

RK611
RK06, RK07

RK6 FCTNL CTRL
CZR6KGO

AH-9130G-MC
FICHE 1 OF 2

MAR 1982
COPYRIGHT © 76-81
MADE IN USA



RK611
RK06, RK07

RK6 FCTNL CTRL
CZR6KGO

AH-9130G-MC
FICHE 2 OF 2

MAR 1982
COPYRIGHT © 76-81
MADE IN USA



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9128G-MC
PRODUCT NAME: CZR6KGO RK611 FCTNL CTRL
DATE: AUGUST 10 1981
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: BRIAN LEBLANC

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

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

COPYRIGHT (C) 1976,1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:
DIGITAL PDP UNIBUS MASSBUS
DEC DECUS DECTAPE

46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	HARDWARE REQUIREMENTS
2.2	PRELIMINARY PROGRAMS
3.0	OPERATING PROCEDURE
3.1	LOADING PROCEDURE
3.2	STARTUP PROCEDURE
3.3	CONSOLE SWITCH REGISTER
3.4	SOFTWARE SWITCH REGISTERS
3.5	CONTROL C (^C) OPERATION
3.6	CONTROL S (^S) OPERATION
3.7	CONTROL Q (^Q) OPERATION
3.8	UNIBUS ADDRESS
3.9	EXECUTION TIM
4.0	PROGRAM DESCRIPTION
5.0	ERROR REPORTING
6.0	SUBROUTINES
7.0	REVISION HISTORY

E

91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146

1.0 ABSTRACT

THE RK611 FUNCTIONAL CONTROLLER DIAGNOSTIC (CZR6K) IS A SERIES OF TESTS THAT COMPLETES THE TESTING OF AN RK611/RK06-RK07 SUBSYSTEM. THE DISKLESS CONTROLLER DIAGNOSTIC AND THE RK06-RK07 DRIVE DIAGNOSTIC ARE PREREQUISITES TO THE RUNNING OF THIS PROGRAM. THE PURPOSE OF THIS PROGRAM IS TO TEST THOSE AREAS IN THE CONTROLLER THAT COULD NOT BE TESTED IN A DISKLESS ENVIRONMENT AND THOSE AREAS OF THE DRIVE THAT COULD NOT BE TESTED UNTIL CONTROLLER OPERATION IN A DIAGNOSTIC OR MAINTENANCE MODE HAS BEEN TESTED.

THE TESTS PERFORMED ARE MAINLY FUNCTIONALLY ORIENTED BUT DIAGNOSTIC MODE IS USED IN NUMEROUS OCCASSIONS TO ACCOMPLISH THE OBJECTIVES, MAINLY THE FORCING OF ERRORS. IN THESE CASES, THE CONTROLLER IS PLACED IN DIAGNOSTIC MODE AND OPERATION IS CLOCKED PART WAY THROUGH. DIAGNOSTIC MODE IS THEN RESET AND THE CONTROLLER ALLOWED TO COMPLETE THE OPERATION. DEPENDING ON THE OPERATION AND HOW FAR THROUGH IT BEFORE DIAGNOSTIC MODE IS RESET VARIOUS ERROR CONDITIONS CAN BE MADE TO OCCUR. THIS DOCUMENT DOES NOT ATTEMPT TO EXPLAIN WHY THESE ERROR CONDITIONS ARE SET BUT THE INDIVIDUAL TEST DESCRIPTIONS SPECIFY WHAT ERROR IS BEING FORCED AND THE PROCEDURE USED TO FORCE IT.

C A U T I O N

THIS PROGRAM SHOULD BE HALTED ONLY BY TYPING A ^C. IF THE PROGRAM IS HALTED USING THE HALT KEY THE POSSIBILITY EXISTS THAT THE CARTRIDGE FORMAT WILL BE INCORRECT, THE CYLINDER ADDRESS IN THE DRIVE MAY BE INVALID, OR THE DRIVE MAY NOT BE READY.

2.0 REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

PDP-11 SYSTEM (16K MEMORY)
CONSOLE TERMINAL
DECTAPE, PAPERTAPE, OR DISK
LINE CLOCK (KW11-L) (OPTIONAL)
PARITY OPTION (MM11) (OPTIONAL)
RK611 CONTROLLER
AT LEAST 1 AND UP TO 8 RK06-RK07 DRIVES
FORMATTED RK06K ON EACH DRIVE

2.2 PRELIMINARY PROGRAMS

147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC (ALL PARTS) AND THE

UNIBUS RK06-RK07 DRIVE DIAGNOSTIC (ALL PARTS) SHOULD HAVE RUN SUCCESSFULLY.

3.0 OPERATING PROCEDURE

3.1 LOADING PROCEDURE

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

IT CAN BE LOADED AND RUN UNDER APT OR ACT AND IT CAN BE CHAINED BY XXDP.

3.2 STARTUP PROCEDURE

THE PROGRAM START LOCATION IS 200(8). THIS START WILL AUTOMATICALLY SIZE THE SYSTEM UNLESS RUNNING UNDER APT. THE PROGRAM ASSUMES THE STANDARD UNIBUS ADDRESS, VECTOR ADDRESS, AND BUS PRIORITY LEVEL (177440, 210, AND 4 RESPECTIVELY). IF STARTED AT 200 AND THE XXDP MEDIA IS RK06 (PROGRAM LOADED FROM RK06) DRIVE 0 IS NOT TESTED.

LOCATION 204(8) IS THE PROGRAM RESTART.

LOCATION 214(8) IS THE PARAMETERIZATION START LOCATION. THE OPERATOR WILL BE ASKED TO IDENTIFY THE BUS ADDRESS, VECTOR ADDRESS, AND BUS PRIORITY. IF THE PROGRAM WAS LOADED FROM RK06, THE OPERATOR WILL BE ASKED TO MOUNT A WORK CARTRIDGE ON DRIVE 0 OR TO PLACE IT OFF-LINE IF IT IS NOT TO BE TESTED.

LOCATION 220(8) IS THE PHASE LOCKED LOOP CLOCK ADJUSTMENT START. THE PROGRAM FIRST RUNS THE FIRST THREE TESTS AND THEN JUMPS TO THE ADJUSTMENT ROUTINE. THE PROGRAM WILL CONTINUE TO LOOP IN THIS ROUTINE UNTIL THE PROCESSOR IS HALTED.

ALL DRIVES THAT ARE TO BE TESTED MUST BE ON-LINE, READY, AND WRITE LOCK RESET. IF ALL THREE CONDITIONS ARE NOT MET, THAT DRIVE IS NOT TESTED.

3.3 CONSOLE SWITCH REGISTER

THE CONSOLE SWITCH REGISTER IS USED TO PROVIDE PROGRAM CONTROL AS DESCRIBED BELOW:

- SW15 - HALT ON ERROR
- SW14 - LOOP ON TEST
- SW13 - INHIBIT ERROR REPORT
- SW12 - ABORT PROGRAM AFTER 20 ERRORS
- SW11 - INHIBIT ITERATIONS
- SW10 - BELL ON ERROR
- SW09 - LOOP ON ERROR
- SW08 - EXECUTE TEST NUMBER SPECIFIED IN SW<7-0>.

SW<7-0> - EXECUTE THIS TEST IF SW08 SET.

EXECUTING A SPECIFIC TEST MUST BE USED WITH CAUTION. SOME TESTS REQUIRE OTHERS TO BE RUN TO FORMAT THE PACK IN A SPECIFIC MANNER OR WRITE SPECIFIC DATA. TESTS THAT REQUIRE OTHERS TO BE RUN INDICATE THIS IN THE TEST DESCRIPTION. IT IS SUGGESTED THAT THE PROGRAM BE RUN IN THE DEFAULT SEQUENCE THE FIRST TIME AFTER IT HAS BEEN LOADED.

NOTE: TEST 3 MUST BE RUN BEFORE ANY SUBSEQUENT TEST. THIS TEST DETERMINES WHICH DRIVES ARE ON THE DRIVE BUS FOR ALL FOLLOWING TESTS. LIKEWISE, TEST 20 MUST BE RUN BEFORE ANY TEST SUBSEQUENT TO IT. THIS TEST READS THE BAD SECTOR FILES AND BUILDS TABLES USED BY THE FOLLOWING TESTS. THESE TESTS, HAVING BEEN RUN ONCE, NEED NOT BE RUN AGAIN IF A DIFFERENT TEST IS SELECTED.

3.4 '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.

3.5 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.

3.6 CONTROL S (^S) OPERATION

IF ^S IS TYPED AT ANY TIME THE PROGRAM WILL GO INTO A STALL

203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258

259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314

LOOP UNTIL A CONTROL Q (^Q) IS TYPED.

3.7 CONTROL Q (^Q) OPERATION

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

3.8 UNIBUS ADDRESSES

STANDARD UNIBUS ADDRESSES ARE ASSUMED FOR THE KW11-L AND MM11 OPTIONS. THESE ADDRESSES MAY BE CHANGED BY CHANGE THE APPROPRIATE MEMORY LOCATIONS. THE FOLLOWING TAGS AND LOCATIONS HAVE BEEN USED:

KW11-L	TAG	LOCATION
UNIBUS ADDRESS	KWLADD	1710
VECTOR ADDRESS	KWLVEC	1712

3.9 EXECUTION TIME

THE FIRST PASS OF THE PROGRAM FOR ONE DRIVE IS APPROXIMATELY 65 SECONDS AND EACH SUBSEQUENT PASS IS APPROXIMATELY 2 MINUTES 20 SECONDS.

THE EXECUTION TIME FOR MULTIPLE DRIVES IN THE FIRST PASS IS:

$$((\text{NUMBER OF DRIVES}) \times 45 \text{ SEC}) + 20 \text{ SEC}$$

FOR SUBSEQUENT PASSES THE RUN TIME IS THE PRODUCT OF 2 MINUTES 20 SECONDS TIMES THE NUMBER OF DRIVES PLUS 25 SECONDS FOR EACH DRIVE AFTER THE FIRST.

4.0 PROGRAM DESCRIPTION

THE FOLLOWING TEST SEQUENCE IS EXECUTED ASSUMING TWO OR MORE DRIVES.

FIRST PASS - FIRST DRIVE:
ALL TESTS UP TO THE MULTI-DRIVE OPERATIONS ARE PERFORMED ONCE.

FIRST PASS - ALL REMAINING DRIVES:
STATUS VALID TESTS UP TO THE MULTI-DRIVE OPERATIONS ARE PERFORMED ONCE ON EACH DRIVE.

THEN MULTI-DRIVE OPERATIONS ARE PERFORMED ONCE ON EACH COMBINATION OF DRIVES.

SECOND AND ALL SUBSEQUENT PASSES:
THE SAME SEQUENCE OF TESTING IS REPEATED EXCEPT FOR TEST ITERATIONS WHICH ARE SPECIFIED FOR EACH TEST.

315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370

**BASIC INTERFACE AND OPTION TESTS

TEST 1 RK611 BASE ADDRESS TEST

CHECK THAT READING THE RK611 BASE ADDRESS (RKCS1) DOES NOT CAUSE A NON-EXISTANT MEMORY TRAP.

TEST 2 INTERRUPT VECTOR ADDRESS TEST CHECK THAT THE INTERRUPT VECTOR FOR THE RK611 IS SET TO THE EXPECTED ADDRESS.

**STATUS VALID TESTS

TEST 3 SELECT ALL DRIVES

IF NOT RUNNING IN APT AUTOMATIC ENVIRONMENT, DETERMINE WHAT DRIVES ARE ON-LINE BY SELECTING ALL DRIVES. IF NON-EXISTENT DRIVE REPORTED MAKE SURE STATUS VALID IS RESET. IF DRIVE PRESENT MAKE SURE NO ERROR EXISTS, DRIVE IS CYCLED UP, AND STATUS VALID SET, AND DSC RESET.

IF RUNNING IN APT AUTOMATIC ENVIRONMENT, THE DRIVES IDENTIFIED IN ETABLE ARE TESTED FOR NO ERROR, DRIVE CYCLED UP, AND STATUS VALID SET.

IF LOCATION 41 INDICATES THE XXDP MEDIA IS ON THE RK06, DRIVE 0 WILL ONLY BE TESTED IF THE PARAM START (214) WAS USED. IF THE AUTOMATIC START (200) IS USED, DRIVE 0 IS NOT TESTED. THE RESTART (204) WILL RETAIN THE TEST STATUS OF DRIVE 0.

IF THE PARAM START IS USED, THE OPERATOR MUST EITHER PLACE DRIVE 0 OFF LINE IF IT IS NOT TO BE TESTED OR UNLOADED AND A SCRATCH MEDIA MOUNTED IF IT IS TO BE TESTED. THE PROGRAM WILL MONITOR OFF LINE AND VOLUME VALID TO DETERMINE THE TEST STATUS OF DRIVE 0.

ALL DRIVES TO BE TESTED MUST BE ON-LINE, CYCLED UP, AND WRITE LOCK RESET. ADDRESSES OF DRIVES THAT ARE NON-EXISTANT EITHER BECAUSE THE DRIVE DOES NOT EXIST OR IS OFF-LINE ARE USED TO VERIFY NON-EXISTANT DRIVE ERROR DETECTION. DRIVES THAT ARE ON-LINE BUT NOT CYCLED UP OR ARE WRITE LOCKED ARE NOT TESTED.

AT COMPLETION OF THE TEST A MESSAGE WILL BE GIVEN TO IDENTIFY THE DRIVES TO BE USED IN TESTING.

NOTE: THIS TEST MUST BE RUN AT LEAST ONCE BEFORE ANY OTHER TEST THAT FOLLOWS.
THE DETERMINATION OF WHETHER EACH DRIVE IS AN RK06 OR RK07 IS MADE IN THIS TEST, AND

A TABLE IS FILLED ACCORDINGLY.

TEST 4 RELEASE ALL DRIVES

RELEASE ALL DRIVES. MAKE SURE NO ERROR SETS AND STATUS VALID IS RESET.

TEST 5 NON-STANDARD MESSAGES AND SVAL

PICK ONE OF THE AVAILABLE DRIVES AND GET NON-STANDARD MESSAGES. MAKE SURE NO ERROR OCCURS AND STATUS VALID DOES NOT SET AND THAT NON-STANDARD MESSAGES CAUSE STATUS VALID TO RESET.

TEST 6 WRITING CS2 AND STATUS VALID

SELECT AN AVAILABLE DRIVE. MAKE SURE STATUS VALID IS SET. WRITE COMMAND AND STATUS REGISTER 2. MAKE SURE STATUS VALID RESETS.

**CONTROLLER ERROR TESTS

TEST 7 DRIVE TYPE ERROR

CREATE A DRIVE TYPE ERROR MAKE SURE DRIVE TYPE ERROR SETS AND STATUS VALID SETS.

TEST 10 STATUS VALID AND PARITY ERROR

ISSUE A SELECT TO AN AVAILIABLE DRIVE WITH BAD PARITY. MAKE SURE SPAR, CONTROLLER ERROR, ATTENTION, DRIVE STATUS CHANGES, DRPAR, DRIVE INTERRUPT, AND STATUS VALID SET. ISSUE A CONTROLLER CLEAR. MAKE SURE DRIVE INTERRUPT AND ATTENTION ARE STILL SET. SELECT DRIVE AGAIN WITH GOOD PARITY. MAKE SURE ATTENTION, DRIVE STATUS CHANGE, DRPAR, CONTROLLER ERROR, DRIVE INTERRUPT, AND STATUS VALID ARE SET AND SPAR IS RESET. ISSUE A CONTROLLER CLEAR. GET NON-STANDARD MESSAGES AND MAKE SURE ONLY DRIVE INTERRUPT AND ATTENTION ARE SET. CLEAR ATTENTION WITH DRIVE CLEAR. REPEAT FOR ALL AVAILIABLE DRIVES.

TEST 11 UNIT FIELD ERROR ON RELEASE

ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE DRIVE. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A RELEASE COMMAND. CLOCK THROUGH PHASE ADDRESS 2. TURN OFF DIAGNOSTIC MODE. MAKE SURE UNIT FIELD ERROR SETS.

