	062372	052120	047111	020107	
9323	062400	047524	041440	042510	
9324	062406	045503	051440	047131	
9325	062414	044103	042040	052105	
9326	062422	041505	020124	047117	
9327	062430	051040	040505	020104	
9328	062436	042510	042101	051105	
9329	062444	005015			
9330	062446	044103	041505	044513	.ASCIZ /CHECKING ZERO DETECT/
9331	062454	043516	055040	051105	
9332	062462	020117	042504	042524	
9333	062470	052103	000		
9334	062473	101	052124	046505	EM230: .ASCII /ATTEMPTING TO CHECK WRITING OF ZEROES BETWEEN INDEX/<15><12>
9335	062500	052120	047111	020107	
9336	062506	047524	041440	042510	
9337	062514	045503	053440	044522	
9338	062522	044524	043516	047440	
9339	062530	020106	042532	047522	
9340	062536	051505	041040	052105	
9341	062544	042527	047105	044440	
9342	062552	042116	054105	005015	
9343	062560	047101	020104	042523	.ASCIZ /AND SECTOR PULSE/
9344	062566	052103	051117	050040	
9345	062574	046125	042523	000	
9346	062601	101	052124	046505	EM231: .ASCII /ATTEMPTING TO RESET WRITE GATE BY SETTING/<15><12>
9347	062606	052120	047111	020107	
9348	062614	047524	051040	051505	
9349	062622	052105	053440	044522	
9350	062630	042524	043440	052101	
9351	062636	020105	054502	051440	
9352	062644	052105	044524	043516	
9353	062652	005015			
9354	062654	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
9355	062662	050040	046125	042523	
9356	062670	044440	020116	020101	
9357	062676	051127	052111	020105	
9358	062704	042510	042101	051105	
9359	062712	041440	046517	040515	
9360	062720	042116	000		
9361	062723	101	052124	046505	EM232: .ASCII /ATTEMPTING TO SET WRITE GATE BY RESETTING/<15><12>
9362	062730	052120	047111	020107	
9363	062736	047524	051440	052105	
9364	062744	053440	044522	042524	

9365	062752	043440	052101	020105	
9366	062760	054502	051040	051505	
9367	062766	052105	044524	043516	
9368	062774	005015			
9369	062776	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
9370	063004	050040	046125	042523	
9371	063012	044440	020116	020101	
9372	063020	051127	052111	020105	
9373	063026	042510	042101	051105	
9374	063034	041440	046517	040515	
9375	063042	042116	000		
9376	063045	101	052124	046505	EM233: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER/
9377	063052	052120	047111	020107	
9378	063060	047524	053440	044522	
9379	063066	042524	051440	047131	
9380	063074	044103	047440	020106	
9381	063102	042510	042101	051105	
9382	063110	000			
9383	063111	101	052124	046505	EM234: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA/
9384	063116	052120	047111	020107	
9385	063124	047524	053440	044522	
9386	063132	042524	044040	040505	
9387	063140	042504	020122	040504	
9388	063146	040524	000		
9389	063151	101	052124	046505	EM235: .ASCII /ATTEMPTING TO RESET WRITE GATE WITH SECOND/<15><12>
9390	063156	052120	047111	020107	
9391	063164	047524	051040	051505	
9392	063172	052105	053440	044522	
9393	063200	042524	043440	052101	
9394	063206	020105	044527	044124	
9395	063214	051440	041505	047117	
9396	063222	006504	012		
9397	063225	111	042116	054105	.ASCIZ /INDEX PULSE OF WRITE HEADER/
9398	063232	050040	046125	042523	
9399	063240	047440	020106	051127	
9400	063246	052111	020105	042510	
9401	063254	042101	051105	000	
9402	063261	101	052124	046505	EM236: .ASCIZ /ATTEMPTING TO COMPLETE WRITE HEADER IN MAINT MODE/
9403	063266	052120	047111	020107	
9404	063274	047524	041440	046517	
9405	063302	046120	052105	020105	
9406	063310	051127	052111	020105	
9407	063316	042510	042101	051105	
9408	063324	044440	020116	040515	
9409	063332	047111	020124	047515	
9410	063340	042504	000		
9411	063343	101	052124	046505	EM237: .ASCIZ /ATTEMPTING TO WRITE GAP OR DATA SYNCH/
9412	063350	052120	047111	020107	
9413	063356	047524	053440	044522	
9414	063364	042524	043440	050101	
9415	063372	047440	020122	040504	
9416	063400	040524	051440	047131	
9417	063406	044103	000		
9418	063411	101	052124	046505	EM238: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD/
9419	063416	052120	047111	020107	
9420	063424	047524	053440	044522	

9421	063432	042524	042040	052101	
9422	063440	020101	044506	046105	
9423	063446	000104			
9424	063450	052101	042524	050115	EM239: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER USING 24 SECTOR FORMAT/
9425	063456	047111	020107	047524	
9426	063464	053440	044522	042524	
9427	063472	051440	047131	044103	
9428	063500	047440	020106	042510	
9429	063506	042101	051105	052440	
9430	063514	044523	043516	031040	
9431	063522	020064	042523	052103	
9432	063530	051117	043040	051117	
9433	063536	040515	000124		
9434	063542	052101	042524	050115	EM240: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA USING 24 SECTOR FORMAT/
9435	063550	044524	043516	052040	
9436	063556	020117	051127	052111	
9437	063564	020105	042510	042101	
9438	063572	051105	042040	052101	
9439	063600	020101	051525	047111	
9440	063606	020107	032062	051440	
9441	063614	041505	047524	020122	
9442	063622	047506	046522	052101	
9443	063630	000			
9444	063631	101	052124	046505	EM241: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26 SECTOR)/
9445	063636	052120	047111	020107	
9446	063644	047524	043040	051117	
9447	063652	042503	043040	051117	
9448	063660	040515	020124	051105	
9449	063666	047522	020122	041450	
9450	063674	046506	020124	020075	
9451	063702	033062	051440	041505	
9452	063710	047524	024522	000	
9453	063715	101	052124	046505	EM242: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24 SECTOR)/
9454	063722	052120	047111	020107	
9455	063730	047524	043040	051117	
9456	063736	042503	043040	051117	
9457	063744	040515	020124	051105	
9458	063752	047522	020122	041450	
9459	063760	046506	020124	020075	
9460	063766	032062	051440	041505	
9461	063774	047524	024522	000	
9462	064001	101	052124	046505	EM243: .ASCIZ /ATTEMPTING TO FORCE CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS/
9463	064006	044520	043516	052040	
9464	064014	020117	047506	041522	
9465	064022	020105	047503	052116	
9466	064030	047522	046114	051105	
9467	064036	042440	051122	051117	
9468	064044	053440	052111	020110	
9469	064052	040506	046125	020124	
9470	064060	044502	020124	047111	
9471	064066	042040	044522	042526	
9472	064074	046440	051505	000123	
9473	064102	052101	042524	050115	EM244: .ASCIZ /ATTEMPTING TO CLEAR ERROR/
9474	064110	044524	043516	052040	
9475	064116	020117	046103	040505	
9476	064124	020122	051105	047522	