TEST 12 UNIT FIELD ERROR ON SELECT

ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE DRIVE.

371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426

427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482

PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A SELECT
COMMAND WITH MESSAGE ID = 3 AND DRIVE SELECTED = 0.
CLOCK THROUGH PHASE ADDRESS 6. TURN OFF DIAGNOSTIC

MODE. MAKE SURE UNIT FIELD ERROR SETS.

**ATTENTION HANDLING BY CONTROLLER

TEST 13 DOUBLE INTERRUPT

ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. MAKE
SURE STATUS VALID IS SET. CHECK THAT SECOND INTERRUPT
OCCURS. AFTER SECOND INTERRUPT CHECK THAT STATUS
VALID IS RESET. ISSUE SELECT AND MAKE SURE STATUS
VALID SETS. CLEAR DRIVE. CHECK THAT DRIVE STATUS
CHANGE SETS (BIT 14 OF DRIVE STATUS REGISTER)

TEST 14 SINGLE INTERRUPT FROM ATTENTION

DO A SEEK TO CYLINDER 0. WAIT FOR INTERRUPT FROM
DRIVE ATTENTION. LOWER PRIORITY AGAIN AND MAKE SURE
ANOTHER INTERRUPT DOES NOT OCCUR. CLEAR DRIVE.

TEST 15 RESET ATTENTIONS WITH UNIBUS INIT

DO A SEEK TO CYLINDER 0 ON ALL AVAILIABLE DRIVES.
ISSUE A RESET. MAKE SURE ALL ATTENTION RESET.

**ILLEGAL DISK ADDRESS ERROR TESTS

TEST 16 ILLEGAL DISK ADDRESS (PART 1)

ISSUE A SEEK TO CYLINDER 0, HEAD 3. MAKE SURE ILLEGAL
ADDRESS ERROR AND SEEK INCOMPLETE SETS. CLEAR
CONTROLLER AND CLEAR DRIVE. REPEAT FOR HEADS 4-7,
CHECKING THAT BOTH IDAE AND SEEK INCOMPLETE SET FOR
HEAD 7 AND IDAE ALONE SETS FOR HEADS 4, 5, AND 6.

TEST 17 ILLEGAL DISK ADDRESS (PART 2)

ISSUE A SEEK TO CYLINDER 1000, HEAD 0. MAKE SURE
ILLEGAL DISK ADDRESS ERROR SETS. CLEAR CONTROLLER AND
DRIVE
THIS TEST IS BYPASSED BY THE RK07 CONTROLLER BECAUSE
IT DOES NOT RECOGNIZE ANY ILLEGAL DISK ADDRESS.

**WRITE HEADER TESTS

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

TEST 20 READ BAD SECTOR INFORMATION

ISSUE A READ DATA OF 400 WORDS TO CYLINDER 632, TRACK 2 TO GET THE FACTORY DETECTED BAD SECTOR FILE, 26 SECTOR MODE.
CYLINDER 1456 IS USED FOR THE RK07.

IF AN ERROR OCCURS, READ SECTOR 2, 4, 6, OR 10(8) UNTIL A SUCCESSFUL READ IS DONE. IF NONE READ SUCCESSFULLY REMOVE THIS DRIVE FROM TEST. WHEN A READ IS SUCCESSFUL, TEST THAT THE PACK IS NOT AN ALIGNMENT PADK AND STORE THE ENTRIES FOR LATER USE.

REPEAT THIS SERIES OF OPERATIONS FOR FACTORY DETECTED BAD SECTORS 24 SECTOR MODE, SOFTWARE DETECTED BAD SECTORS 26 SECTOR MODE, AND SOFTWARE DETECTED BAD SECTORS 24 SECTOR MODE. IF THE NUMBER OF BAD SECTORS FOR 24 OR 26 SECTOR MODE EXCEED 20(10) THE DRIVE IS REMOVED FROM TESTING.

NOTE: THIS TEST IS RUN IN THE FIRST (QUICK VERIFY) PASS ONLY.

TEST 21 FORMAT IN 26 SECTOR FORMAT

FORMAT CYLINDER 312, TRACK 0 AND TRACK 1 FOR 26 SECTOR FORMAT. VERIFY FORMAT AND THAT DATA LATE DID NOT OCCUR WITH WRITE HEADER ON IN READING DATA BUFFER AFTER READ HEADER.

**HEADER RECOGNITION TESTS

TEST 22 BAD SECTOR ERROR

FORMAT CYLINDER 312, TRACK 0, ON SCRATCH PACK TO HAVE SECTOR 0 (BIT 15 OR WORD 2 OF HEADER RESET) AND SECTOR 1 (BIT 14 OR WORD 2 OF HEADER RESET) TO BE BAD SECTORS AND ALL OTHER SECTORS GOOD.

ISSUE A WRITE DATA OR 400 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS. ISSUE A WRITE DATA TO CYLINDER 0, TRACK 0, SECTOR 1 OF 400 WORDS. MAKE SURE BAD SECTOR ERROR SET. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 0, TRACK 0, SECTOR 2. MAKE SURE NO ERROR SETS.

TEST 23 HEADER VRC ERROR

FORMAT CYLINDER 312, TRACK 0, ON SCRATCH PACK TO HAVE 16 SECTORS WITH BAD HEADER VRC. ISSUE A WRITE DATA OF EACH OF THE SECTORS WITH A BAD HEADER VRC. MAKE SURE HEADER VRC ERROR SETS. ISSUE A WRITE DATA TO A GOOD

539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594

HEADER AND MAKE SURE NO ERROR OCCURS.

TEST 24 BAD SECTOR ERROR AND HVRC ERROR

FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR ZERO HAS BOTH A BAD SECTOR ERROR AND HEADER VRC. ISSUE A WRITE

DATA TO CYLINDER 0, TRACK 0, SECTOR 0. MAKE SURE ONLY HEADER VRC ERROR SETS.

TEST 25 OPERATION INCOMPLETE

FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR 21 HAS THE WRONG FORMAT. ISSUE A WRITE DATA OF 400 TO CYLINDER 0, TRACK 0, SECTOR 21. MAKE SURE OPI SET.

TEST 26 OPI WITH HVRC ERROR

FORMAT CYLINDER 312, TRACK 0 SUCH THAT A HEADER VRC ERROR IS PRESENT AND SECTOR 17 HAS THE WRONG FORMAT. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 312, TRACK 0, SECTOR 17. THAT BOTH OPERATION INCOMPLETE AND HEADER VRC SET.

TEST 27 HVRC IGNORE ON NON-ADDRESSED SECTOR

FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR 20 HAS AN HVRC ERROR. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 312, TRACK 0, AND SECTOR 21. MAKE SURE HVRC IS NOT SET AT THE END OF THE OPERATION.

**DATA TRANSFER TESTS

TEST 30 WRITE AND READ ONE SECTOR

FORMAT CYLINDER 312, ALL TRACKS AND CYLINDER 313, TRACK 2 TO AGREE WITH BAD SECTOR INFORMATION. ISSUE A WRITE DATA OF ONE SECTOR ON CYLINDER 312, TRACK 0. READ IT BACK TO MAKE SURE IT AGREES WITH WHAT IS WRITTEN.

TEST 31 WRITE DATA WITH BUS ADDRESS INCREMENT INHIBIT

ISSUE A WRITE DATA OF ONE SECTOR TO CYLINDER 312, TRACK 2, SECTOR 12 WITH INHIBIT BUS ADDRESS INCREMENT. READ DATA BACK TO MAKE SURE EVERY WORD IS THE SAME AND CORRECT.

TEST 32 WRITE DATA ADDRESS GREATER THAN 32K

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

ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 177770.
MAKE SURE CORRECT DATA IS ON DISK.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K OF
MEMORY IS PRESENT.

TEST 33 READ DATA ADDRESS GREATER THAN 32K

ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 177770.
CHECK MEMORY FOR CORRECT TRANSFER.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K OF
MEMORY IS PRESENT.

TEST 34 WRITE DATA ADDRESS GREATER THAN 64K

ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 377770.
MAKE SURE CORRECT DATA IS ON DISK.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K OF
MEMORY IS PRESENT.

TEST 35 READ DATA ADDRESS GREATER THAN 64K

ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 377770.
CHECK MEMORY FOR CORRECT TRANSFER.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K OF
MEMORY IS PRESENT.

TEST 36 WRITE DATA ADDRESS GREATER THAN 96K

ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 577770.
MAKE SURE CORRECT DATA IS ON DISK.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K OF
MEMORY IS PRESENT.

TEST 37 READ DATA ADDRESS GREATER THAN 96K

ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 577770.
CHECK MEMORY FOR CORRECT TRANSFER.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K OF
MEMORY IS PRESENT.

TEST 40 PARTIAL SECTOR WRITE DATA

ISSUE A WRITE DATA OF 103 WORDS TO CYLINDER 312, HEAD
0, SECTOR 0 MAKE SURE THE SECTOR WAS ZERO FILLED
CORRECTLY.

TEST 41 PARTIAL SECTOR READ DATA

651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706

WRITE CYLINDER 312, TRACK 0, SECTOR ZERO WITH A KNOWN CONFIGURATION. ISSUE A READ DATA OF 103 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE ONLY 103 WORDS GET TRANSFERRED TO MEMORY.

TEST 42 WRITE DATA WITH NON-EXISTENT MEMORY

ISSUE A WRITE DATA OF 1 WORD USING ADDRESS 776000. MAKE SURE NON-EXISTENT MEMORY SETS.

TEST 43 READ DATA WITH NON-EXISTENT MEMORY

ISSUE A READ DATA OF 1 WORD USING ADDRESS 776000. MAKE SURE NON-EXISTENT MEMORY SETS.

TEST 44 UNIBUS PARITY ERROR

INITIALIZE A MEMORY LOCATION WITH BAD PARITY. ISSUE A WRITE DATA OF 400 WORDS STARTING AT A LOCATION 112 WORDS BEFORE THE LOCATION WITH BAD PARITY. MAKE SURE THAT UNIBUS PARITY ERROR SETS.

NOTE: THIS TEST IS ONLY EXECUTED IF MEMORY PARITY OPTION EXISTS FOR BUFFER. IT WILL NOT BE EXECUTED ON 'ECC' TYPE MEMORY.

**MULTIPLE SECTOR OPERATIONS

TEST 45 TWO SECTOR WRITE DATA (PART 1)

ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0. READ DATA BACK ONE SECTOR AT A TIME AND MAKE SURE IT IS CORRECT.

TEST 46 TWO SECTOR WRITE DATA (PART 2)

ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 23. READ DATA BACK ONE SECTOR AT A TIME AND MAKE SURE A MID-TRANSFER SEEK DID NOT TAKE PLACE.

TEST 47 TWO SECTOR WRITE DATA (PART 3)

ISSUE A WRITE DATA OF 401 WORDS TO CYLINDER 312, TRACK 0, SECTOR 10. READ DATA BACK ONE SECTOR AT A TIME AND CHECK ZERO FILL OF SECOND SECTOR.

TEST 50 MID-TRANSFER SEEK ON WRITE (PART 1)

ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312.

707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762

TRACK 0, SECTOR 25. READ DATA BACK ONE SECTOR AT A TIME AND MAKE SURE A MID-TRANSFER SEEK DID TAKE PLACE.

TEST 51 MID-TRANSFER SEEK ON WRITE (PART 2)

ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312, TRACK 2, SECTOR 25. READ DATA BACK ONE SECTOR AT A TIME AND MAKE SURE A MID-TRANSFER SEEK DID TAKE PLACE.

TEST 52 TWO SECTOR READ DATA (PART 1)

ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0, VERIFY THAT CORRECT DATA IS READ.

NOTE: TWO SECTOR WRITE DATA (PART 1) MUST BE EXECUTED BEFORE THIS TEST.

TEST 53 TWO SECTOR READ DATA (PART 2)

ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 23. VERIFY THAT CORRECT DATA IS READ AND A MID-TRANSFER SEEK DOES NOT OCCUR.

NOTE: TWO SECTOR WRITE DATA (PART 2) MUST BE EXECUTED BEFORE THIS TEST.

TEST 54 TWO SECTOR READ DATA (PART 3)

ISSUE A READ DATA OF 401 WORDS TO CYLINDER 312, TRACK 0, SECTOR 10. VERIFY THAT ALL 401 WORDS ARE PLACED IN MEMORY.

NOTE: TWO SECTOR WRITE DATA (PART 3) MUST BE EXECUTED

BEFORE THIS TEST.

TEST 55 MID-TRANSFER SEEK ON READ (PART 1)

ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 25. VERIFY THAT CORRECT DATA IS READ AND A MID-TRANSFER SEEK DOES OCCUR.

NOTE: MID-TRANSFER SEEK ON WRITE (PART 1) MUST BE EXECUTED BEFORE THIS TEST.

TEST 56 MID-TRANSFER SEEK ON READ (PART 2)

ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 2, SECTOR 25. VERIFY THAT CORRECT DATA IS READ AND A MID-TRANSFER SEEK DOES OCCUR.

NOTE: MID-TRANSFER SEEK ON WRITE (PART 2) MUST BE EXECUTED BEFORE THIS TEST.

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

TEST 57 CYLINDER ADDRESS OVERFLOW (PART 1)

ISSUE A READ DATA OF 400 WORDS TO CYLINDER 632, TRACK 2, SECTOR 25. MAKE SURE CYLINDER ADDRESS OVERFLOW ERROR DOES NOT OCCUR. CYLINDER 1456 IS USED FOR THE RK07.

TEST 60 CYLINDER ADDRESS OVERFLOW (PART 2)

ISSUE A READ DATA OF 401 WORDS TO CYLINDER 632, TRACK 2, SECTOR 25. MAKE SURE CYLINDER ADDRESS OVERFLOW ERROR DOES OCCUR. CYLINDER 1456 IS USED FOR THE RK07.

**18 BIT DATA TRANSFER TESTS

TEST 61 FORMAT IN 24 SECTOR FORMAT

FORMAT CYLINDER 312, TRACK 0, AND TRACK 1 FOR 24 SECTOR FORMAT. VERIFY FORMAT AND THAT DATA LATE DID NOT OCCUR WITH WRITE HEADER ON IN READING DATA BUFFER AFTER READ HEADER.

TEST 62 24 SECTOR FORMAT DATA TRANSFER (PART 1)

ISSUE A WRITE DATA OF 400 WORDS IN 24 SECTOR FORMAT TO CYLINDER 312, TRACK 0, SECTOR 0. READ SECTOR BACK AND MAKE SURE IT IS CORRECT.

TEST 63 24 SECTOR FORMAT DATA TRANSFER (PART 2)

ISSUE A WRITE DATA OF 1000 WORDS IN 24 SECTOR FORMAT TO CYLINDER 312, TRACK 0, SECTOR 23. READ SECTOR BACK AND MAKE SURE IT IS CORRECT. MAKE SURE THAT MID-TRANSFER SEEK HAS TAKEN PLACE.

**SPECIAL DATA TRANSFER TESTS

TEST 64 MULTI-SECTOR DATA TRANSFER AND BSE

FORMAT CYLINDER 312, TRACK 0 IN 26 SECTOR FORMAT WITH SECTOR 1 MARKED BAD. ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS AND RKDA IS CORRECT. READ SECTOR 0 AND MAKE SURE IT IS CORRECT.

ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS AND THE PREVIOUS SECTOR IS LOADED CORRECTLY INTO MEMORY.

TEST 65 FORMAT TEST

819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874

FORMAT CYLINDER 312, TRACKS 0 AND 1 IN 26 SECTOR
FORMAT. MAKE SURE NO ERRORS SET. READ SECTORS 0-25
AND MAKE SURE DATA CHECK DOES NOT OCCUR.

**WRITE CHECK TESTS

TEST 66 WRITE-CHECK WITH NO ERROR

WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH A KNOWN
PATTERN. DO A WRITE-CHECK OF 400 WORDS. MAKE SURE NO
ERROR OCCURS.