9477	064133	000122			
9478	064134	047503	046515	047101	EM3000: .ASCIZ /COMMAND AND STATUS REG 1 INCORRECT/
9479	064142	020104	047101	020104	
9480	064150	052123	052101	051525	
9481	064156	051040	043505	030440	
9482	064164	044440	041516	051117	
9483	064172	042522	052103	000	
9484	064177	115	051505	040523	EM3001: .ASCIZ /MESSAGE A INCORRECT/
9485	064204	042507	040440	044440	
9486	064212	041516	051117	042522	
9487	064220	052103	000		
9488	064223	115	051505	040523	EM3002: .ASCIZ /MESSAGE B INCORRECT/
9489	064230	042507	041040	044440	
9490	064236	041516	051117	042522	
9491	064244	052103	000		
9492	064247	103	030523	044440	EM3003: .ASCIZ /CSI INCORRECT AFTER SENDING DRIVE CLEAR/
9493	064254	041516	051117	042522	
9494	064262	052103	040440	052106	
9495	064270	051105	051440	047105	
9496	064276	044504	043516	042040	
9497	064304	044522	042526	041440	
9498	064312	042514	051101	000	
9499	064317	103	030523	044440	EM3004: .ASCIZ /CSI INCORRECT AFTER AFTER DATA SIMULATION/
9500	064324	041516	051117	042522	
9501	064332	052103	040440	052106	
9502	064340	051105	040440	052106	
9503	064346	051105	042040	052101	
9504	064354	020101	044523	052515	
9505	064362	040514	044524	047117	
9506	064370	000			
9507	064371	115	044501	052116	EM3005: .ASCII /MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9508	064376	051040	043505	020056	
9509	064404	020061	047111	047503	
9510	064412	051122	041505	020124	
9511	064420	052504	044522	043516	
9512	064426	042040	052101	020101	
9513	064434	044523	052515	040514	
9514	064442	044524	047117		
9515	064446	005015	043101	042524	.ASCIZ <15><12>/AFTER SECTOR PULSE/
9516	064454	020122	042523	052103	
9517	064462	051117	050040	046125	
9518	064470	042523	000		
9519	064473	115	044501	052116	EM3006: .ASCII /MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9520	064500	051040	043505	020056	
9521	064506	020061	047111	047503	
9522	064514	051122	041505	020124	
9523	064522	052504	044522	043516	
9524	064530	042040	052101	020101	
9525	064536	044523	052515	040514	
9526	064544	044524	047117		
9527	064550	005015	043101	042524	.ASCIZ <15><12>/AFTER INDEX PULSE/
9528	064556	020122	047111	042504	
9529	064564	020130	052520	051514	
9530	064572	000105			
9531	064574	040515	047111	020124	EM3007: .ASCII /MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9532	064602	042522	027107	030440	

9533	064610	044440	041516	051117	
9534	064616	042522	052103	042040	
9535	064624	051125	047111	020107	
9536	064632	040504	040524	051440	
9537	064640	046511	046125	052101	
9538	064646	047511	116		
9539	064651	015	047012	020117	.ASCIZ <15><12>/NO SECTOR OR INDEX PULSE SUPPLIED/
9540	064656	042523	052103	051117	
9541	064664	047440	020122	047111	
9542	064672	042504	020130	052520	
9543	064700	051514	020105	052523	
9544	064706	050120	044514	042105	
9545	064714	000			
9546	064715	102	051525	040440	EM3008: .ASCIZ /BUS ADDRESS INCORRECT AFTER SENDING DRIVE CLEAR/
9547	064722	042104	042522	051523	
9548	064730	044440	041516	051117	
9549	064736	042522	052103	040440	
9550	064744	052106	051105	051440	
9551	064752	047105	044504	043516	
9552	064760	042040	044522	042526	
9553	064766	041440	042514	051101	
9554	064774	000			
9555	064775	127	051117	020104	EM3009: .ASCIZ /WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR/
9556	065002	047503	047125	020124	
9557	065010	047111	047503	051122	
9558	065016	041505	020124	043101	
9559	065024	042524	020122	042523	
9560	065032	042116	047111	020107	
9561	065040	051104	053111	020105	
9562	065046	046103	040505	000122	
9563	065054	051503	020061	044103	EM3010: .ASCIZ /CSI CHANGED DURING COMMAND EXECUTION/
9564	065062	047101	042507	020104	
9565	065070	052504	044522	043516	
9566	065076	041440	046517	040515	
9567	065104	042116	042440	042530	
9568	065112	052503	044524	047117	
9569	065120	000			
9570	065121	102	051525	040440	EM3011: .ASCIZ /BUS ADDRESS CHANGED BEFORE INDEX PULSE/
9571	065126	042104	042522	051523	
9572	065134	041440	040510	043516	
9573	065142	042105	041040	043105	
9574	065150	051117	020105	047111	
9575	065156	042504	020130	052520	
9576	065164	051514	000105		
9577	065170	047527	042122	041440	EM3012: .ASCIZ /WORD COUNT CHANGED BEFORE INDEX PULSE/
9578	065176	052517	052116	041440	
9579	065204	040510	043516	042105	
9580	065212	041040	043105	051117	
9581	065220	020105	047111	042504	
9582	065226	020130	052520	051514	
9583	065234	000105			
9584	065236	051503	020061	044103	EM3013: .ASCIZ /CSI CHANGE AFTER INDEX PULSE/
9585	065244	047101	042507	040440	
9586	065252	052106	051105	044440	
9587	065260	042116	054105	050040	
9588	065266	046125	042523	000	

9589	065273	115	044501	052116	EM3014: .ASCIZ /MAINT REG 1 INCORRECT BEFORE INDEX PULSE/
9590	065300	051040	043505	030440	
9591	065306	044440	041516	051117	
9592	065314	042522	052103	041040	
9593	065322	043105	051117	020105	
9594	065330	047111	042504	020130	
9595	065336	052520	051514	000105	
9596	065344	052502	020123	042101	EM3015: .ASCIZ /BUS ADDRESS CHANGED AFTER INDEX PULSE/
9597	065352	051104	051505	020123	
9598	065360	044103	047101	042507	
9599	065366	020104	043101	042524	
9600	065374	020122	047111	042504	
9601	065402	020130	052520	051514	
9602	065410	000105			
9603	065412	047527	042122	041440	EM3016: .ASCIZ /WORD COUNT CHANGED AFTER INDEX PULSE/
9604	065420	052517	052116	041440	
9605	065426	040510	043516	042105	
9606	065434	040440	052106	051105	
9607	065442	044440	042116	054105	
9608	065450	050040	046125	042523	
9609	065456	000			
9610	065457	103	046517	040515	EM3018: .ASCIZ /COMMAND AND STATUS REG. 2 INCORRECT/
9611	065464	042116	040440	042116	
9612	065472	051440	040524	052524	
9613	065500	020123	042522	027107	
9614	065506	031040	044440	041516	
9615	065514	051117	042522	052103	
9616	065522	000			
9617	065523	102	051525	040440	EM3019: .ASCIZ /BUS ADD REG INCORRECT/
9618	065530	042104	051040	043505	
9619	065536	044440	041516	051117	
9620	065544	042522	052103	000	
9621	065551	127	051117	020104	EM3020: .ASCIZ /WORD COUNT REG INCORRECT/
9622	065556	047503	047125	020124	
9623	065564	042522	020107	047111	
9624	065572	047503	051122	041505	
9625	065600	000124			
9626	065602	040504	040524	051040	EM3021: .ASCIZ /DATA READ INCORRECT/
9627	065610	040505	020104	047111	
9628	065616	047503	051122	041505	
9629	065624	000124			
9630	065626	051503	020061	047111	EM3022: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
9631	065634	047503	051122	041505	
9632	065642	020124	043101	042524	
9633	065650	020122	042522	042101	
9634	065656	047111	020107	040504	
9635	065664	040524	041040	043125	
9636	065672	042506	000122		
9637	065676	051503	020062	047111	EM3023: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
9638	065704	047503	051122	041505	
9639	065712	020124	043101	042524	
9640	065720	020122	042522	042101	
9641	065726	047111	020107	040504	
9642	065734	040524	041040	043125	
9643	065742	042506	000122		
9644	065746	051105	047522	020122	EM3024: .ASCIZ /ERROR REG INCORRECT/

9645	065754	042522	020107	047111	
9646	065762	047503	051122	041505	
9647	065770	000124			
9648	065772	040515	047111	020124	EM3025: .ASCIZ /MAINT REG 1 INCORRECT AFTER INDEX PULSE/
9649	066000	042522	020107	020061	
9650	066006	047111	047503	051122	
9651	066014	041505	020124	043101	
9652	066022	042524	020122	047111	
9653	066030	042504	020130	052520	
9654	066036	051514	000105		
9655	066042	051503	020061	047111	EM3026: .ASCIZ /CS1 INCORRECT AFTER COMMAND COMPLETION/
9656	066050	047503	051122	041505	
9657	066056	020124	043101	042524	
9658	066064	020122	047503	046515	
9659	066072	047101	020104	047503	
9660	066100	050115	042514	044524	
9661	066106	047117	000		
9662	066111	103	031123	044440	EM3027: .ASCIZ /CS2 INCORRECT AFTER COMMAND COMPLETION/
9663	066116	041516	051117	042522	
9664	066124	052103	040440	052106	
9665	066132	051105	041440	046517	
9666	066140	040515	042116	041440	
9667	066146	046517	046120	052105	
9668	066154	047511	000116		
9669	066160	051503	020061	047111	EM3028: .ASCIZ /CS1 INCORRECT AFTER READING EMPTY SILO/
9670	066166	047503	051122	041505	
9671	066174	020124	043101	042524	
9672	066202	020122	042522	042101	
9673	066210	047111	020107	046505	
9674	066216	052120	020131	044523	
9675	066224	047514	000		
9676	066227	103	031123	044440	EM3029: .ASCIZ /CS2 INCORRECT AFTER READING EMPTY SILO/
9677	066234	041516	051117	042522	
9678	066242	052103	040440	052106	
9679	066250	051105	051040	040505	
9680	066256	044504	043516	042440	
9681	066264	050115	054524	051440	
9682	066272	046111	000117		
9683	066276	040515	047111	020124	EM3030: .ASCIZ /MAINT REG 1 INCORRECT/
9684	066304	042522	020107	020061	
9685	066312	047111	047503	051122	
9686	066320	041505	000124		
9687	066324	051104	053111	020105	EM3031: .ASCIZ /DRIVE STATUS REG INCORRECT/
9688	066332	052123	052101	051525	
9689	066340	051040	043505	044440	
9690	066346	041516	051117	042522	
9691	066354	052103	000		
9692	066357	105	051122	051117	EM3032: .ASCIZ /ERROR REG INCORRECT/
9693	066364	051040	043505	044440	
9694	066372	041516	051117	042522	
9695	066400	052103	000		
9696	066403	115	030522	044440	EMW1: .ASCIZ /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
9697	066410	041516	051117	042522	
9698	066416	052103	047440	020116	
9699	066424	051461	020124	050125	
9700	066432	040527	042122	052040	