TEST 67 WRITE CHECK ERROR (PART 1)

WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH ALL ZEROES.
WRITE CHECK CYLINDER 312, TRACK 0, SECTOR 0 WITH SAME.
DATA EXCEPT WORD 110 HAS ONE OF THE FOLLOWING
CONFIGURATIONS:

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

MAKE SURE WRITE CHECK ERROR SET FOR EACH OF THE
CONFIGURATIONS AND THAT THE BUS ADDRESS AND WORD COUNT
IS CORRECT.

TEST 70 WRITE CHECK ERROR (PART 2)

WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH 17777 IN
ALL WORDS. WRITE CHECK CYLINDER 312, TRACK 0, SECTOR
0 WITH THE SAME DATA EXCEPT WORD 120 HAS ONE OF THE
FOLLOWING CONFIGURATIONS:

177776 177757 177377 167777
177775 177737 176777 157777
177773 177677 175777 137777
177767 177577 173777 077777

MAKE SURE WRITE CHECK ERROR SET FOR EACH OF THE
CONFIGURATIONS AND THAT THE BUS ADDRESS AND WORD COUNT
IS CORRECT.

TEST 71 WRITE CHECK OF PARTIAL SECTOR

WRITE CYLINDER 312, TRACK 0, SECTOR WITH A KNOWN
CONFIGURATIONS. ISSUE A WRITE CHECK COMMAND OF 110
WORDS MAKING SURE THE 111TH WORD IS DIFFERENT THAN
DATA ON DISK. MAKE SURE WRITE CHECK ERROR DOES NOT
SET.

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
926
927
928
929
930

**MAXIMUM DATA TRANSFER AND CONTROLLER TIME OUT

TEST 72 MAXIMUM DATA TRANSFER (PART 1)

IN THE FIRST PASS OF THE PROGRAM, THE HEADERS OF THE FIRST 4 CYLINDERS ARE WRITTEN. THIS IS DONE TO INSURE THE FORMAT IS CORRECT.

ZERO OUT THE FIRST 256 SECTORS OF THE DISK WITH ONE SECTOR WRITES. ISSUE A SEEK TO CYLINDER 0, TRACK 0. ISSUE A WRITE DATA OF MAXIMUM DATA TRANSFER 200000 WORDS TO CYLINDER 0, TRACK 0, SECTOR 0. MAKE SURE CONTROLLER TIME OUT IS NOT SET. CHECK CYLINDER ADDRESS, DISK ADDRESS, BUS ADDRESS AND WORD COUNT. READ EACH SECTOR TO MAKE SURE IT WAS WRITTEN CORRECTLY.

NOTE: THIS TEST IS EXECUTED ONLY IF NO BAD SECTORS ARE PRESENT IN THE FIRST 256 SECTORS ON THE PACK.

TEST 73 MAXIMUM DATA TRANSFER (PART 2)

ZERO OUT FIRST 256 SECTORS OF THE DISK WITH 200000 WORD WRITE. SEEK TO CYLINDER 632. ISSUE A WRITE OF MAXIMUM DATA TRANSFER 200000 WORD WRITE. MAKE SURE CONTROLLER TIME OUT IS NOT SET. CHECK CYLINDER ADDRESS DISK ADDRESS, BUS ADDRESS AND WORD COUNT. SEEK TO CYLINDER 632. ISSUE A WRITE CHECK OF 200000 WORDS. MAKE SURE NO ERROR SETS.

NOTE: THIS TEST IS EXECUTED ONLY IF NO BAD SECTORS ARE PRESENT IN THE FIRST 256 SECTORS ON THE PACK.
CYLINDER 1456 IS USED FOR THE RK07.

TEST 74 CONTROLLER TIME OUT

SEEK TO CYLINDER 632. ISSUE A RECALIBRATE AND DO NOT WAIT FOR SECOND INTERRUPT. NOW ISSUE A READ HEADER OF CYLINDER 0, TRACK 0. MAKE SURE CONTROLLER TIME OUT SETS.
CYLINDER 1456 IS USED FOR THE RK07.

**ERRORS DURING DATA TRANSFER

TEST 75 LIMIT DETECT ON DATA TRANSFER

ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. ISSUE

931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986

A SEEK TO CYLINDER 2 WITH BAD PARITY. ISSUE A DRIVE CLEAR. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 1, TRACK 0, HEAD 0. SEEK INCOMPLETE BECAUSE OF OUTER

TEST 76 PROGRAMMING ERROR

ISSUE A SUBSYSTEM CLEAR. ISSUE A READ DATA OF 400 WORDS ON CYLINDER 0, TRACK 0, SECTOR 0. DURING READ ISSUE A WRITE TO THE SPARE REGISTER. MAKE SURE PROGRAMMING ERROR SETS.

TEST 77 ECC HARD

ISSUE A SUBSYSTEM CLEAR. ISSUE A WRITE DATA WORDS CONSISTING OF 177777 TO CYLINDER 0, TRACK 0, SECTOR 0. NOW WRITE ALL ZEROS TO CYLINDER 0, TRACK 0, SECTOR 0. DURING WRITE ISSUE CONTROLLER CLEAR. MAKE SURE PROGRAMMING ERROR IS RESET. NOW ISSUE A READ DATA TO CYLINDER 0, TRACK 0, HEAD 0 AND AN ECC HARD ERROR SHOULD SET.

TEST 100 DRIVE TIMING ERROR

ISSUE A SUBSYSTEM CLEAR. SEEK TO CYLINDER 632. ISSUE A RECALIBRATE BUT DO NOT WAIT FOR SECOND INTERRUPT. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER OF CYLINDER 0, TRACK 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. TURN OFF DIAGNOSTIC MODE. DRIVE TIMING ERROR SHOULD SET BECAUSE OF NO DATA TRANSISTIONS ON DATA LINE.

**ERROR FORCING IN DRIVE

TEST 101 INITIALIZE CLEARING SACK

ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE DRIVE. ISSUE A SUBSYSTEM CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A SELECT COMMAND WITH MESSAGE ID = 3 AND DRIVE SELECTED = 0. CLOCK THROUGH PHASE ADDRESS 6. TURN OFF DIAGNOSTIC MODE. MAKE SURE UNIT FIELD ERROR DOES NOT SET.

TEST 102 DRIVE OFF TRACK

ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. ISSUE OFFSET OF +1200 MICRO-INCHES. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 1 WORD TO CYLINDER 0, TRACK 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. TURN OFF DIAGNOSTIC MODE. DRIVE OFF TRACK SHOULD SET IN DRIVE. REPEAT FOR ALL AVAILIABLE DRIVES.

987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042

TEST 103 FILE UNSAFE

ISSUE A SUBSYSTEM CLEAR. ISSUE A RECLAIBRATE. ISSUE A READ HEAD OF CYLINDER 0, TRACK 0 IN 24 SECTOR FORMAT. DO A SELECT COMMAND IN 26 SECTOR FORMAT. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO CYLINDER 0, TRACK 0, ONE WORD IN 26 SECTOR FORMAT. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE. TURN OFF DIAGNOSTIC MODE. FILE UNSAFE SHOULD SET BECAUSE OF ATTEMPTING TO WRITE THROUGH SECTOR PULSE. REPEAT FOR ALL AVAILIABLE DRIVES.

TEST 104 DUMMY TEST FOR PREVIOUS TEST EXIT

THIS TEST IS PRESENT TO MAKE \$SW08TB TABLE HAVE AN ENTRY WHICH RELATES TO 'NEWDRV'. THIS IS NECESSARY IF AN ERROR OCCURS IN THE PRECEEDING TEST AND THAT ERROR ABORTS THE TEST. IF THIS TEST WERE NOT PRESENT, THE PROGRAM WOULD SKIP THE 'NEWDRV' ROUTINE AND GO TO THE TEST FOLLOWING 'NEWDRV'.

IN ADDITION, THE DRIVE IS CLEARED AND THE HEADS ARE

ALLOWED TO RELOAD. THIS MUST BE DONE TO PREVENT UNEXPECTED INTERRUPTS FROM THE DRIVE BECOMING READY AT A LATER TIME.

**MULTI-DRIVE OPERATIONS

TEST 105 RESET ATTENTIONS WITH UNIBUS INIT

DO A RECALIBRATE ON ALL AVAILIABLE DRIVES. ISSUE A RESET. MAKE SURE ALL ATTENTION RESET.

TEST 106 RESET ATTENTIONS WITH SUBSYSTEM CLEAR

DO A RECALIBRATE ON ALL AVAILIABLE DRIVES. ISSUE A SUBSYSTEM CLEAR. MAKE SURE ALL ATTENTIONS RESET.

TEST 107 SVAL AND ATTENTION FROM OTHER DRIVE

DO A RECALIBRATE ON ONE AVAILIABLE DRIVE. DO A SELECT ON ANOTHER AVAILIABLE DRIVE. MAKE SURE STATUS VALID IS SET. WAIT FOR SECOND INTERRUPT FROM RECALIBRATE MAKE SURE STATUS VALID REMAINS SET AND DRIVE STATUS CHANGE REMAINS RESET.

REPEAT FOR ALL COMBINATIONS OF TWO AVAILIABLE DRIVES.

NOTE: THIS TEST WILL ONLY BE DONE IF AT LEAST TWO DRIVES ARE AVAILIABLE.

1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089

TEST 110 OVERLAPPED OPERATIONS

DO A RECALIBRATE ON BOTH DRIVE A AND DRIVE B. ISSUE A SEEK ON DRIVE A TO CYLINDER 1. IMMEDIATELY ISSUE A WRITE DATA TO CYLINDER 100, TRACK 0, HEAD 0 ON DRIVE B. AT THE END OF THE DATA TRANSFER NO ERRORS SHOULD BE SET AND DRIVE A HAS ATTENTION SET.

REPEAT FOR ALL COMBINATIONS OF TWO DRIVES.

NOTE: IF ONLY ONE DRIVE IS AVAILABLE THE TEST WILL NOT BE DONE.

5.0 ERROR REPORTING

A DETAILED DESCRIPTION OF THE ERROR FORMATS AND REPORTS CONTENTS IS GIVEN HERE. THIS IS ESSENTIALLY THE SAME AS CAN BE FOUND IN THE LISTING COMMENTS UNDER THE ERROR POINTER TABLE.

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

EM AND DH ARE ASCIZ DATA. EM IS ALWAYS A MESSAGE BUT DH CAN BE A MESSAGE OR A SET OF COLUMN LABELS SPACED ACCROSS THE PAGE, DT IS A STRING OF WORDS THAT POINT TO THE DATA TO BE TYPED, AND DF IS A STRING OF WORDS THAT TELL HOW THE DT WORDS ARE TO BE TYPED. IF ANY OF THE POINTERS ARE NOT NEEDED, FOR A PARTICULAR FORMAT, IT IS REPLACED WITH A ZERO. THE NORMAL USAGE OF THE ERROR TABLE IS TO HAVE A TABLE ENTRY FOR EACH ERROR MESSAGE THAT CAN OCCUR. IN THE INTEREST OF ECONOMICS OF CORE MEMORY, THIS PROGRAM USES THE ERROR TABLE IN A SLIGHTLY DIFFERENT MANNER AS DESCRIBED BELOW.

THE ERROR TABLE ENTRIES MAKE UP A SET OF REPORT FORMATS THAT

1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145

ARE USED THROUGHOUT THE PROGRAM. WHEN AN ERROR IS TO BE REPORTED, THE TABLE ENTRY THAT PROVIDES THE DESIRED FORMAT IS CHOSEN FROM THE DEFINED SET. THE TABLE ENTRY CHOSEN IS THEN ALTERED BY CHANGING THE FIRST (AND POSSIBLY THE SECOND) WORD TO CONTAIN THE ADDRESS OF THE ASCIZ STRING THAT MAKES UP THE MESSAGE PORTION OF THE REPORT. THE DATA FIELDS FOR THAT ENTRY ARE NEVER CHANGED, NOR ARE THE COLUMN LABELS OR POSITIONS.

THE FORMAT THAT EACH TABLE ENTRY PROVIDES IS SHOWN BELOW WITH THE DEFINITION OF THE ENTRY. ALL DATA FIELDS ARE TYPED IN OCTAL.

ERROR ITEM 1
(MESSAGE)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM

ERROR ITEM 2
(MESSAGE)
(MESSAGE)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA
T.CS1 T.CS2 T.DS T.ER T.ASOF T.DCYL T.DA
RKBA RKWC
T.BA T.WC

ERROR ITEM 3
(MESSAGE)
TST NUM ERR PC DRIVE

\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RKMRI
T.CS1 T.CS2 T.DS T.ER T.ASF T.MRI

ERROR ITEMS 4, 5, 6, AND 7 ARE USED TO REPORT ERRORS THAT ARE THE RESULT OF A HARDWARE ERROR INDICATOR BEING SET WHEN NOT EXPECTED, NOT SET WHEN IT IS EXPECTED, OR BOTH. THE ERROR REPORT WILL CONTAIN (1) ALL THE ERRORS THAT WERE DETECTED, (2) ALL THE EXPECTED ERRORS THAT DID NOT OCCUR, OR (3) ALL THE EXPEDTED BUT NOT SET ERRORS AND THE UNEXPECTED BUT SET ERRORS.

THE MESSAGE ITSELF EXPLAINS THE CIRCUMSTANCE FOR THE REPORT. INCLUDED IN THE REPORT WILL BE ONE OR MORE OF THE FOLLOWING STATEMENTS:

"THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:"
"THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:"
"THE ABOVE ARE ERRORS SET IN OPERATION:"

PRECEEDING ANY ONE OF THESE LINES WILL BE ONE OR MORE LINES THAT SPECIFY THE EXACT ERROR. FOLLOWING THE LAST LINE WILL BE A LINE THAT IDENTIFIES THE OPERATION BEING PERFORMED.

1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201

EXAMPLE:
NON-EXISTANT DRIVE
THE ABOVE ARE ERRORS SET IN OPERATION:
DRIVE SELECT

(ADDITIONAL LINES OF INFORMATION)

THIS IS THE RESULT OF AN ERROR SET IN A SELECT OPERATION.

EXAMPLE:
NON-EXISTANT DRIVE
THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:
DRIVE SELECT

(ADDITIONAL LINES OF INFORMATION)

THIS IS THE RESULT OF AN EXPECTED ERROR THAT DID NOT OCCUR,
I.E. A NON-EXISTANT DRIVE WAS ADDRESSED BUT NED WAS NOT SET.

EXAMPLE:
NON-EXISTANT MEMORY
THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:
UNIBUS PARITY ERROR
THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:
WRITE DATA

(ADDITIONAL LINES OF INFORMATION)

THIS IS AN EXAMPLE OF NON-EXISTANT MEMORY BEING SET WHEN
UNIBUS PARITY ERROR WAS EXPECTED.

ERROR ITEM 4
(DESCRIPTION OF ERROR)
ERROR IN OPERATION
(DESCRIPTION OF OPERATION)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA
T.CS1 T.CS2 T.DS T.ER T.ASOF T.DCYL T.DA
RKBA RKWC
T.BA T.WA
A00 B00 A01 B01 A02 B02 A03 B03
\$REG10 \$REG11 \$REG12 \$REG13 \$REG14 \$REG15 \$REG16 \$REG17

THE ERRORS REPORTED BY THIS FORMAT ARE:
CONTROLLER DETECTED DRIVE BUS ERROR
DRIVE DETECTED DRIVE BUS ERROR
SEEK INCOMPLETE
NON-EXECUTABLE DRIVE FUNCTION
DRIVE TIMING ERROR
DRIVE UNSAFE
AC LOW
SPINDLE SPEED LOSS
DRIVE OFF TRACK

1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257

ILLEGAL DRIVE ADDRESS ERROR
CYLINDER OVERFLOW
DRIVE TYPE ERROR
FORMAT ERROR
WRITE LOCK ERROR

ERROR ITEM 5
THIS ENTRY IS THE SAME AS ITEM 4 WITH THE ADDITION OF A
MESSAGE THAT FOLLOWS. THIS MESSAGE IS:

'ANY FIELD WITH ALL ONES MUST BE CONSIDERED INVALID'

THIS REPORT WILL BE PRINTED WHEN THE GATHERING OF DATA FOR A00
THRU B03 IS NOT ACCOMPLISHED WITHOUT ERROR.