9701	066440	040522	051516	051511
9702	066446	044524	047117	047440
9703	066454	020106	040515	047111
9704	066462	020124	046103	041517
9705	066470	000113		
9706	066472	051115	020061	047111
9707	066500	047503	051122	041505
9708	066506	020124	047117	030440
9709	066514	052123	042040	053517
9710	066522	053516	051101	020104
9711	066530	051124	047101	044523
9712	066536	052123	047511	020116
9713	066544	043117	046440	044501
9714	066552	052116	041440	047514
9715	066560	045503	000	
9716	066563	115	030522	044440
9717	066570	041516	051117	042522
9718	066576	052103	047440	020116
9719	066604	047062	020104	050125
9720	066612	040527	042122	052040
9721	066620	040522	051516	051511
9722	066626	044524	047117	047440
9723	066634	020106	040515	047111
9724	066642	020124	046103	041517
9725	066650	000113		
9726	066652	051115	020061	047111
9727	066660	047503	051122	041505
9728	066666	020124	047117	031040
9729	066674	042116	042040	053517
9730	066702	053516	051101	020104
9731	066710	051124	047101	044523
9732	066716	052123	047511	020116
9733	066724	043117	046440	044501
9734	066732	052116	041440	047514
9735	066740	045503	000	

EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/

EMW3: .ASCIZ /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/

EMW4: .ASCIZ /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/

.SBTTL DATA PATTERNS

9736			
9737			
9738		066744	
9739	066744	000000	
9740	066746	177777	
9741	066750	125252	
9742	066752	052515	
9743	066754	101706	
9744	066756	060422	
9745	066760	130715	
9746	066762	177777	
9747	066764	000000	
9748	066766	177777	
9749	066770	000000	
9750	066772	177777	
9751	066774	000000	
9752	066776	125252	
9753	067000	052525	
9754	067002	125252	
9755	067004	052525	
9756	067006	125252	
9757	067010	052525	
9758	067012	044444	
9759	067014	022222	
9760	067016	111111	
9761	067020	052012	
9762	067022	100520	
9763	067024	052012	
9764	067026	155555	
9765	067030	066666	
9766	067032	155555	
9767	067034	104210	
9768	067036	104210	
9769	067040	104210	
9770	067042	100	100
9771	067044	101	101
9772	067046	102	102
9773	067050	103	103
9774	067052	104	104
9775	067054	105	105
9776	067056	106	106
9777	067060	107	107
9778	067062	110	110
9779	067064	111	111
9780	067066	112	112
9781	067070	113	113
9782	067072	114	114
9783	067074	115	115
9784	067076	116	116
9785	067100	117	117
9786	067102	120	120
9787	067104	121	121
9788	067106	122	122
9789	067110	123	123
9790	067112	124	124
9791	067114	125	125

.EVEN
 PATTERN: .WORD 000000
 .WORD 177777
 .WORD 125252
 .WORD 052515
 .WORD 101706
 .WORD 060422
 .WORD 130715
 HEAD1: .WORD 177777
 .WORD 000000
 .WORD 177777
 HEAD2: .WORD 000000
 .WORD 177777
 .WORD 000000
 HEAD3: .WORD 125252
 .WORD 052525
 .WORD 125252
 .WORD 052525
 .WORD 125252
 .WORD 052525
 HEAD4: .WORD 044444
 .WORD 022222
 .WORD 111111
 HEAD5: .WORD 052012
 .WORD 100520
 .WORD 052012
 HEAD6: .WORD 155555
 .WORD 066666
 .WORD 155555
 HEAD7: .WORD 104210
 .WORD 104210
 .WORD 104210
 NPRBUF: .BYTE 100,100
 .BYTE 101,101
 .BYTE 102,102
 .BYTE 103,103
 .BYTE 104,104
 .BYTE 105,105
 .BYTE 106,106
 .BYTE 107,107
 .BYTE 110,110
 .BYTE 111,111
 .BYTE 112,112
 .BYTE 113,113
 .BYTE 114,114
 .BYTE 115,115
 .BYTE 116,116
 .BYTE 117,117
 .BYTE 120,120
 .BYTE 121,121
 .BYTE 122,122
 .BYTE 123,123
 .BYTE 124,124
 .BYTE 125,125

9792	067116	126	126	.BYTE	126,126
9793	067120	127	127	.BYTE	127,127
9794	067122	130	130	.BYTE	130,130
9795	067124	131	131	.BYTE	131,131
9796	067126	132	132	.BYTE	132,132
9797	067130	133	133	.BYTE	133,133
9798	067132	134	134	.BYTE	134,134
9799	067134	135	135	.BYTE	135,135
9800	067136	136	136	.BYTE	136,136
9801	067140	137	137	.BYTE	137,137
9802	067142	140	140	.BYTE	140,140
9803	067144	141	141	.BYTE	141,141
9804	067146	142	142	.BYTE	142,142
9805	067150	143	143	.BYTE	143,143
9806	067152	144	144	.BYTE	144,144
9807	067154	145	145	.BYTE	145,145
9808	067156	146	146	.BYTE	146,146
9809	067160	147	147	.BYTE	147,147
9810	067162	150	150	.BYTE	150,150
9811	067164	151	151	.BYTE	151,151
9812	067166	152	152	.BYTE	152,152
9813	067170	153	153	.BYTE	153,153
9814	067172	154	154	.BYTE	154,154
9815	067174	155	155	.BYTE	155,155
9816	067176	156	156	.BYTE	156,156
9817	067200	157	157	.BYTE	157,157
9818	067202	160	160	.BYTE	160,160
9819	067204	161	161	.BYTE	161,161
9820	067206	162	162	.BYTE	162,162
9821	067210	163	163	.BYTE	163,163
9822	067212	164	164	.BYTE	164,164
9823	067214	165	165	.BYTE	165,165
9824	067216	166	166	.BYTE	166,166
9825	067220	167	167	.BYTE	167,167
9826	067222	170	170	.BYTE	170,170
9827	067224	171	171	.BYTE	171,171
9828	067226	172	172	.BYTE	172,172
9829	067230	173	173	.BYTE	173,173
9830	067232	174	174	.BYTE	174,174
9831	067234	175	175	.BYTE	175,175
9832	067236	176	176	.BYTE	176,176
9833	067240	177	177	.BYTE	177,177
9834	067242	200	200	.BYTE	200,200
9835	067244	201	201	.BYTE	201,201
9836	067246	202	202	.BYTE	202,202
9837	067250	203	203	.BYTE	203,203
9838	067252	204	204	.BYTE	204,204
9839	067254	205	205	.BYTE	205,205
9840	067256	206	206	.BYTE	206,206
9841	067260	207	207	.BYTE	207,207
9842	067262	210	210	.BYTE	210,210
9843	067264	000102		.BLKW	66.
9844		067300		BADPAR=	WRBUFF+14
9845		000001		.END	

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

E15

CZR6CCD RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 187
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0186