ERROR ITEM 6

(DESCRIPTION OF ERROR)
ERROR IN OPERATION
(DESCRIPTION OF OPERATION)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA
T.CS1 T.CS2 T.DS T.ER T.ASOF T.DCYL T.DA
RKBA RKWC
T.BA T.WC

THE ERRORS REPORTED BY THIS FORMAT ARE:
DATA CHECK
WRITE CHECK

ECC HARD
DATA LATE
OPERATION INCOMPLETE
HEADER VRC ERROR
BAD SECTOR ERROR

ERROR ITEM 7

(DESCRIPTION OF ERROR)
ERROR IN OPERATION
(DESCRIPTION OF OPERATION)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF
T.CS1 T.CS2 T.DS T.ER T.ASOF

THE ERRORS REPORTED BY THIS FORMAT ARE:
NON-EXISTANT DRIVE
NON-EXISTANT MEMORY
CONTROLLER TIME OUT
UNIT FIELD ERROR
MULTIPLE DRIVE SELECT
PROGRAMMING ERROR
UNIBUS PARITY ERROR

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
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313

ILLEGAL FUNCTION CODE

DESCRIPTION OF OPERATION CAN BE ANY COMMAND, EITHER LEGAL OR ILLEGAL.

ERROR ITEM 10

(DESCRIPTION OF ERROR)
ERROR AT COMPLETION OF OPERATION
(DESCRIPTION OF OPERATION)
TST NUM ERR PC DRIVE
\$TESTN ERRPC DRVNUM
EXPT IS
\$REG10 \$REG11

THE ERRORS REPORTED BY THIS FORMAT ARE SOFTWARE DETECTED BY COMPARING EXPECTED RESULTS WITH ACTUAL RESULTS. THE SPECIFIC ERRORS ARE:

WORD COUNT INCORRECT
BUS ADDRESS INCORRECT
CYLINDER ADDRESS INCORRECT
TRACK ADDRESS INCORRECT
SECTOR ADDRESS INCORRECT

ERROR ITEM 11

(ERROR INDICATOR OR STATUS BIT)
NOT SET AS A RESULT OF
(ANOTHER ERROR INDICATOR, STATUS BIT, OR OPERATION)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RKMRI

T.CS1 T.CS2 T.DS T.ER T.ASOF T.MR1

ERROR ITEM 12

THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:

'NOT RESET AS A RESULT OF'

ERROR ITEM 13

THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:

'SET AS A RESULT OF'

ERROR ITEM 14

THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:

'RESET AS A RESULT OF'

ERROR ITEM 15

(HEADER WORD MISCOMPARE) OR (DATA MISCOMPARE)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
GOOD BAD WORD NUM
\$REG10 \$REG11 \$REG12

1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369

ERROR ITEM 16
ADDITIONAL LINES OF GOOD, BAD, WORD NUM FOR ERROR 15.

6.0 SUBROUTINES

IN THE INTEREST OF CONSERVING MEMORY, IT IS NECESSARY TO MAKE EXTENSIVE USE OF SUBROUTINES. HOWEVER, IN THE INTEREST OF PRESERVING CODE READABILITY, SUBROUTINE NAMING IS DESCRIPTIVE OF THE FUNCTION PERFORMED. THE SUBROUTINE FUNCTION IS KEPT SMALL AND IN GENERAL A SUBROUTINE ONLY PERFORMS ONE FUNCTION, I.E., LOAD THE RK611 REGISTER AND START AN OPERATION (TLOADRK) OR WAIT A SPECIFIED NUMBER OF MILLISECONDS FOR AN INTERRUPT (TWTNN WHERE NN VARIES FROM CALL TO CALL AND IS THE TIME TO WAIT). THE FOLLOWING IS A DESCRIPTION OF THE SUBROUTINES NOT PROVIDED BY SYSMAC:

LINE CLOCK CALIBRATE

WAITS FOR A LINE CLOCK INTERRUPT TO CALIBRATE THE INTERRUPTS TO A MEANINGFUL TIME VALUE. IN ADDITION IT PRESETS THE TIMCNT IF THERE IS NO LINE CLOCK. TIMCNT IS USED IN THE LINE CLOCK SIMULATOR.

CALL: JSR PC,CLKCAL

OPTION PRESENT TEST AND SETUP

THIS ROUTINE CHECKS IF THE MEMORU PARITY OPTION AND THE LINE CLOCK ARE ON THE SYSTEM. FLAGS ARE SET IF PRESENT; CLEARED OTHERWISE, AND THE APPROPRIATE INTERRUPT VECTORS ARE SET UP.

CALL: JSR PC,OPTTST

LINE CLOCK SIMULATION ROUTINE

THIS ROUTINE IS USED TO SIMULATE THE LINE CLOCK. TO DO THIS THE VALUE STORED IN MILCNT IS USED AS THE BASE AND REPRESENTS THE NUMBER OF TIMES A DECREMENT AND BRANCH LOOP CAN BE DONE IN 1 MILLISECOND. THE TIMCNT VALUE IS DECREMENTED AND IF IT REACHED 0 THE LINE CLOCK TICK COUNTER IS BUMPED. THEN THE TIMCNT IS RESET TO 16 (REPRESENTS 16 MILLISECONDS PER LINE CLOCK TICK).

THUS THE ROUTINE RETURNS TO THE CALLER AFTER 1

MILLISECOND AND BUMPS THE LINE CLOCK TICK COUNTER FOR EACH 16 CALLS.

CALL: JSR PC,MYTIME

WAIT FOR INTERRUPT ROUTINE

THE ROUTINE IS ENTERED BY ONE OF FOURTEEN TRAP CALLS. THE CALL SPECIFIES HOW MANY TICKS OF THE LINE CLOCK ARE TO ELAPSE WHILE WAITING FOR INTERRUPT. IF INTERRUPT DOES NOT OCCUR IN THAT PERIOD OF TIME, AN ERROR MESSAGE IS PREPARED (BUT NOT PRINTED IN THE ROUTINE) AND THEN RETURNS TO THE LOCATION FOLLOWING THE CALL. IF INTERRUPT OCCURS THE RETURN IS BUMPED BY 2. NORMALLY AN ERROR CALL WILL BE IN THE LOCATION AFTER THE CALL TO INTERRUPT WAIT.

CALL: TWAT16 THROUGH TWAT159, TWAT1S, TWAT2S, TWAT8S, AND TWAT1M

'L' REGISTER LOADING ROUTINE

THE PARAMETERS FOLLOWING THE CALL ARE TRANSFERRED INTO THE 'L' REGISTERS L.CS1-L.DCYL. L.MR1 IS NOT LOADED IN THIS MANNER SINCE IT IS NOT COMMONLY LOADED FOR AN OPERATION. L.CS2 IS LOADED FROM DRVNUM.

CALL: JSR R4,LRLDAD
COMMAND
WORD COUNT
BUS ADDRESS
.BYTE SECTOR ADDRESS
.BYTE TRACK ADDRESS
CYLINDER ADDRESS

LOAD RK611 FOR OPERATION

THE REGISTER SETUP STORAGE IS TRANSFERRED TO THE RK611 REGISTER. THIS IS A STRAIGHT TRANSFER WITH NO CHECKING OR MANIPULATION OF THE REGISTER CONTENTS. L.CS1 IS TRANSFERRED LAST AS IT SHOULD BE IF THE GO BIT IS SET.

CALL: TLOADRK

STORE RK611 REGISTERS

1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425

1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481

ALL THE RK611 REGISTERS ARE STORED IN THE TEST TABLE T WITH THE EXCEPTION OF THE DATA BUFFER WHICH IS NOT STORED IN THIS ROUTINE.

CALL: TGETRK

BIT COUNTER IN A WORD

THE WORD WHOSE BITS MUST BE COUNTED IS PLACED ON THE STACK BY THE CALLING ROUTINE. THE NUMBER OF BITS FOUND IN THE WORD ARE PASSED BACK ON THE STACK.

CALL: JSR R4,BITCNT

MAINTENANCE CLOCK ROUTINE

THE PARAMETERS PASSED TO THIS ROUTINE ARE LOCATED IN THE ADDRESS AFTER THE CALL. THE FIRST BYTE CONTAINS THE NUMBER OF PHASE ADDRESSES THE CALLING ROUTINE WANTS THE CONTROLLER CLOCKED THROUGH AND THE SECOND BYTE CONTAINS THE NUMBER OF CLOCK TRANSITIONS (PARTIAL PHASES) TO BE DONE.

CALL: JSR R4,MCLOCK
.BYTE ;NUMBER OF CLOCK TRANSITIONS

.BYTE ;NUMBER OF PHASE ADDRESSES

READ AND SORT HEADERS

THE HEADERS IN THE CYLINDER AND TRACK SPECIFIED BY THE FIELDS IN THE "L" REGISTERS ARE READ AND STORED IN ASCENDING ORDER. CONTROLLER ERRORS ARE CHECKED IN THE READ HEADER OPERATION AND DATA LATE IS CHECKED AFTER EACH READ OF THE DATA BUFFER.

CALL: JSR R4,RDSTHD
TCHKOP ;RETURN POINT IF CERR IN READ
;HDR
ERROR 4 ;OR 5, 6, 7
ERROR 13 ;RETURN IF DATA LATE IN DB
;UNLOAD
ERROR 2 ;RETURN IF TOO SLOW OR
;IF HDR 0 NOT FOUND

GET DRIVE STATUS

1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537

THIS ROUTINE GETS ALL THE DRIVE STATUS AND PLACES IT IN \$REG10 THROUGH \$REG17. THESE REGISTORS ARE FIRST CLEARED TO ALL ONES AND THEN IF ERROR OCCURS WHILE GETTING STATUS, THE 1'S ARE LEFT IN THE REGISTERS.

CALL: JSR R4,GETDRS
BR ERROR PROCESSING ERROR RETURN
BR NO ERROR PROCESSING GOOD RETURN

SUBSYSTEM INITIALIZE AND INITIALIZE STATE TEST

THE SUBSYSTEM IS INITIALIZED WITH A SUBSYSTEM CLEAR COMMAND. CERR AND DI ARE MONITORED FOR A SHORT PEIROD OF TIME DURING WHICH THEY SHOULD BOTH RESET.

IF THEY DO RESET, READY IS TESTED TO INSURE IF SETS.

IF ANY OF THESE THREE CONDITIONS ARE NOT MET AN APPROPRIATE ERROR MESSAGE IS PREPARED AND REPORTED WHEN THE ROUTINE RETURN TO THE CALL. IF EVERY THING IS GOOD, THE RETURN SKIPS OVER THE ERROR CALL AND TEST ABORT.

THE USUAL CALL TO THIS ROUTINE WILL BE FOLLOWED BY AN ERROR MESSAGE AND BRANCH TO END OF TEST. THIS IS DONE BECAUSE FAILURE TO INITIALIZE CORRECTLY IS FATAL TO

THE TEST.

CALL: TSSINIT

WORD COUNT AT END OF OPERATION CHECK

THIS ROUTINE COMPARES THE CONTENTS OF THE TEST STORAGE FOR THE WORD COUNT AGAINST THE SUPPLIED VALUE. IF UNEQUAL, THE ERROR FLAG (WCERR) IS SET IN GROUP 4 ERROR FLAGS (GRP4ER)

CALL: JSR R4,CHKWC
.WORD ;EXPECTED WC VALUE

BUS ADDRESS AT END OF OPERATION CHECK

THIS ROUTINE COMPUTES THE EXPECTED BUS ADDRESS AT THE END OF A TRANSFER BY USING THE INITIAL BUS ADDRESS, ADDING IN THE INITIAL WORD COUNT, AND SUBTRACTING ANY

1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593

RESIDUAL WORD COUNT. IF THIS COMPUTED BA DOES NOT EQUAL THE CONTENTS OF RKBA AN ERROR FLAG (BAERR) IS SET IN GROUP 4 ERROR FIELD (GRP4ER)

IF BUS ADDRESS INCREMENT INHIBIT WAS SET, THE EXPECTED BUS ADDRESS IS THE STARTING BUS ADDRESS.

CALL: JSR R4,CHKBA

CYLINDER,TRACK,SECTOR TEST AT END OF OPERATION

THIS ROUTINE CHECKS THAT THE CONTENTS OF THE RKDCYL AND RKDA ARE CORRECT FOR ANY SIZE DATA TRANSFER AT THE END OF THE OPERATION. THE CONTENTS OF THE LOAD REGISTER STORAGE ARE COUNTED ON TO HAVE THE INITIAL VALUES TO MAKE THE NECESSARY CALCULATION.

ALL THREE VALUES ARE GENERATED AND STORED IN EXPECTED VALUES STORAGE EXPCYL, EXPTRK, EXPSEC. ALL 3 ARE CHECKED AND IF ONE OR MORE ARE WRONG, THE CORRESPONDING BIT IN THE ERROR FLAGS FIELD (GRP4ER) IS SET.

CALL: JSR R4,CHKCTS

OPERATION CHECK ROUTINE

THIS IS WHERE ALL HARDWARE ERROR INDICATORS AND SOME SOFTWARE ERRORS ARE CHECKED. THE GENERAL PROCEDURE FLOW IS AS FOLLOWS: THE ROUTINE IS CALLED WITH A TRAP (TCHKOP). THE LOCATION FOLLOWING THE TRAP CALL WILL HAVE AN ERROR TRAP WHICH THE ROUTINE WILL BYPASS IF NO ERROR IS FOUND. IF AN ERROR IS DETECTED, THE ERROR TRAP CALL IS MODIFIED BY THIS ROUTINE SUCH THAT THE ERROR TABLE ITEM WILL BE THE PROPER ITEM FOR THE FORMAT REQUIRED BY THIS ERROR. THE ERROR TRAP WILL BE MADE EITHER ERROR 4,5,6,7, OR 10. REFER TO THE ERROR ITEM TABLE FOR A DESCRIPTION OF THE FORMAT AND WHICH ERRORS ARE DISPLAYED IN WHAT FORMAT.

FOR NO EXPECTED ERRORS:
CALL: TCHKOP

FOR EXPECTED ERRORS:
CALL: TCHKWE
 .WORD :GROUP 1 EXPECTED ERRORS
 .WORD :GROUP 2 EXPECTED ERRORS
 .WORD :GROUP 3 EXPECTED ERRORS

WHERE EACH BIT IN THE THREE WORDS FOLLOWING THE CALL

REPRESENT A SPECIFIC ERROR. THE BIT ASSIGNMENTS ARE GIVEN BELOW:

- GROUP 1 ERRORS:
BIT 0 - CONTROLLER DETECTED DRIVE BUS PARITY ERROR
BIT 1 - SEEK INCOMPLETE
BIT 2 - NON-EXECUTABLE DRIVE FUNCTION
BIT 3 - DRIVE DETECTED DRIVE BUS PARITY ERROR
BIT 4 - FORMAT ERROR
BIT 5 - DRIVE TYPE ERROR
BIT 6 - AC LOW ERROR
BIT 7 - SPEED LOSS ERROR
BIT 8 - DRIVE OFF TRACK ERROR
BIT 9 - CYLINDER OVERFLOW ERROR
BIT 10 - ILLEGAL DISK ADDRESS ERROR
BIT 11 - WRITE LOCK ERROR
BIT 12 - DRIVE TIMING ERROR
BIT 13 - NO CERR WITH OTHER ERROR SET ERROR
BIT 14 - DRIVE UNSAFE ERROR
BIT 15 - CERR BUT NO OTHER ERROR SET ERROR

- GROUP 2 ERRORS:
BIT 0 - ECC HARD ERROR
BIT 1 - DATA CHECK ERROR
BIT 2 - WRITE CHECK ERROR
BIT 3 - DATA LATE ERROR
BIT 4 - OPERATION INCOMPLETE ERROR
BIT 5 - HEADER VRC ERROR
BIT 6 - BAD SECTOR ERROR

- GROUP 2 ERRORS:
BIT 0 - NON-EXISTANT DRIVE ERROR
BIT 1 - CONTROLLER TIMEOUT ERROR
BIT 2 - UNIT FIELD ERROR
BIT 3 - MULTIPLE DRIVE SELECT ERROR
BIT 4 - PROGRAMMING ERROR
BIT 5 - NON-EXISTANT MEMORY ERROR
BIT 6 - UNIBUS PARITY ERROR
BIT 7 - ILLEGAL FUNTION ERROR

BAD SECTOR CHECK

THE FIELD WHOSE ADDRESS IS IN THE LOCATION AFTER THE CALL IS CHECKED TO SEE IF ANY SECTORS ARE LISTED THEREIN THAT HAVE THE CYLINDER AND TRACK ADDRESS SPECIFIED IN L.DCYL AND L.DT. IF A SECTOR IS FOUND IN THIS FIELD THAT IS BAD FOR THAT CYLINDER AND TRACK, THE SECTOR NUMBER IS PLACED ON THE STACK. THE TOTAL NUMBER OF BAD SECTORS IS PLACED ON THE STACK AFTER THE ENTIRE FIELD IS SEARCHED.