DH072B	056601	8694	8708	8979#										
DH072C	056637	8696	8985#											
DH072D	056716	8698	8993#											
DH150A	056773	8716	9001#											
DH150B	057027	8718	9006#											
DH164A	057065	8726	9012#											
DH164B	057131	8728	9019#											
DH170A	057177	8736	9026#											
DH170B	057276	8738	9037#											
DH171A	057374	8746	9048#											
DH171B	057413	8748	9051#											
DH175A	057431	8756	8766	9054#										
DH175B	057530	8758	8768	9065#										
DH211A	057625	8770	9076#											
DH211B	057653	8772	9080#											
DI =	040000	1099#												
DISPLA	001142	1260#	2310*	2318*	7594*	7726*								
DISPRE	000174	1189#	2318											
DLT =	100000	1118#												
DMD =	000040	1160#												
	2608	2609	2610	2613	2614	2625	2626	2627	2643	2697	2698	2699	2702	
	2703	2714	2716	2717	2720	2722	2778	2779	2780	2783	2784	2796	2854	
	2855	2856	2861	2862	2886	2889	2890	2891	2892	2920	2921	2922	2925	
	2946	2947	2948	2949	2976	2978	2979	2982	2985	2986	2993	3037	3046	
	3047	3051	3052	3053	3056	3058	3059	3126	3135	3136	3140	3141	3142	
	3145	3148	3149	3178	3179	3242	3250	3251	3255	3256	3257	3260	3263	
	3264	3298	3299	3371	3379	3380	3384	3385	3386	3389	3392	3393	3431	
	3432	3469	3470	3545	3554	3555	3559	3560	3561	3564	3567	3568	3600	
	3601	3663	3672	3673	3677	3678	3679	3682	3684	3685	3766	3771	3772	
	3776	3777	3778	3781	3783	3784	3825	3833	3834	3838	3839	3840	3843	
	3846	3847	3929	3934	3935	3939	3940	3941	3944	3946	3947	3988	3996	
	3997	4001	4002	4003	4006	4009	4010	4092	4097	4098	4102	4103	4104	
	4107	4109	4110	4151	4159	4160	4164	4165	4166	4169	4172	4173	4261	
	4269	4270	4274	4275	4276	4279	4281	4282	4327	4335	4336	4340	4341	
	4342	4345	4347	4348	4372	4373	4422	4431	4432	4436	4437	4438	4441	
	4445	4446	4478	4479	4559	4568	4569	4573	4574	4575	4578	4581	4582	
	4605	4606	4670	4673	4674	4677	4678	4780	4783	4784	4787	4788	4890	
	4893	4894	4897	4898	5000	5003	5004	5007	5008	5110	5113	5114	5117	
	5118	5219	5222	5223	5226	5227	5322	5325	5326	5329	5330	5386	5389	
	5390	5393	5394	5395	5396	5493	5494	5495	5501	5502	5506	5507	5508	
	5511	5512	5515	5516	5534	5541	5569	5575	5576	5580	5581	5582	5585	
	5587	5588	5591	5592	5595	5643	5645	5646	5649	5651	5656	5657	5689	
	5695	5696	5700	5701	5702	5705	5707	5708	5711	5712	5715	5763	5765	
	5766	5769	5771	5776	5777	5808	5814	5815	5819	5820	5821	5824	5826	
	5827	5830	5831	5834	5882	5884	5885	5888	5890	5895	5896	5927	5933	
	5934	5938	5939	5940	5943	5945	5946	5949	5950	5953	6001	6003	6004	
	6007	6009	6014	6015	6046	6052	6053	6057	6058	6059	6062	6064	6065	
	6068	6069	6072	6120	6122	6123	6126	6128	6133	6134	6165	6171	6172	
	6176	6177	6178	6181	6183	6184	6187	6188	6191	6239	6241	6242	6245	
	6247	6252	6253	6284	6290	6291	6295	6296	6297	6300	6302	6303	6306	
	6307	6310	6358	6360	6361	6364	6366	6371	6372	6410	6417	6418	6422	
	6423	6424	6428	6431	6432	6435	6436	6438	6488	6490	6491	6494	6496	
	6501	6502	6535	6542	6543	6547	6548	6549	6552	6554	6555	6560	6561	
	6563	6644	6646	6647	6650	6652	6658	6659	6681	6688	6689	6693	6694	
	6695	6698	6700	6701	6706	6707	6709	6790	6792	6793	6796	6798	6804	
	6805	6831	6838	6839	6843	6844	6845	6849	6852	6853	6856	6857	6859	

F15

		6909	6911	6912	6915	6917	6922	6923	6949	6953	6954	7026	7030	7031
		7103	7109	7110	7285	7315	7329	7344	7370	7382	7394	7419	7422	7423
		7424	7425	7428	7429	7430	7431							
DRA	= 000001	1141#	6969	7046	7125									
DRDY	= 000200	1148#												
DRDT	= 000040	1146#												
DRPAR	= 000010	1125#	7047											
DRVMSK	= 000007	1105#												
DSC	= 040000	1152#												
DSWR	= 177570	914#	1259	2309										
DTE	= 010000	1134#												
DYE	= 000040	1127#												
DT000	053210	1968	8513#											
DT001	053214	1353	1359	1365	1371	1377	1383	1389	1395	1401	1407	1413	1419	8514#
DT015	053234	1425	1431	2089	8517#									
DT017	053244	1437	1443	1449	8519#									
DT022	053262	1455	1461	1467	1479	1485	1491	8522#						
DT025	053302	1473	8525#											
DT031	053314	1497	1503	1509	1515	8527#								
DT035	053340	1521	1527	1533	1539	1563	1569	1575	1581	1587	1593	1599	1605	1629
		1635	1641	1647	1653	1659	1665	1671	1737	1743	1749	1755	1791	1797
		1803	1809	1833	1839	1845	1851	1857	1863	1869	1875	1899	1905	1911
		1917	1924	1931	1938	1945	8531#							
DT041	053364	1545	1761	1951	8535#									
DT042	053374	1551	1557	1881	1980	1986	2016	2022	8537#					
DT054	053410	1611	1677	2004	2040	8539#								
DT055	053422	1617	1623	1683	1689	1887	1893	1992	1998	2028	2034	8541#		
DT072	053440	1695	1701	1707	1713	1719	1767	1773	1779	1785	1815	8544#		
DT077	053470	1725	1731	1821	1827	1957	1963	8549#						
DT150	053504	1974	2010	8551#										
DT164	053520	2046	2052	2058	2064	8553#								
DT170	053536	2069	8556#											
DT171	053562	2074	2079	2084	8560#									
DT175	053572	2095	2101	2107	2113	2119	2125	2131	2137	2143	2149	2155	2161	8562#
DT211	053616	2167	2173	2179	2185	8566#								
ECCW	= 020000	1168#	2627	2643	2722	2796	2886	2993	5512	5595	5651	5715	5771	5834
		5890	5953	6009	6072	6128	6191	6247	6310	6366	6438	6496	6563	6626
		6632	6652	6709	6772	6778	6798	6859	6917					
ECH	= 000100	1128#												
EMTVEC	= 000030	1003#	2293*	2294*										
EMW	003170	2067#	5525*	5594*	5612*	5714*	5732*	5833*	5851*	5952*	5970*	6071*	6089*	6190*
		6208#	6309*	6327*	6437*	6455*	6562*	6580*	6600*	6613*	6708*	6726*	6746*	6759*
		6858#	6876*	7290*	7320*	7334*	7349*	7375*	7387*	7399*				
EMW1	066403	7290	9696#											
EMW2	066472	7320	7375	9706#										
EMW3	066563	7334	7387	9716#										
EMW4	066652	7349	7399	9726#										
EM000	057710	1966	9087#											
EM200	057755	1351	1357	1363	9094#									
EM201	060037	1369	1375	1381	9103#									
EM202	060122	1387	1393	1399	9112#									
EM203	060213	1405	1411	1417	9122#									
EM204	060305	1423	1429	1435	1441	1447	9132#							
EM205	060373	1453	1459	1465	1471	1477	1483	1489	1495	1501	1507	1513	9142#	
EM206	060461	1519	1525	1531	1537	1543	1549	1555	9152#					
EM207	060524	1561	1567	1573	1579	1609	1615	1621	9158#					

M15

CZR6CCD RK611 DSKLS CTRL PRT3
CZR6CC.P11 02-DEC-77 09:46

MACY11 30(1046) 02-DEC-77 09:56 PAGE 195
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0194

	4999*	5001*	5030	5038*	5051	5065	5109*	5111*	5140	5148*	5161	5175	5218*
	5220*	5249	5257*	5270	5284	5321*	5323*	5337	5352	5362*	5385*	5387*	5418
	5426*	5439	5453	5492*	5498*	5568*	5572*	5660	5688*	5692*	5780	5807*	5811*
	5899	5926*	5930*	6018	6045*	6049*	6137	6164*	6168*	6256	6283*	6287*	6375
	6409*	6414*	6505	6534*	6539*	6662	6680*	6685*	6808	6830*	6835*	6926	6951*
	6959	6963	6988*	6989	7028*	7036	7040	7065*	7066	7107*	7115	7119	7144*
	7145	7842*											
RKCS2 = 000010	1057*	2431*	2476*	2519*	2563*	3041*	3068	3088	3160	3183	3207	3275	3303
	3333	3404	3436	3461	3479	3514	3550*	3579	3605	3634	3689	3715	3793
	3858	3956	4021	4119	4184	4292	4312	4357	4377	4457	4485	4513	4591
	4610	4631	4722	4736	4832	4846	4942	4956	5052	5066	5162	5176	5271
	5285	5338	5353	5440	5454	6948*	6964	6990	7025*	7041	7067	7102*	7120
	7146												
RKDA = 000006	1056*	2430*	2475*	2518*	2562*	3039*							
RKDB = 000024	1062*	3082	3201	3322	3452	3503	3623	3809	3878	3972	4041	4135	4204
	4504	4625	4734	4844	4954	5064	5174	5283	5351	5452			
RKDCYL = 000020	1061*	2429*	2474*	2517*	2561*	3038*	6950*	7027*	7104*				
RKDS = 000012	1058*	6965	6991	7042	7068	7121	7147						
RKECPS = 000030	1066*												
RKECPT = 000032	1067*												
RKER = 000014	1059*	3692	4288	4380	6966	6992	7043	7069	7122	7148			
RKMR1 = 000026	1065*	2428*	2435*	2436*	2473*	2480*	2481*	2516*	2523*	2524*	2560*	2567*	2568*
	2608*	2609*	2610*	2613*	2614*	2625*	2626*	2633	2646	2658	2668	2698*	2699*
	2702*	2703*	2714*	2716*	2717*	2720*	2727	2739	2750	2779*	2780*	2783*	2784*
	2801	2813	2824	2854*	2855*	2856*	2861*	2862*	2889*	2890*	2891*	2892*	2893
	2920*	2921*	2922*	2925*	2946*	2947*	2948*	2949*	2950	2976*	2978*	2979*	2982*
	2985*	2986*	2989	3037*	3046*	3047*	3051*	3052*	3053*	3056*	3058*	3059*	3126*
	3135*	3136*	3140*	3141*	3142*	3145*	3148*	3149*	3178*	3179*	3242*	3250*	3251*
	3255*	3256*	3257*	3260*	3263*	3264*	3298*	3299*	3371*	3379*	3380*	3384*	3385*
	3386*	3389*	3392*	3393*	3431*	3432*	3469*	3470*	3545*	3554*	3555*	3559*	3560*
	3561*	3564*	3567*	3568*	3600*	3601*	3663*	3672*	3673*	3677*	3678*	3679*	3682*
	3684*	3685*	3766*	3771*	3772*	3776*	3777*	3781*	3783*	3784*	3825*	3825*	3833*
	3834*	3838*	3839*	3840*	3843*	3846*	3847*	3929*	3934*	3935*	3939*	3940*	3941*
	3944*	3946*	3947*	3988*	3996*	3997*	4001*	4002*	4003*	4006*	4009*	4010*	4092*
	4097*	4098*	4102*	4103*	4104*	4107*	4109*	4110*	4151*	4159*	4160*	4164*	4165*
	4166*	4169*	4172*	4173*	4261*	4269*	4270*	4274*	4275*	4276*	4279*	4281*	4282*
	4327*	4335*	4336*	4340*	4341*	4342*	4345*	4347*	4348*	4372*	4373*	4422*	4431*
	4432*	4436*	4437*	4438*	4441*	4445*	4446*	4478*	4479*	4559*	4568*	4569*	4573*
	4574*	4575*	4578*	4581*	4582*	4605*	4606*	4670*	4673*	4674*	4677*	4678*	4780*
	4783*	4784*	4787*	4788*	4890*	4893*	4894*	4897*	4898*	5000*	5003*	5004*	5007*
	5008*	5110*	5113*	5114*	5117*	5118*	5219*	5222*	5223*	5226*	5227*	5322*	5325*
	5326*	5329*	5330*	5386*	5389*	5390*	5393*	5394*	5395*	5396*	5493*	5494*	5495*
	5501*	5502*	5506*	5507*	5508*	5511*	5515*	5516*	5534*	5535*	5541*	5542*	5569*
	5575*	5576*	5580*	5581*	5582*	5585*	5587*	5588*	5591*	5592*	5643*	5645*	5646*
	5649*	5650	5656*	5657*	5689*	5695*	5696*	5700*	5701*	5702*	5705*	5707*	5708*
	5711*	5712*	5763*	5765*	5766*	5769*	5770	5776*	5777*	5808*	5814*	5815*	5819*
	5820*	5821*	5824*	5826*	5827*	5830*	5831*	5882*	5884*	5885*	5888*	5889	5895*
	5896*	5927*	5933*	5934*	5938*	5939*	5940*	5943*	5945*	5946*	5949*	5950*	6001*
	6003*	6004*	6007*	6008	6014*	6015*	6046*	6052*	6053*	6057*	6058*	6059*	6062*
	6064*	6065*	6068*	6069*	6120*	6122*	6123*	6126*	6127	6133*	6134*	6165*	6171*
	6172*	6176*	6177*	6178*	6181*	6183*	6184*	6187*	6188*	6239*	6241*	6242*	6245*
	6246	6252*	6253*	6284*	6290*	6291*	6295*	6296*	6297*	6300*	6302*	6303*	6306*
	6307*	6358*	6360*	6361*	6364*	6365	6371*	6372*	6410*	6417*	6418*	6422*	6423*
	6424*	6428*	6431*	6432*	6435*	6436*	6488*	6490*	6491*	6494*	6495	6501*	6502*
	6535*	6542*	6543*	6547*	6548*	6549*	6552*	6554*	6555*	6560*	6561*	6644*	6646*
	6647*	6650*	6651	6658*	6659*	6681*	6688*	6689*	6693*	6694*	6695*	6698*	6700*