1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649

CALL: JSR R4,BDSRCK
<ADDRESS OF FIELD TO BE SEARCHED>

DATA GENERATION AND COMPARE ROUTINE

CALLS: JSR R4,GENCOM
CONTROL WORD

JSR R4,GENCOM
CONTROL WORD
LENGTH

JSR R4,GENCOM
CONTROL WORD
RELOCATION CONSTANT
LENGTH

RETURN: RTS R4

R4 IS ADJUSTED IN THE CODE FOR THE FOLLOWING RETURNS:
THE FIRST CALL RETURNS TO THE LOCATION FOLLOWING THE
CONTROL WORD. THIS IS UNCONDITIONAL.

THE SECOND CALL RETURNS TO THE LOCATION FOLLOWING THE
LENGTH IF THE OPERATION REQUIRES DATA COMPARE AND DATA
MISCOMPARED. IF DATA IS TO BE GENERATED ONLY OR NO
DATA COMPARE ERRORS OCCURRED, THE RETURN IS TO LENGTH
+4.

THE THIRD CALL IS IDENTICAL TO THE SECOND.

DEFINITION OF CONTROL WORD:

- BIT 15 - DO COMPARE OPERATION OF Ibuff (SOURCE) TO
Obuff (DESTINATION). EXPECTED VALUES ARE
IN Obuff (DESTINATION).
- BIT 14 - RESUME COMPARE OPERATION FROM POINT LEFT
BY LAST COMPARE.
- BIT 13 - INVOKE MEMORY MANAGEMENT FOR SOURCE
(IBUFF).
- BIT 12 - INVOKE MEMORY MANAGEMENT FOR DESTINATION
(OBUFF).
- BIT 11 - REPEAT FIRST WORD OF SELECTED PATTERN
THROUGHOUT OBUFF.
- BIT 10 - CLEAR Ibuff TO PATTERN SELECTED.
- BIT 9 - BUILD HEADERS, CONSIDERING BS FILES
- BIT 8 - BUILD HEADERS, ALL SECTORS INDICATE GOOD
SECTORS.
- BIT 7 - HEADER OPERATION SPECIFIED (EITHER
COMPARE OR BUILD).
- BIT 6 TO 0 - PATTERN SELECT FIELD, OCTAL ENCODED. 0
INDICATES NO DATA GENERATION, 1 IS ALL

1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705

1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761

ZEROS, AND 7 IS ALL ONES. OTHER PATTERNS PROVIDED ARE PATTERNS 2-6, 8-16. REFER TO THE PROGRAM LISTING FOR PAT02 THROUGH PAT16.

EXPLANATION OF CALLS:

THE CALL WITH CONTROL WORD THE ONLY PARAMETER IS USED FOR BUILDING OR COMPARING HEADERS OR RESUMING A COMPARE OPERATION.

THE CALL WITH CONTROL WORD AND LENGTH AS PARAMETERS IS USED FOR DATA GENERATION OR COMPARE AND FOR Ibuff INITIALIZATION.

THE CALL WITH CONTROL WORD, RELOCATION CONSTANT, AND LENGTH IS USED FOR DATA GENERATION OR COMPARE WITH MEMORY MANAGEMENT.

DESCRIPTION:

THIS ROUTINE IS MULTI-PURPOSE AND WILL PERFORM THE FOLLOWING:

- A. BUILD HEADERS, EITHER 20 OR 22 SECTORS/TRACK MODE. THE ROUTINE WILL BUILD THE HEADERS AS ALL GOOD SECTORS (BIT 8) OR TAKE THE BAD SECTOR FILES (HARDWARE OR SOFTWARE) FOR EITHER FORMAT) INTO ACCOUNT AND BUILD THE HEADERS WITH THE SECTORS MARKED BAD IF ANY SECTORS FOR THE CYLINDER - TRACK ARE LISTED THEREIN (BIT 9).
- B. COMPARE THE CONTENTS OF Ibuff AND Obuff (BIT 15). THE CONTENTS OF THE BUFFER MAY BE HEADERS OR DATA. A HEADER COMPARE OPERATION MAY BE SPECIFIED (BIT 7) WHICH WILL CAUSE THE COMPARE TO BE LIMITED TO 74(8) OR 102(8) WORDS OF HEADERS. THE LENGTH DEPENDS ON THE FORMAT BIT THAT WAS LAST SPECIFIED IN L.CS1. THE HEADERS MAY BE BUILT BEFORE THE COMPARE AS PART OF THE OPERATION (BIT 15 AND BIT 8 OR 9). DATA CAN ALSO BE GENERATED BEFORE THE COMPARE (NON-ZERO BITS 6-0).
- C. RESUME COMPARE OPERATION. IF A COMPARE OPERATION DETECTS A MISCOMPARE, THE ROUTINE RETURNS TO CALLER BUT STORES PARAMETERS SUCH THAT THE COMPARE CAN BE RESUMED. THIS IS DONE BY CALLING GENCOM WITH BIT 14 SET IN THE CONTROL WORD.
- D. DATA GENERATION OR COMPARE USING MEMORY MANAGEMENT. MEMORY MANAGEMENT CAN BE INVOKED FOR EITHER SOURCE OR DESTINATION BUT NOT FOR BOTH. IN THIS MANNER, DATA GENERATION CAN BE MADE TO PLACE DATA ANYWHERE IN AVAILABLE MEMORY. LIKEWISE DATA COMPARE WILL COMPARE THE CONTENTS OF Ibuff TO ANY

AREA OF AVAILABLE MEMORY.

PHASE LOCKED LOOP CLOCK ADJUSTMENT ROUTINE

THIS ROUTINE IS ENTERED VIA START LOCATION 220(8).
THE PROGRAM FIRST RUNS TEST 1, 2, AND 3 TO SET UP THE
INTERNAL PROGRAM VARIABLES AND THEN JUMPS TO THE CLOCK
ADJUST ROUTINE. THE ROUTINE SELECTS THE FIRST
AVAILABLE DRIVE AND SETS AND RESETS THE DIAGNOSTIC
MODE BIT IN RKMRI. INSTRUCTIONS ON WHERE TO SCOPE AND
WHAT TO ADJUST ARE TYPED ON THE CONSOLE.

THIS ROUTINE WILL LOOP UNTIL THE PROCESSOR IS HALTED.

7.0 REVISION HISTORY

- REVFO 1) ADD CODE IN PROGRAM INITIALIZATION TO
CLEAR DATA BUFFER MEMORY LOCATIONS
2) MODIFY CHAIN MODE LOOP ADDRESS IN \$EOP
FOR XXDP+ COMPATIBILITY
3) MODIFY 'OPTTS;' ROUTINE TO SUPPORT NEW
PROCESSOR AND MEMORY OPTIONS

%

1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787

```
1788 ; *** REV 007 ***
1789
1790 .NLIST CND,MD,MC,TOC
1791 .LIST ME
1792 .ENABL ABS,AMA
1793 $SWR= 167400
1794 $TN= 1
1795 .TITLE CZR6KGO RK6 FCTNL CTRL DIAG
1796 ;*COPYRIGHT (C) 1976,1981
1797 ;*DIGITAL EQUIPMENT CORP.
1798 ;*MAYNARD, MASS. 01754
1799 ;*
1800 ;*
1801 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1802 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
1803 ;*
1804 .SBTTL OPERATIONAL SWITCH SETTINGS
1805 ;*
1806 ;* SWITCH USE
1807 ;* -----
1808 ;* 15 HALT ON ERROR
1809 ;* 14 LOOP ON TEST
1810 ;* 13 INHIBIT ERROR TYPEOUTS
1811 ;* 12 ABORT PROGRAM AFTER 20 ERRORS
1812 ;* 11 INHIBIT ITERATIONS
1813 ;* 10 BELL ON ERROR
1814 ;* 9 LOOP ON ERROR
1815 ;* 8 LOOP ON TEST IN SWR<7:0>
1816 .SBTTL BASIC DEFINITIONS
1817
1818 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1819 STACK= 1100
1820 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
1821 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
1822
1823 ;*MISCELLANEOUS DEFINITIONS
1824 HT= 11 ;;CODE FOR HORIZONTAL TAB
1825 LF= 12 ;;CODE FOR LINE FEED
1826 CR= 15 ;;CODE FOR CARRIAGE RETURN
1827 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
1828 PS= 177776 ;;PROCESSOR STATUS WORD
1829 .EQUIV PS,PSW
1830 STKLMT= 177774 ;;STACK LIMIT REGISTER
1831 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
1832 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
1833 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
1834
1835 ;*GENERAL PURPOSE REGISTER DEFINITIONS
1836 R0= %0 ;;GENERAL REGISTER
1837 R1= %1 ;;GENERAL REGISTER
1838 R2= %2 ;;GENERAL REGISTER
1839 R3= %3 ;;GENERAL REGISTER
1840 R4= %4 ;;GENERAL REGISTER
1841 R5= %5 ;;GENERAL REGISTER
1842 R6= %6 ;;GENERAL REGISTER
1843 R7= %7 ;;GENERAL REGISTER
```

```
1844      000006      SP=      %6          ::STACK POINTER
1845      000007      PC=      %7          ::PROGRAM COUNTER
1846
1847      ;*PRIORITY LEVEL DEFINITIONS
1848      000000      PR0=      0          ::PRIORITY LEVEL 0
1849      000040      PR1=      40         ::PRIORITY LEVEL 1
1850      000100      PR2=      100        ::PRIORITY LEVEL 2
1851      000140      PR3=      140        ::PRIORITY LEVEL 3
1852      000200      PR4=      200        ::PRIORITY LEVEL 4
1853      000240      PR5=      240        ::PRIORITY LEVEL 5
1854      000300      PR6=      300        ::PRIORITY LEVEL 6
1855      000340      PR7=      340        ::PRIORITY LEVEL 7
1856
1857      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1858      100000      SW15=     100000
1859      040000      SW14=     40000
1860      020000      SW13=     20000
1861      010000      SW12=     10000
1862      004000      SW11=     4000
1863      002000      SW10=     2000
1864      001000      SW09=     1000
1865      000400      SW08=     400
1866      000200      SW07=     200
1867      000100      SW06=     100
1868      000040      SW05=     40
1869      000020      SW04=     20
1870      000010      SW03=     10
1871      000004      SW02=     4
1872      000002      SW01=     2
1873      000001      SW00=     1
1874      .EQUIV     SW09,SW9
1875      .EQUIV     SW08,SW8
1876      .EQUIV     SW07,SW7
1877      .EQUIV     SW06,SW6
1878      .EQUIV     SW05,SW5
1879      .EQUIV     SW04,SW4
1880      .EQUIV     SW03,SW3
1881      .EQUIV     SW02,SW2
1882      .EQUIV     SW01,SW1
1883      .EQUIV     SW00,SW0
1884
1885      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1886      100000      BIT15=    100000
1887      040000      BIT14=    40000
1888      020000      BIT13=    20000
1889      010000      BIT12=    10000
1890      004000      BIT11=    4000
1891      002000      BIT10=    2000
1892      001000      BIT09=    1000
1893      000400      BIT08=    400
1894      000200      BIT07=    200
1895      000100      BIT06=    100
1896      000040      BIT05=    40
1897      000020      BIT04=    20
1898      000010      BIT03=    10
1899      000004      BIT02=    4
```

```
1900      000002      BIT01= 2
1901      000001      BIT00= 1
1902      .EQUIV BIT09,BIT9
1903      .EQUIV BIT08,BIT8
1904      .EQUIV BIT07,BIT7
1905      .EQUIV BIT06,BIT6
1906      .EQUIV BIT05,BIT5
1907      .EQUIV BIT04,BIT4
1908      .EQUIV BIT03,BIT3
1909      .EQUIV BIT02,BIT2
1910      .EQUIV BIT01,BIT1
1911      .EQUIV BIT00,BIT0
1912
1913      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1914      000004      ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
1915      000010      RESVEC= 10      ;; RESERVED AND ILLEGAL INSTRUCTIONS
1916      000014      TBITVEC=14      ;; "T" BIT
1917      000014      TRTVEC= 14      ;; TRACE TRAP
1918      000014      BPTVEC= 14      ;; BREAKPOINT TRAP (BPT)
1919      000020      IOTVEC= 20      ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
1920      000024      PWRVEC= 24      ;; POWER FAIL
1921      000030      EMTVEC= 30      ;; EMULATOR TRAP (EMT) **ERROR**
1922      000034      TRAPVEC=34      ;; "TRAP" TRAP
1923      000060      TKVEC= 60      ;; TTY KEYBOARD VECTOR
1924      000064      TPVEC= 64      ;; TTY PRINTER VECTOR
1925      000240      PIRQVEC=240      ;; PROGRAM INTERRUPT REQUEST VECTOR
1926      .SBTTL MEMORY MANAGEMENT DEFINITIONS
1927
1928      ;*KT11 VECTOR ADDRESS
1929
1930      000250      MMVEC= 250
1931
1932      ;*KT11 STATUS REGISTER ADDRESSES
1933
1934      177572      SR0= 177572
1935      177574      SR1= 177574
1936      177576      SR2= 177576
1937      172516      SR3= 172516
1938
1939      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1940
1941      172300      KIPDR0= 172300
1942      172302      KIPDR1= 172302
1943      172304      KIPDR2= 172304
1944      172306      KIPDR3= 172306
1945      172310      KIPDR4= 172310
1946      172312      KIPDR5= 172312
1947      172314      KIPDR6= 172314
1948      172316      KIPDR7= 172316
1949
1950      ;*KERNEL "I" PAGE ADDRESS REGISTERS
1951
1952      172340      KIPAR0= 172340
1953      172342      KIPAR1= 172342
1954      172344      KIPAR2= 172344
1955      172346      KIPAR3= 172346
```

```
1956      172350      KIPAR4= 172350
1957      172352      KIPAR5= 172352
1958      172354      KIPAR6= 172354
1959      172356      KIPAR7= 172356
1960
1961      000210      AVECT1= 210      ;DEFINE RK611 VECTOR INTERRUPT
1962      000240      APRIOR= PR5      ;DEFINE RK611 PRIORITY
1963      177440      ABASE= 177440   ;DEFINE RK611 BASE BUS ADDRESS
1964
1965      .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1966
1967      000000      RKCS1= 0      ;CONTROL AND STATUS REGISTER 1
1968      000002      RKWC= 2      ;WORD COUNT REGISTER
1969      000004      RKBA= 4      ;BUS ADDRESS REGISTER
1970      000006      RKDA= 6      ;DESIRED TRACK SECTOR REGISTER
1971      000010      RKCS2= 10     ;CONTROL AND STATUS REGISTER 2
1972      000012      RKDS= 12     ;DRIVE STATUS REGISTER
1973      000014      RKER= 14     ;ERROR REGISTER
1974      000016      RKASOF= 16   ;ATTENTION SUMMARY AND OFFSET REGISTER
1975      000020      RKDCYL= 20   ;DESIRED CYLINDER REGISTER
1976      000024      RKDB= 24     ;DATA BUFFER
1977      000026      RKMR1= 26   ;MAINTENANCE REGISTER 1
1978      000034      RKMR2= 34   ;MAINTENANCE REGISTER 2
1979      000036      RKMR3= 36   ;MAINTENANCE REGISTER 3
1980      000030      RKECPS= 30  ;ECC POSITION INFORMATION
1981      000032      RKECPT= 32  ;ECC PATTERN INFORMATION
1982      000022      RKSPAR= 22  ;SPARE REGISTER
1983
1984      .SBTTL  DRIVE COMMANDS
1985
1986      000101      SELDRV= 101  ;SELECT DRIVE
1987      000103      PACK= 103   ;PACK ACKNOWLEDGE
1988      000105      CLEAR= 105  ;DRIVE CLEAR
1989      000107      UNLOAD= 107 ;UNLOAD
1990      000111      SRTSPL= 111 ;START SPINDLE
1991      000113      RECAL= 113  ;RECALIBRATE
1992      000115      OFFSET= 115 ;OFFSET
1993      000117      SEEK= 117   ;SEEK
1994      000121      RDDATA= 121 ;READ DATA
1995      000123      WRDATA= 123 ;WRITE DATA
1996      000125      RDHEAD= 125 ;READ HEADER
1997      000127      WRHEAD= 127 ;WRITE HEADER AND DATA
1998      000131      WRTCHK= 131 ;WRITE CHECK
1999      000300      INTR= 300  ;GENERATE INTERRUPT TO CPU
2000
2001      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
2002
2003      000001      GO= BIT0    ;GO BIT
2004      000100      IE= BIT6    ;INTERRUPT ENABLE
2005      000200      RDY= BIT7   ;CONTROLLER READY
2006      000400      BA16= BIT8  ;BUS ADDRESS BIT 16
2007      001000      BA17= BIT9  ;BUS ADDRESS BIT 17
2008      002000      CDT= BIT10  ;CONTROLLER DRIVE TYPE (0=RK06)
2009      004000      CTO= BIT11  ;CONTROLLER TIMED OUT WAITING FOR
2010      ; DRIVE RESPONSE
2011      010000      CFMT= BIT12 ;CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
```