\$TMP3 001166
\$TMP4 001170
\$TMP5 001172
\$TMP6 001174
\$TMP7 001176
\$TN = 000054

1272#
1273#
1274#
1275#
1276#
875#
2542 866 2412 2425# 2449 2454 2457 2470# 2494 2499 2502 2513# 2537
2545 2581 2566 2592 2605# 2623 2638 2651 2663 2673# 2678
2681 2694# 2712 2732 2744 2754 2759 2762 2775# 2793 2806 2818 2828
2833 2836 2851# 3010 3024# 3099 3101 3116# 3223# 3226 3238# 3347 3367#
3528 3541# 3648 3660# 3722 3725 3741# 3746 3888 3904# 3909 4051 4067#
4072 4214 4242# 4245 4407 4419# 4527# 4544# 4547 4647 4667# 4709 4743
4748 4757 4777# 4819 4853 4858 4867 4887# 4929 4963 4968 4977 4997#
5039 5073 5078 5087 5107# 5149 5183 5188 5197 5216# 5258 5292 5297
5306 5319# 5342 5347 5364 5383# 5427 5461 5466 5477 5490# 5544 5547
5566# 5663 5666 5686# 5783 5786 5805# 5902 5905 5924# 6021 6024 6043#
6140 6143 6162# 6259 6262 6281# 6378 6381 6407# 6508 6511 6532# 6665
6668 6678# 6811 6814 6828# 6929 6934 6946# 7011 7023# 7088 7100# 7559
7599

\$TPB 001152
\$TPFLG 001157
\$TPS 001150
\$TRAP 053120
\$TRAP2 053142
\$TRP = 000016

1264# 7923# 7934
1268# 7872 7934
1263# 7921 7934
2295 8473#
8484# 8495#
8488# 8497# 8498# 8499# 8500# 8501# 8502 8503# 8504 8505# 8506# 8507# 8508#
8509# 8510# 8511#
8478 8495#

\$TRPAD 053154
\$STSM 001004
\$STSTNM 001102
\$TTYIN 052456
\$STYPBN = ***** U
\$STYPDS 051046
\$TYPE 050336
\$TYPEC 050550
\$TYPEX 050616
\$TYPOC 050644
\$TYPON 050660
\$TYPOS 050620
\$UNIT 001226
\$UNITM 001010
\$USWR 001240
\$VECT1 001264
\$VECT2 001266
\$XTSTR 046656
\$\$GET4= 000000
\$\$SW08= 000054

1228#
1241# 7178# 7529 7561# 7588# 7589 7594 7598 7726 7762
8276 8277 8289 8307 8321 8325#
8501
8023# 8500
7682 7872# 8488 8496
7902 7909 7916 7921# 7922 8224
7927 7929 7932#
7964# 8497
7963 7966# 8499
7959# 8498
1294#
1230#
1301#
1326# 2364 2373* 2374* 2377 2397* 2398* 2399 2401
1327#
7543#
7206#
7598# 7599 7600# 7601# 7602# 7603# 7604# 7605# 7606# 7607# 7608# 7609# 7610#
7611# 7612# 7613# 7614# 7615# 7616# 7617# 7618# 7619# 7620# 7621# 7622# 7623#
7624# 7625# 7626# 7627# 7628# 7629# 7630# 7631# 7632# 7633# 7634# 7635# 7636#
7637# 7638# 7639# 7640# 7641# 7642#
7960# 7964# 7974 8009#
7540 7736
1184# 1188# 1194# 1200 1201# 1203# 1205# 1206# 1209# 1215 1216# 1218# 1220#
1238# 1283 2288 2303 2304 2347# 7191# 7214 7215# 7597 7598 7703# 7762
7934 8077# 8081 8085# 8086 8087# 8325# 8326 8333# 8387 8442 9738# 9843#
7655 7658
1215# 1220