2012	020000	SPAR= BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
2013	040000	DI= BIT14	:DRIVE INTERRUPT
2014	100000	CERR= BIT15	:CONTROLLER ERROR
2015	100000	CCLR= BIT15	:CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

2019	000007	DRVMSK= 7	:MASK FOR DRIVE SELECTION CODE
2020	000010	RLS= BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
2021	000020	BAI= BIT4	:BUS ADDRESS INCREMENT INHIBIT
2022	000040	SCLR= BIT5	:CLEAR CONTROLLER AND ALL DRIVES
2023	000100	IR= BIT6	:INPUT READY
2024	000200	OR= BIT7	:OUTPUT READY
2025	000400	UFE= BIT8	:UNIT FIELD ERROR
2026	001000	MDS= BIT9	:MULTIPLE DRIVE SELECT
2027	002000	PGE= BIT10	:PROGRAMMING ERROR
2028	004000	NEM= BIT11	:NON-EXISTENT MEMORY
2029	010000	NED= BIT12	:NON-EXISTENT DRIVE
2030	020000	UPE= BIT13	:UNIBUS PARITY ERROR
2031	040000	WCE= BIT14	:WRITE CHECK ERROR
2032	100000	DLT= BIT15	:DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

2036	000001	ILF= BIT0	:ILLEGAL FUNCTION CODE
2037	000002	SKI= BIT1	:SEEK INCOMPLETE
2038	000004	NXF= BIT2	:NON-EXECUTABLE DRIVE FUNCTION
2039	000010	DRPAR= BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
2040	000020	FMTE= BIT4	:FORMAT ERROR
2041	000040	DTYPE= BIT5	:DRIVE TYPE ERROR
2042	000100	ECH= BIT6	:ECC HARD
2043	000200	BSE= BIT7	:BAD SECTOR ERROR
2044	000400	HVRC= BIT8	:HEADER VRC ERROR
2045	001000	COE= BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
2046	002000	IDAE= BIT10	:INVALID DISK ADDRESS ERROR
2047	004000	WLE= BIT11	:WRITE LOCK ERROR
2048	010000	DTE= BIT12	:DRIVE TIMING ERROR
2049	020000	OPI= BIT13	:OPERATION (SEARCH) INCOMPLETE
2050	040000	UNS= BIT14	:DRIVE UNSAFE
2051	100000	DCK= BIT15	:DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

2055	000001	DRA= BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF : THIS BIT IS RESET)
2057	000004	OFST= BIT2	:DRIVE OFFSET
2058	000010	ACLO= BIT3	:AC LOW
2059	000020	SPDLSS= BIT4	:SPEED LOSS
2060	000040	DROT= BIT5	:DRIVE OFF TRACK
2061	000100	VV= BIT6	:VOLUME VALID
2062	000200	DRDY= BIT7	:DRIVE READY
2063	000400	DDT= BIT8	:DRIVE TYPE (0=RK06)
2064	004000	WRL= BIT11	:WRITE LOCK
2065	020000	PIP= BIT13	:POSITIONING IN PROGRESS
2066	040000	DSC= BIT14	:DRIVE STATUS CHANGE
2067	100000	SVAL= BIT15	:STATUS VALID


```
2068
2069      .SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION
2070
2071      000017      MESMSK= 17      ;MESSAGE MASK
2072
2073      000020      PAT= BIT4      ;FORCE EVEN PARITY ON SERCON MESSAGE LINES
2074      000040      DMD= BIT5      ;DIAGNOSTIC MODE
2075      000100      MSP= BIT6      ;MAINTENANCE SECTOR PULSE
2076      000200      MIND= BIT7     ;MAINTENANCE INDEX
2077      000400      MCLK= BIT8     ;MAINTENANCE CLOCK
2078      001000      MERD= BIT9     ;MAINTENANCE ENCODED READ DATA
2079      002000      MEWD= BIT10    ;MAINTENANCE ENCODED WRITE DATA
2080      004000      PCA= BIT11    ;PRECOMPENSATION ADVANCE
2081      010000      PCD= BIT12    ;PRECOMPENSATION DELAY
2082      020000      ECCW= BIT13   ;ECC WORD IS BEING READ OR WRITTEN
2083      040000      WRTGAT= BIT14 ;WRITE GATE
2084      100000      RDGATE= BIT15 ;READ GATE
2085
2086      .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
2087
2088      000040      S.DRA= BIT5     ;DRIVE AVAILIABLE
2089      000100      S.VV= BIT6     ;VOLUME VALID
2090      000200      S.DRY= BIT7    ;DRIVE READY
2091      000400      S.TYPE= BIT8   ;DRIVE TYPE
2092      001000      S.FORM= BIT9   ;DRIVE FORMAT
2093      002000      S.OFF= BIT10   ;OFFSET
2094      004000      S.WRL= BIT11   ;WRITE LOCK
2095      010000      S.SPIN= BIT12  ;SPINDLE ON
2096      020000      S.PIP= BIT13  ;POSITIONING IN PROGRESS
2097      040000      S.DSC= BIT14  ;DRIVE STATUS CHANGE
2098
2099      .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
2100
2101      000040      S.ICYL= BIT5    ;ILLEGAL CYLINDER ADDRESS
2102      000100      S.ACLO= BIT6   ;AC LOW
2103      000200      S.FLT= BIT7    ;DRIVE FAULT
2104      000400      S.ILF= BIT8    ;ILLEGAL FUNCTION
2105      001000      S.PAR= BIT9    ;DRIVE DETECTED SERCON PARITY ERROR
2106      002000      S.SKI= BIT10   ;SEEK INCOMPLETE
2107      004000      S.WLE= BIT11   ;WRITE LOCK ERROR
2108      010000      S.SPLS= BIT12  ;SPEED LOSS
2109      020000      S.DROT= BIT13  ;DRIVE OFF TRACK
2110      040000      S.UNS= BIT14  ;DRIVE UNSAFE
2111
2112      .SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
2113
2114      000020      S.XDOK= BIT4    ;TRANSDUCER OK
2115      000040      S.HDHM= BIT5   ;HEADS HOME
2116      000100      S.BRHM= BIT6   ;BRUSHES HOME
2117      000200      S.DOOR= BIT7   ;DOOR INTERLOCKED
2118      000400      S.CART= BIT8   ;CARTRAGE INTERLOCK
2119      001000      S.SPOK= BIT9   ;SPEED OK
2120      002000      S.FWD= BIT10   ;FORWARD
2121      004000      S.REV= BIT11   ;REVERSE
2122      010000      S.LOAD= BIT12  ;HEADS LOADING
2123      020000      S.RTZ= BIT13  ;RETURN TO ZERO
```

```
2124          040000          S.UNLD= BIT14          ;HEADS UNLOADING
2125
2126          .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
2127
2128          000020          S.SECT= BIT4          ;SECTOR ERROR
2129          000040          S.WCLK= BIT5          ;WRITE CLOCK AND NO WRITE GATE
2130          000100          S.WGAT= BIT6          ;WRITE GATE AND NO TRANSISTIONS
2131          000200          S.HDFL= BIT7          ;HEAD FAULT
2132          000400          S.MHD= BIT8          ;MULTIPLE HEAD SELECT
2133          001000          S.XERR= BIT9          ;INDEX ERROR
2134          002000          S.DIB= BIT10         ;DIBIT ERROR
2135          004000          S.PLO= BIT11         ;PLO ERROR
2136          010000          S.NMOV= BIT12        ;SEEK AND NO MOTION
2137          020000          S.LIMD= BIT13        ;LIMIT DETECT ON SEEK
2138          040000          S.BRKE= BIT14        ;SERVO-BRAKE
2139
2140          .SBTTL  COMMON MASKS
2141
2142          000007          M.DRV= 7          ;DRIVE CODE
2143          100000          M.PAR= BIT15         ;PARITY
2144          000003          M.ID= 3          ;BYTE ID
2145          017760          M.CDIF= 17760        ;CYLINDER DIFFERENCE/OFFSET
2146          017760          M.CADD= 17760        ;CYLINDER ADDRESS
2147          077770          M.SER= 77770        ;DRIVE SERIAL NUMBER
2148          000760          M.SECT= 760         ;SECTOR COUNT
2149          007000          M.HEAD= 7000        ;HEAD DECODE
2150          .SBTTL  TRAP CATCHER
2151
2152          000000          .=0
2153          ;*ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
2154          ;*A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
2155          ;*TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
2156          ;*THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
2157          ;*TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
2158          ;*LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
2159          ;*THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
2160          ;* PC=YYYYYY UNEXPECTED TRAP TO XXX
2161          ;*AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
2162          ;*WHERE XXX=LOCATION OF ILLEGAL TRAP
2163          ;* YYYYYY=PC AT TIME OF TRAP
2164          ;*NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
2165          ;* CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.
2166
2167          000000          000000          $40CAT: HALT          ;;HALT
2168          000002          000737          BR          .-100      ;;BRANCH TO 177700 & TIME OUT (NOT ON
2169          ;;11/05)
2170          000004          001772          .WORD          START    ;;VECTOR TO STARTING ADDRESS
2171          000006          000340          .WORD          340      ;;WITH PRIORITY LEVEL 7
2172          000174          000174          .=174
2173          000174          000000          DISPREG: .WORD          0      ;;SOFTWARE DISPLAY REGISTER
2174          000176          000000          SWREG: .WORD          0      ;;SOFTWARE SWITCH REGISTER
2175          .SBTTL  STARTING ADDRES(ES)
2176          000200          000137          001772          JMP          @#START ;;GO TO START OF PROGRAM
2177          000204          000137          001762          JMP          RESTRT    ;JUMP TO RESTART ROUTINE
2178          000214          000214          .=214
2179          000214          000137          001752          JMP          PARM      ;JUMP TO OPERATOR ASSIGNED PARMETERS
```

2180
2181 000220 000220 001742
2182
2183
2184
2185
2186
2187 000224
2188 000024 000200
2189 000024 000044
2190 000044 000224
2191 000044 000224
2192
2193
2194
2195
2196
2197 000224
2198 000224 000000
2199 000226 001276
2200 000230 000024
2201 000232 000074
2202 000234 000740
2203 000236 000031

```
      .=220  
      JMP      SETCLK      ; JUMP TO SET CLOCK ROUTINE  
.SBTTL APT PARAMETER BLOCK  
  
:*****  
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
:*****  
      . $X=      ;; 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  
      . = $X      ;; RESET LOCATION COUNTER  
:*****  
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
:INTERFACE SPEC.  
  
$APTHD:   
$HIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
$MBADR: .WORD $MAIL  ;; ADDRESS OF APT MAILBOX (BITS 0-15)  
$TSTM:  .WORD 20.    ;; RUN TIM OF LONGEST TEST  
$PASTM: .WORD 60.    ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
$UNITM: .WORD 480.   ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
      .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE (WORDS)
```

2204
2205
2206
2207
2208
2209
2210 001100
2211 001100
2212 001100 000000
2213 001102 000
2214 001103 000
2215 001104 000000
2216 001106 000000
2217 001110 000000
2218 001112 000000
2219 001114 000
2220 001115 001
2221 001116 000000
2222 001120 000000
2223 001122 000000
2224 001124 000000
2225 001126 000000
2226 001130 000000
2227 001132 000000
2228 001134 000
2229 001135 000
2230 001136 000000
2231 001140 177570
2232 001142 177570
2233 001144 177560
2234 001146 177562
2235 001150 177564
2236 001152 177566
2237 001154 000
2238 001155 002
2239 001156 012
2240 001157 000
2241 001160 000000
2242
2243 001162 000000
2244 001164 000000
2245 001166 000000
2246 001170 000000
2247 001172 000000
2248 001174 000000
2249 001176 000000
2250 001200 000000
2251 001202 000000
2252 001204 000000
2253 001206 000000
2254 001210 000000
2255 001212 000000
2256 001214 000000
2257 001216 000000
2258 001220 000000
2259 001222 000000

.SBTTL COMMON TAGS

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

. =1100

\$CMTAG:

.WORD 0
\$TSTNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
\$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
\$REGAD: .WORD 0
\$REG0: .WORD 0
\$REG1: .WORD 0
\$REG2: .WORD 0
\$REG3: .WORD 0
\$REG4: .WORD 0
\$REG5: .WORD 0
\$REG6: .WORD 0
\$REG7: .WORD 0
\$REG10: .WORD 0
\$REG11: .WORD 0
\$REG12: .WORD 0
\$REG13: .WORD 0
\$REG14: .WORD 0
\$REG15: .WORD 0
\$REG16: .WORD 0
\$REG17: .WORD 0
\$TMPO: .WORD 0

:: START OF COMMON TAGS

:: 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)
:: CONTAINS THE ADDRESS FROM
:: WHICH (\$REG0) WAS OBTAINED
:: CONTAINS ((\$REGAD)+0)
:: CONTAINS ((\$REGAD)+2)
:: CONTAINS ((\$REGAD)+4)
:: CONTAINS ((\$REGAD)+6)
:: CONTAINS ((\$REGAD)+10)
:: CONTAINS ((\$REGAD)+12)
:: CONTAINS ((\$REGAD)+14)
:: CONTAINS ((\$REGAD)+16)
:: CONTAINS ((\$REGAD)+20)
:: CONTAINS ((\$REGAD)+22)
:: CONTAINS ((\$REGAD)+24)
:: CONTAINS ((\$REGAD)+26)
:: CONTAINS ((\$REGAD)+30)
:: CONTAINS ((\$REGAD)+32)
:: CONTAINS ((\$REGAD)+34)
:: CONTAINS ((\$REGAD)+36)
:: USER DEFINED

```
2260 001224 000000 $TMP1: .WORD 0 ;;USER DEFINED
2261 001226 000000 $TMP2: .WORD 0 ;;USER DEFINED
2262 001230 000000 $TMP3: .WORD 0 ;;USER DEFINED
2263 001232 000000 $TMP4: .WORD 0 ;;USER DEFINED
2264 001234 000000 $TMP5: .WORD 0 ;;USER DEFINED
2265 001236 000000 $TMP6: .WORD 0 ;;USER DEFINED
2266 001240 000000 $TMP7: .WORD 0 ;;USER DEFINED
2267 001242 000000 $TMP10: .WORD 0 ;;USER DEFINED
2268 001244 000000 $TMP11: .WORD 0 ;;USER DEFINED
2269 001246 000000 $TMP12: .WORD 0 ;;USER DEFINED
2270 001250 000000 $TMP13: .WORD 0 ;;USER DEFINED
2271 001252 000000 $TMP14: .WORD 0 ;;USER DEFINED
2272 001254 000000 $TMP15: .WORD 0 ;;USER DEFINED
2273 001256 000000 $TMP16: .WORD 0 ;;USER DEFINED
2274 001260 000000 $TMP17: .WORD 0 ;;USER DEFINED
2275 001262 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
2276 001264 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
2277 001266 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
2278 001272 077 $QUES: .ASCII /?/ ;;QUESTION MARK
2279 001273 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
2280 001274 000012 $LF: .ASCIZ <12> ;;LINE FEED
2281 *****
2282 .SBTTL APT MAILBOX-ETABLE
2283 *****
2284 *****
2285 .EVEN
2286 001276 $MAIL: ;;APT MAILBOX
2287 001276 000000 $MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
2288 001300 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
2289 001302 000000 $TESTN: .WORD ATESTN ;;TEST NUMBER
2290 001304 000000 $PASS: .WORD APASS ;;PASS COUNT
2291 001306 000000 $DEVCT: .WORD ADEVCT ;;DEVICE COUNT
2292 001310 000000 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
2293 001312 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
2294 001314 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
2295 001316 $ETABLE: ;;APT ENVIRONMENT TABLE
2296 001316 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
2297 001317 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
2298 001320 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
2299 001322 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
2300 001324 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE, OPTIONS
2301 * BITS 15-11=CPU TYPE
2302 * 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2303 * 11/70=06,PDQ=07,Q=10
2304 * BIT 10=REAL TIME CLOCK
2305 * BIT 9=FLOATING POINT PROCESSOR
2306 * BIT 8=MEMORY MANAGEMENT
2307 001326 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS, M.S. BYTE
2308 001327 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE, BLK#1
2309 * MEM. TYPE BYTE -- (HIGH BYTE)
2310 * 900 NSEC CORE=001
2311 * 300 NSEC BIPOLAR=002
2312 * 500 NSEC MOS=003
2313 001330 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS, BLK#1
2314 * MEM. LAST ADDR. -3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
2315 001332 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS, M.S. BYTE
```

2316	001333	000	\$MTYP2:	.BYTE	AMTYP2	::MEM.TYPE,BLK#2
2317	001334	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
2318	001336	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
2319	001337	000	\$MTYP3:	.BYTE	AMTYP3	::MEM.TYPE,BLK#3
2320	001340	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
2321	001342	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
2322	001343	000	\$MTYP4:	.BYTE	AMTYP4	::MEM.TYPE,BLK#4
2323	001344	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
2324	001346	000210	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
2325	001350	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
2326	001352	177440	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
2327	001354	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
2328	001356	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
2329	001360		\$ETEND:			
2330			.MEXIT			

2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386

001360

001360 000000
001362 057236
001364 060610
001366 060702

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;* EM AND DH ARE ASCIZ DATA. EM IS ALWAYS A MESSAGE BUT DH
;* CAN BE A MESSAGE OR A SET OF COLUMN LABELS SPACED ACCROSS
;* THE PAGE, DT IS A STRING OF WORDS THAT POINT TO THE DATA TO
;* BE TYPED, AND DF IS A STRING OF WORK THAT TELL HOW THE DT WORDS
;* ARE TO BE TYPED. IF ANY OF THE POINTERS ARE NOT NEEDED, FOR A
;* PARTICULAR FORMAT, IT IS REPLACED WITH A ZERO.
;* THE NORMAL USAGE OF THE ERROR TABLE IS TO HAVE A TABLE ENTRY FOR
;* EACH ERROR MESSAGE THAT CAN OCCUR. IN THE INTEREST OF ECONOMICS
;* OF CORE MEMORY, THIS PROGRAM USES THE ERROR TABLE IN A
;* SLIGHTLY DIFFERENT MANNERS AS DESCRIBED BELOW.
;* THE ERROR TABLE ENTRIES MAKE UP A SET OF REPORT FORMATS THAT ARE USED
;* THROUGHOUT THE PROGRAM. WHEN AN ERROR IS TO BE REPORTED, THE
;* TABLE ENTRY THAT PROVIDES THE DESIRED FORMAT IS CHOSEN FROM
;* THE DEFINED SET. THE TABLE ENTRY CHOSEN IS THEN ALTERED
;* BY CHANGING THE FIRST (AND POSSIBLY THE SECOND) WORD TO CONTAIN
;* THE ADDRESS OF THE ASCIZ STRING THAT MAKES UP THE MESSAGE
;* PORTION OF THE REPORT. THE DATA FIELDS FOR THAT ENTRY ARE NEVER
;* CHANGED, NOR ARE THE COLUMN LABELS OR POSITIONS.
;* THE FORMAT THAT EACH TABLE ENTRY PROVIDES IS SHOWN BELOW WITH
;* THE DEFINITION OF THE ENTRY. ALL DATA FIELDS ARE TYPED IN OCTAL.

;* ;ERROR ITEM 1
;* (MESSAGE)
;* TST NUM ERR PC DRIVE
;* \$TESTN \$ERRPC DRVNUM

EM1N: .WORD 0
DH001
DT001
DF001

;* ;ERROR ITEM 2
;* (MESSAGE)
;* (MESSAGE)
;* TST NUM ERR PC DRIVE
;* \$TESTN \$ERRPC DRVNUM
;* RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA

	T.CS1	T.CS2	T.DS	T.ER	T.ASOF	T.DCVL	T.DA
2387							
2388	RKBA	RKWC					
2389	T.BA	T.WC					
2390							
2391	001370	000000					
2392	001372	000000					
2393	001374	060616					
2394	001376	060706					
2395							
2396							
2397							
2398							
2399							
2400							
2401							
2402	001400	000000					
2403	001402	057264					
2404	001404	060566					
2405	001406	060726					
2406							
2407							
2408							
2409							
2410							
2411							
2412							
2413							
2414							
2415							
2416							
2417							
2418							
2419							
2420							
2421							
2422							
2423							
2424							
2425							
2426							
2427							
2428							
2429							
2430							
2431							
2432							
2433							

EM2N: .WORD 0
DH2N: .WORD 0
DT002
DF002
;*ERROR ITEM 3
(MESSAGE)
TST NUM ERR PC DRIVE
\$TESTN \$ERRPC DRVNUM
RKCS1 RKCS2 RKDS RKER RKASOF RMR1
T.CS1 T.CS2 T.DS T.ER T.AST T.MR1

EM3N: .WORD 0
DH002A
DT003
DF003

;* ERROR ITEMS 4,5,6,87 ARE USED TO REPORT ERRORS THAT ARE THE RESULT
;* OF A HARDWARE ERROR INDICATOR BEING SET WHEN NOT EXPECTED,
;* NOT SET WHEN IT IS EXPECTED, OR BOTH. THE ERROR REPORT WILL
;* CONTAIN (1) ALL THE ERRORS THAT WERE DETECTED, (2) ALL THE EXPECTED
;* ERRORS THAT DID NOT OCCUR, OR (3) ALL THE EXPEDTED BUT NOT SET ERRORS
;* AND THE UNEXPECTED BUT SET ERRORS.
;* THE MESSAGE ITSELF EXPLAINS THE CIRCUMSTANCE FOR THE REPORT.
;* INCLUDED IN THE REPORT WILL BE ONE OR MORE OF THE FOLLOWING
;* STATEMENTS:
;* "THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:"
;* "THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:"
;* "THE ABOVE ARE ERRORS SET IN OPERATION:"
;* PRECEEDING ANY ONE OF THESE LINES WILL BE ONE OR MORE LINES THAT
;* SPECIFY TJE EXACT ERROR. FOLLOWING THE LAST LINE WILL BE A LINE
;* THAT IDENTIFIES THE OPERATION BEING PERFORMED.
;* EXAMPLE:
;* NON-EXISTANT DRIVE
;* THE ABOVE ARE ERRORS SET IN OPERATION:
;* DRIVE SELECT
;* (ADDITIONAL LINES OF INFORMATION)
;* THIS IS THE RESULT OF AN ERROR SET IN A SELECT OPERATION.

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
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489

:* EXAMPLE:
:* NON-EXISTANT DRIVE
:* THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:
:* DRIVE SELECT
:* (ADDITIONAL LINES OF INFORMATION)
:*
:* THIS IS THE RESULT OF AN EXPECTED ERROR THAT DID NOT OCCUR, I.E.
:* A NON-EXISTANT DRIVE WAS ADDRESSED BUT NED WAS NOT SET.

:* EXAMPLE:
:* NON-EXISTANT MEMORY
:* THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:
:* UNIBUS PARITY ERROR
:* THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:
:* WRITE DATA
:* (ADDITIONAL LINES OF INFORMATION)

:* THIS IS AN EXAMPLE OF NON-EXISTANT MEMORY BEING SET WHEN UNIBUS
:* PARITY ERROR WAS EXPECTED.

```

ERROR ITEM 4
(DESCRIPTION OF ERROR)
ERROR IN OPERATION
(DESCRIPTION OF OPERATION)
  IST NUM  ERR PC  DRIVE
  $TESTN  $ERRPC  DRVNUM
  RKCS1   RKCS2   RKDS   RKER   RKASOF   RKDCYL   RKDA
  T.CS1   T.CS2   T.DS   T.ER   T.ASOF   T.DCYL   T.DA
  RKBA    RKWC
  T.BA    T.WA
  A00     B00     A01     B01     A02     B02     A03     B03
  $REG10  $REG11  $REG12  $REG13  $REG14  $REG15  $REG16  $REG17

```

THE ERRORS REPORTED BY THIS FORMAT ARE:
CONTROLLER DETECTED DRIVE BUS ERROR
DRIVE DETECTED DRIVE BUS ERROR
SEEK INCOMPLETE
NON-EXECUTABLE DRIVE FUNCTION
DRIVE TIMING ERROR
DRIVE UNSAFE
AC LOW
SPINDLE SPEED LOSS
DRIVE OFF TRACK
ILLEGAL DRIVE ADDRESS ERROR
CYLINDER OVERFLOW
DRIVE TYPE ERROR
FORMAT ERROR
WRITE LOCK ERROR

001410 000000
001412 000000
001414 060616
001416 060736

EM4N: .WORD 0
DH4N: .WORD 0
DT004
DF004

:* ERROR ITEM 5
:* THIS ENTRY IS THE SAME AS ITEM 4 WITH THE ADDITION

```
2490      : *      OF A MESSAGE THAT FOLLOWS. THIS MESSAGE IS:
2491      : *
2492      : *      "ANY FIELD WITH ALL ONES MUST BE CONSIDERED INVALID"
2493      : *
2494      : *      THIS REPORT WILL BE PRINTED WHEN THE GATHERING OF DATA FOR
2495      : *      A00 THRU B03 IS NOT ACCOMPLISHED WITHOUT ERROR.
2496
2497 001420 000000      EM5N: .WORD 0
2498 001422 000000      DH5N: .WORD 0
2499 001424 060616      DT005
2500 001426 060766      DF005
2501
2502      : *      ERROR ITEM 6
2503      : *      (DESCRIPTION OF ERROR)
2504      : *      ERROR IN OPERATION
2505      : *      (DESCRIPTION OF OPERATION)
2506      : *      TST NUM ERR PC DRIVE
2507      : *      $TESTN $ERRPC DRVNUM
2508      : *      RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA
2509      : *      T.CS1 T.CS2 T.DS T.ER T.ASOF T.DCYL T.DA
2510      : *      RKBA RKWC
2511      : *      T.BA T.WC
2512
2513      : *      THE ERRORS REPORTED BY THIS FORMAT ARE:
2514      : *      DATA CHECK
2515      : *      WRITE CHECK
2516      : *      ECC HARD
2517      : *      DATA LATE
2518      : *      OPERATION INCOMPLETE
2519      : *      HEADER VRC ERROR
2520      : *      BAD SECTOR ERROR
2521
2522 001430 000000      EM6N: .WORD 0
2523 001432 000000      DH6N: .WORD 0
2524 001434 060616      DT006
2525 001436 061022      DF006
2526
2527      : *      ERROR ITEM 7
2528      : *      (DESCRIPTION OF ERROR)
2529      : *      ERROR IN OPERATION
2530      : *      (DESCRIPTION OF OPERATION)
2531      : *      TST NUM ERR PC DRIVE
2532      : *      $TESTN $ERRPC DRVNUM
2533      : *      RKCS1 RKCS2 RKDS RKER RKASOF
2534      : *      T.CS1 T.CS2 T.DS T.ER T.ASOF
2535
2536      : *      THE ERRORS REPORTED BY THIS FORMAT ARE:
2537      : *      NON-EXISTANT DRIVE
2538      : *      NON-EXISTANT MEMORY
2539      : *      CONTROLLER TIME OUT
2540      : *      UNIT FIELD ERROR
2541      : *      MULTIPLE DRIVE SELECT
2542      : *      PROGRAMMING ERROR
2543      : *      UNIBUS PARITY ERROR
2544      : *      ILLEGAL FUNCTION CODE
2545
```

```
2546      :*      DESCRIPTION OF OPERATION CAN BE ANY COMMAND, EITHER LEGAL OR ILLEGAL
2547
2548 001440 000000      EM7N:  .WORD  0
2549 001442 000000      DH7N:  .WORD  0
2550 001444 060616      DT007
2551 001446 061046      DF007
2552
2553      :*      ERROR ITEM 10
2554      :*      (DESCRIPTION OF ERROR)
2555      :*      ERROR AT COMPLETION OF OPERATION
2556      :*      (DESCRIPTION OF OPERATION)
2557      :*      TST NUM ERR PC  DRIVE
2558      :*      $TESTN $ERRPC  DRVNUM
2559      :*      EXPT  1S
2560      :*      $REG10 $REG11
2561
2562      :*      THE ERRORS REPORTED BY THIS FORMAT ARE SOFTWARE DETECTED BY
2563      :*      COMPARING EXPECTED RESULTS WITH ACTUAL RESULTS.  THE SPECIFIC
2564      :*      ERRORS ARE:
2565      :*      WORD COUNT INCORRECT
2566      :*      BUS ADDRESS INCORRECT
2567      :*      CYLINDER ADDRESS INCORRECT
2568      :*      TRACK ADDRESS INCORRECT
2569      :*      SECTOR ADDRESS INCORRECT
2570
2571 001450 000000      EM10N: .WORD  0
2572 001452 060077      DH010
2573 001454 060666      DT015
2574 001456 061066      DF010
2575
2576      :*      ERROR ITEM 11
2577      :*      (ERROR INDICATOR OR STATUS BIT)
2578      :*      NOT SET AS A RESULT OF
2579      :*      (ANOTHER ERROR INDICATOR, STATUS BIT, OR OPERATION)
2580      :*      TST NUM ERR PC  DRIVE
2581      :*      $TESTN $ERRPC  DRVNUM
2582      :*      RKCS1  RKCS2  RKDS  RKER  RKASOF  RKMR1
2583      :*      T.CS1  T.CS2  T.DS  T.ER  T.ASOF  T.MR1
2584
2585 001460 000000      EM11N: .WORD  0
2586 001462 060223      DH011
2587 001464 060616      DT010
2588 001466 061106      DF011
2589
2590      :*      ERROR ITEM 12
2591      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2592      :*      "NOT RESET AS A RESULT OF"
2593
2594 001470 000000      EM12N: .WORD  0
2595 001472 060252      DH012
2596 001474 060616      DT010
2597 001476 061106      DF011
2598
2599      :*      ERROR ITEM 13
2600      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2601      :*      "SET AS A RESULT OF"
```

```
2602
2603 001500 000000      EM13N: .WORD 0
2604 001502 060303      DH013
2605 001504 060616      DT010
2606 001506 061106      DF011
2607
2608      :*      ERROR ITEM 14
2609      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2610      :*      'RESET AS A RESULT OF'
2611 001510 000000      EM14N: .WORD 0
2612 001512 060326      DH014
2613 001514 060616      DT010
2614 001516 061106      DF011
2615
2616      :*      ERROR ITEM 15
2617      :*      (HEADER WORD MISCOMPARE) OR (DATA MISCOMPARE)
2618      :*      TST NUM ERR PC  DRIVE
2619      :*      $TESTN $ERRPC  DRVNUM
2620      :*      GOOD    BAD    WORD NUM
2621      :*      $REG10 $REG11 $REG12
2622
2623 001520 000000      EM15N: .WORD 0
2624 001522 057264      DH002A
2625 001524 060666      DT015
2626 001526 061130      DF015
2627
2628      :*      ERROR ITEM 16
2629      :*      ADDITIONAL LINES OF GOOD, BAD, WORD NUM FOR ERROR 15
2630
2631      0
2632      0
2633 001534 060674      DT015A
2634 001536 061140      DF016
2635
2636      .SBTTL REGISTER STORAGE FOR TEST
2637
2638 001540 000000      T.CS1: .WORD 0
2639 001542 000000      T.WC:  .WORD 0
2640 001544 000000      T.BA:  .WORD 0
2641 001546 000000      T.DA:  .WORD 0
2642 001550 000000      T.CS2: .WORD 0
2643 001552 000000      T.DS:  .WORD 0
2644 001554 000000      T.ER:  .WORD 0
2645 001556 000000      T.ASOF: .WORD 0
2646 001560 000000      T.DCYL: .WORD 0
2647 001562 000000      T.SPAR: .WORD 0
2648 001564 000000      T.DB:  .WORD 0
2649 001566 000000      T.MR1: .WORD 0
2650 001570 000000      T.ECPS: .WORD 0
2651 001572 000000      T.ECPT: .WORD 0
2652 001574 000000      T.MR2: .WORD 0
2653 001576 000000      T.MR3: .WORD 0
2654
2655      .SBTTL REGISTER SETUP STORAGE
2656 001600 000100      L.CS1: .WORD 100      ;PRESET WITH INTERRUPT ENABLE
2657 001602 000000      L.WC:  .WORD 0
```