\$.OFILL 051043
\$.4OCAT= ***** U
. = 067470

.\$ASTA= ***** U
.\$X = 001000

7540 7736
1184# 1188# 1194# 1200 1201# 1203# 1205# 1206# 1209# 1215 1216# 1218# 1220#
1238# 1283 2288 2303 2304 2347# 7191# 7214 7215# 7597 7598 7703# 7762
7934 8077# 8081 8085# 8086 8087# 8325# 8326 8333# 8387 8442 9738# 9843#
7655 7658
1215# 1220

BYPASS	1232#	3746	3909	4072	4245	4547									
CLRMSG	1232#	2514	2558												
CLRPSW	1232#														
COMMEN	1008#														
ENDCOM	1008#														
ERROR	902#	2447	2452	2455	2492	2497	2500	2535	2540	2543	2579	2584	2587	2621	2636
	2649	2661	2671	2679	2710	2730	2742	2760	2791	2804	2816	2834	2873	2878	2883
	2896	2906	2911	2916	2931	2936	2941	2953	2963	2968	2973	2996	2999	3002	3005
	3072	3075	3078	3081	3086	3092	3095	3165	3168	3171	3174	3188	3191	3194	3197
	3205	3213	3216	3278	3281	3284	3287	3308	3311	3314	3317	3326	3339	3342	3407
	3410	3413	3416	3441	3444	3447	3450	3456	3464	3467	3484	3487	3490	3493	3507
	3520	3523	3582	3585	3588	3591	3610	3613	3616	3619	3627	3640	3643	3700	3703
	3706	3709	3712	3720	3723	3799	3802	3805	3808	3813	3862	3865	3868	3871	3882
	3962	3965	3968	3971	3976	4025	4028	4031	4034	4045	4125	4128	4131	4134	4139
	4188	4191	4194	4197	4208	4297	4300	4303	4306	4309	4317	4320	4361	4364	4367
	4370	4388	4391	4394	4397	4400	4463	4466	4469	4472	4490	4493	4496	4499	4508
	4519	4522	4594	4597	4600	4603	4615	4618	4621	4624	4629	4635	4638	4707	4727
	4730	4742	4747	4752	4817	4837	4840	4852	4857	4862	4927	4947	4950	4962	4967
	4972	5037	5057	5060	5072	5077	5082	5147	5167	5170	5182	5187	5192	5256	5276
	5279	5291	5296	5301	5341	5346	5358	5361	5425	5445	5448	5460	5465	5470	5528
	5538	5545	5604	5610	5627	5639	5654	5664	5724	5730	5747	5759	5774	5784	5843
	5849	5866	5878	5893	5903	5962	5968	5985	5997	6012	6022	6081	6087	6104	6116
	6131	6141	6200	6206	6223	6235	6250	6260	6319	6325	6342	6354	6369	6379	6447
	6453	6469	6481	6499	6509	6572	6578	6594	6607	6612	6621	6628	6635	6656	6666
	6718	6724	6740	6753	6758	6767	6774	6781	6802	6812	6868	6874	6890	6902	6920
	6930	6973	6976	6979	6982	6999	7002	7005	7008	7050	7053	7056	7059	7076	7079
	7082	7085	7129	7132	7135	7138	7155	7158	7161	7164	7438				
ESCAPE	1008#														
FORERR	1232#	6947	7024	7101											
GETPRI	1008#	7471													
GETSWR	1008#	2334#													
LDLPER	1232#	3031	3121	3761	3820	3924	3983	4087	4146	4256	4322				
MSG	2412#	2414	2457#	2459	2502#	2504	2545#	2547	2592#	2594	2681#	2683	2762#	2764	2836#
	2838	3010#	3012	3101#	3103	3226#	3228	3347#	3349	3528#	3530	3648#	3650	3725#	3727
	3888#	3890	4051#	4053	4214#	4216	4407#	4409	4527#	4529	4647#	4649	4757#	4759	4867#
	4869	4977#	4979	5087#	5089	5197#	5199	5306#	5308	5364#	5366	5477#	5479	5547#	5549
	5666#	5668	5786#	5788	5905#	5907	6024#	6026	6143#	6145	6262#	6264	6381#	6383	6511#
	6513	6668#	6670	6814#	6816	6934#	6936	7011#	7013	7088#	7090				
MULT	1008#														
NEWTST	1008#	2412	2457	2502	2545	2592	2681	2762	2836	3010	3101	3226	3347	3528	3648
	3725	3888	4051	4214	4407	4527	4647	4757	4867	4977	5087	5197	5306	5364	5477
	5547	5666	5786	5905	6024	6143	6262	6381	6511	6668	6814	6934	7011	7088	
POP	1008#	7697	7698	8064	8376	8425									
PUSH	1008#	7658	7660	7681	8023	8350	8405								
RDLTOP	1232#	4668	4778	4888	4998	5108	5384								
REPORT	1008#														
SCOPE	903#	2424	2469	2512	2556	2604	2693	2774	2850	3023	3115	3237	3366	3540	3659
	3740	3903	4066	4241	4418	4543	4666	4776	4886	4946	5106	5215	5318	5382	5489
	5565	5685	5804	5923	6042	6161	6280	6406	6531	6677	6827	6945	7022	7099	7177
SEKMSG	1232#	2426	2471												
SETPRI	1008#	8254													
SETTRA	8488#	8497	8498	8499	8500	8502	8504	8505	8506	8507	8508	8509	8510		
SETUP	1008#	2282													
SKIP	1008#	2449	2454	2494	2499	2537	2542	2581	2586	2623	2638	2651	2663	2673	2678
	2712	2732	2744	2754	2759	2793	2806	2818	2828	2833	3099	3223	3722	4709	4743
	4748	4819	4853	4858	4929	4963	4968	5039	5073	5078	5149	5183	5188	5258	5292

.SREAD	874#	8078
.SSAVE	874#	8387
.SSCOP	874#	7524
.SSIZE	874#	7447
.STRAP	874#	8465
.STYPD	874#	8011
.STYPE	874#	7855
.STYPO	874#	7934

. ABS. 067470 000

ERRORS DETECTED: 0

RM03:CZR6CC, RM03:CZR6CC, SEQ/SOL/CRF/NL: TOC/DOC=RM03:CZR6CC.P11
RUN-TIME: 32 29 3 SECONDS
RUN-TIME RATIO: 1341/65=20.5
CORE USED: 35K (69 PAGES)

DOCUMENT PAGES: 205