2658	001604	000000	L.BA: .WORD	0	
2659	001606		L.DA:		
2660	001606	000	L.DS: .BYTE	0	
2661	001607	000	L.DT: .BYTE	0	
2662	001610	000000	L.CS2: .WORD	0	
2663	001612	000000	L.ASOF: .WORD	0	
2664	001614	000000	L.DCYL: .WORD	0	
2665	001616	000000	L.MR1: .WORD	0	
2666			.SBTTL	PROGRAM	DEFINED VARIABLES
2667					
2668	001620	000000	RKVEC: .WORD	0	:RK VECTOR
2669	001622	000000	RKPRI: .WORD	0	:RK PRIORITY
2670	001624	000000	SRTFLG: .WORD	0	:START FLAG
2671					: 0 = 200
2672					: 1 = 214
2673					: -1 = 204
2674	001626	000000	DRVNUM: .WORD	0	:DRIVE UNDER TEST
2675	001630	000000	DRVBIT: .WORD	0	:WORD TO STORE BIT TO INDICATE DRIVE UNDER TEST
2676	001632	000024	ERRCNT: .WORD	^D20	:ERROR COUNTER TO LIMIT ERROR
2677					:ERRORS REPORTED IN PROGRAM
2678	001634	000024	ERRLMT: .WORD	^D20	:DATA COMPARE ERROR LIMIT
2679	001636	061270	BSF24P: .WORD	BS24	:POINTER TO BAD SECTORS 24 SECTOR MODE
2680					: (FACTORY)
2681	001640	061144	BSF26P: .WORD	BS26	:POINTER TO BAD SECTORS 26 SECTOR MODE
2682					: (FACTORY)
2683	001642	000000	BSS24P: .WORD	0	:POINTER TO BAD SECTORS 24 SECT MODE
2684					: (SOFTWARE)
2685	001644	000000	BSS26P: .WORD	0	:POINTER TO BAD SECTORS 26 SECTOR MODE
2686					: (SOFTWARE)
2687	001646	000000	BS26CT: .WORD	0	:COUNT OF BAD SECTORS 26 SECTOR MODE
2688	001650	000000	BS24CT: .WORD	0	:COUNT OF BAD SECTORS 24 SECTOR MODE
2689	001652	000764	MILCNT: .WORD	^D500	:COUNT TO APPROXIMATE 1 MILL SEC
2690	001654	000017	TIMCNT: .WORD	^D15	:COUNTER FOR MYTIME ROUTINE
2691		001660		.=1660	
2692	001660	000000		.WORD	0
2693	001662	000000	INTSET: .WORD	0	:NON-ZERO IF RK06 INTERRUPT SINCE
2694					:LAST CLEARED
2695	001664	000000	OPTFLG: .WORD	0	:OPTION FLAGS
2696					
2697		000001		DOTST= BIT0	:DRIVE 0 TO BE TESTED FLAG
2698		000002		MEMSZB= BIT1	:MEMORY SIZE REPORT FLAG
2699		000004		MEMPYB= BIT2	:MEMORY PARITY REPORT FLAG
2700		000010		SRTINS= BIT3	:START UP INSTRUCTIONS REPORTED FLAG
2701		000100		PARPRE= BIT6	:PARITY PRESENT FLAG
2702		000200		BSEBPT= BIT7	:BSE HAS BEEN REPORTED
2703		000400		FPFMT= BIT8	:FIRST PASS FORMAT SWITCH
2704		002000		CP1170= BIT10	:CP IS 11/70 FLAG
2705		004000		DRVRPT= BIT11	:DRIVE NUMBERS REPORTED FLAG
2706		100000		LCLKPR= BIT15	:LINE CLOCK PRESENT
2707					
2708	001666	000000	DRVDRP: .WORD	0	:LIST OF DRIVES DROPPED
2709	001670	000000	MEMPAR: .WORD	0	:WORD OF PARITY FLAGS
2710	001672	000000	CSRPTR: .WORD	0	:POINTER TO CSR TO USE FOR SETTING BAD
2711					: PARITY
2712	001674	000000	LCLKTK: .WORD	0	:LINE CLOCK TICK COUNTER
2713	001676	000000	REFMT: .WORD	0	:REFORMAT SWITCH FOR HALT

```
2714  
2715 : THE FOLLOWING 4 VARIABLES ARE USED TO STORE PARAMETERS FOR  
2716 : HEADER OR DATA COMPARE CONTINUATION PROCESS.  
2717 001700 000000 DESHLD: .WORD 0 ;DESTINATION HOLD  
2718 001702 000000 SRCHLD: .WORD 0 ;SOURCE HOLD  
2719 001704 000000 WRDNUM: .WORD 0 ;WORD NUMBER IN ERROR HOLD  
2720 001706 000000 WRDCNT: .WORD 0 ;WORDS LEFT IN COMPARE HOLD  
2721 001710 177546 KWLADD: .WORD 177546 ;KW11-L ADDRESS  
2722 001712 000100 KWLVEC: .WORD 100 ;KW11-L VECTOR  
2723 001714 172100 MEMBAS: .WORD 172100 ;MM11 ADDRESS  
2724 001716 000114 MMVECA: .WORD 114 ;MM11 VECTOR  
2725  
2726 001720 000000 DTYPE: 0 ;DRV TYP 0=RK06, 2000=RK07  
2727 001722 000000 TYPTBL: 0 ;DRV 0 0=RK06, 2000=RK07  
2728 001724 000000 0 ;DRV 1  
2729 001726 000000 0 ;DRV 2  
2730 001730 000000 0 ;DRV 3  
2731 001732 000000 0 ;DRV 4  
2732 001734 000000 0 ;DRV 5  
2733 001736 000000 0 ;DRV 6  
2734 001740 000000 0 ;DRV7  
2735  
2736  
2737  
2738  
2739  
2740 .SBTTL PROGRAM SETUP  
2741  
2742 001742 012737 000002 001624 SETCLK: MOV #2,SRTFLG ;SET START FLAG FOR CLOCK ADJUST  
2743 001750 000412 BR START1  
2744  
2745 001752 012737 000001 001624 PARM: MOV #1,SRTFLG ;SET START FLAG FOR PARMETER START  
2746 001760 000406 BR START1  
2747  
2748 001762 012737 177777 001624 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR PARMETER START  
2749 001770 000402 BR START1  
2750  
2751 001772 005037 001624 START: CLR SRTFLG ;CLEAR START FLAG  
2752 001776 000005 START1: RESET ;RESET THE WHOLE SYSTEM  
2753 002000 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER  
2754 002004 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS  
2755 002010 012746 002016 MOV #1$,-(SP) ;LOAD START OF PROGRAM  
2756 002014 000002 RTI ;LOAD PSW  
2757  
2758 002016 004737 045220 1$: JSR PC,$TKINT ;INITIALIZE KEYBOARD  
2759 002022 005037 001676 CLR REFORMAT ;CLEAR REFORMAT SWITCH  
2760 .SBTTL INITIALIZE THE COMMON TAGS  
2761 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA  
2762 002026 012706 001100 MOV #$CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED  
2763 002032 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION  
2764 002034 022706 001140 CMP #SWR,R6 ;;DONE?  
2765 002040 001374 BNE .-6 ;;LOOP BACK IF NO  
2766 002042 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER  
2767 ;;INITIALIZE A FEW VECTORS  
2768 002046 012737 032576 000020 MOV #$$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE  
2769 002054 012737 000340 000022 MOV #340,@#IOTVEC+2 ;;LEVEL 7
```

```

2770 002062 012737 033614 000030      MOV    #$ERROR,@#EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
2771 002070 012737 000340 000032      MOV    #340,@#EMTVEC+2  ;;LEVEL 7
2772 002076 012737 047076 000034      MOV    #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2773 002104 012737 000340 000036      MOV    #340,@#TRAPVEC+2;LEVEL 7
2774 002112 012737 046720 000024      MOV    #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2775 002120 012737 000340 000026      MOV    #340,@#PWRVEC+2  ;;LEVEL 7
2776 002126 013737 032026 032020      MOV    $ENDCT,$EOPCT    ;;SETUP END-OF-PROGRAM COUNTER
2777 002134 005037 001262                CLR    $TIMES           ;;INITIALIZE NUMBER OF ITERATIONS
2778 002140 005037 001264                CLR    $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2779 002144 112737 000001 001115      MOV    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
2780 002152 012737 002152 001106      MOV    #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2781 002160 012737 002160 001110      MOV    #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
2782                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2783                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2784 002166 013746 000004                MOV    @#ERRVEC,-(SP)   ;;SAVE ERROR VECTOR
2785 002172 012737 002226 000004                MOV    #64,$@#ERRVEC   ;;SET UP ERROR VECTOR
2786 002200 012737 177570 001140                MOV    #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
2787 002206 012737 177570 001142                MOV    #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2788 002214 022777 177777 176716                CMP    #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2789 002222 001012                BNE    66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2790                ;;AND THE HARDWARE SWR IS NOT = -1
2791 002224 000403                BR     65$             ;;BRANCH IF NO TIMEOUT
2792 002226 012716 002234 64$:                MOV    #65$,(SP)      ;;SET UP FOR TRAP RETURN
2793 002232 000002                RTI
2794 002234 012737 000176 001140 65$:                MOV    #SWREG,SWR     ;;POINT TO SOFTWARE SWR
2795 002242 012737 000174 001142                MOV    #DISPREG,DISPLAY
2796 002250 012637 000004 66$:                MOV    (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2797
2798 002254 005037 001304                CLR    $PASS          ;;CLEAR PASS COUNT
2799 002260 132737 000200 001317                BITB  #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
2800 002266 001403                BEQ    67$            ;;YES,USE NON-APT SWITCH
2801 002270 012737 001320 001140                MOV    #SSWREG,SWR    ;;NO,USE APT SWITCH REGISTER
2802 002276
2803 67$:
2804 .SBTTL TYPE PROGRAM NAME
2805 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2806 002276 005227 177777                INC    #-1            ;;FIRST TIME?
2807 002302 001046                BNE    68$            ;;BRANCH IF NO
2808 002304 022737 032162 000042                CMP    #SENDAD,@#42  ;;ACT-11?
2809 002312 001442                BEQ    68$            ;;BRANCH IF YES
2810 002314 104401 002362                TYPE  ,69$           ;;TYPE ASCIZ STRING
2811 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2812 002320 005737 000042                TST   @#42           ;;ARE WE RUNNING UNDER XXDP/ACT?
2813 002324 001012                BNE    70$            ;;BRANCH IF YES
2814 002326 123727 001316 000001                CMPB  $ENV,#1        ;;ARE WE RUNNING UNDER APT?
2815 002334 001406                BEQ    70$            ;;BRANCH IF YES
2816 002336 023727 001140 000176                CMP    SWR,#SWREG    ;;SOFTWARE SWITCH REG SELECTED?
2817 002344 001005                BNE    71$            ;;BRANCH IF NO
2818 002346 104406                GTSWR                ;;GET SOFT-SWR SETTINGS
2819 002350 000403                BR     71$
2820 002352 112737 000001 001134 70$:                MOV    #1,$AUTOB     ;;SET AUTO-MODE INDICATOR
2821 002360 000417                BR     68$
2822                ;;69$:
2823                .ASCIZ <CRLF>*(CZR6KGO RK6 FCTNL CTRL DIAG*<CRLF>
2824 002420 132737 000200 001317 68$:
2825 002426 001061                BITB  #BIT7,$ENVM    ;;TEST IF DO NOT SIZE
                BNE    3$            ;;YES - SKIP

```

```
2826 002430 004737 032234 JSR PC,$SIZE
2827 002434 023727 032574 000740 CMP $LSTBK,#740 ;MAKE SURE MEMORY IS SUFFICIENT
2828 002442 103007 BHIS 2$ ;YES - SKIP
2829 002444 104401 051165 TYPE ,OPR005 ;MESSAGE (NOT ENOUGH MEMORY)
2830 002450 012737 000001 032020 MOV #1,$EOPCT ;FORCE END OF PROGRAM
2831 002456 000137 031772 JMP $EOP
2832 002462 2$:
2833 002462 012700 070414 MOV #IBUFF,R0 ;SET UP TO CLEAR 2000
2834 002466 012701 002000 MOV #2000,R1 ;WORDS OF BUFFER SPACE
2835 002472 005020 30$: CLR (R0)+ ;LOOP -
2836 002474 005301 DEC R1 ;UNTIL -
2837 002476 001375 BNE 30$ ;DONE
2838 002500 122737 000013 000041 CMPB #13,@#41 ;LOAD FROM XXDP
2839 002506 001003 BNE 99$ ;BRANCH IF NOT
2840 002510 104401 051236 TYPE ,OPR007
2841 002514 000000 HALT
2842 002516 99$:
2843 002516 013700 032574 MOV $LSTBK,R0 ;GET LAST BANK
2844 002522 012701 000006 MOV #6,R1 ;SET SHIFT COUNT
2845 002526 013703 032572 MOV $LSTAD,R3 ;GET LAST ADDRESS
2846 002532 005004 CLR R4 ;CLEAR R4 FOR OVERFLOW
2847 002534 005737 032316 TST $KT11 ;MEM MANAGE PRESENT?
2848 002540 100005 BPL 23$ ;NO - SKIP
2849 002542 006100 22$: ROL R0 ;SHIFT BANK LEFT
2850 002544 006104 ROL R4 ;ADD IN CARRY
2851 002546 005301 DEC R1 ;DECREMENT COUNT
2852 002550 001374 BNE 22$ ;LOOP IF NOT ZERO
2853 002552 050003 BIS R0,R3 ;SET BANK BITS IN LAST ADDRESS
2854 002554 112737 000001 001327 23$: MOVB #1,$MTYP1 ;FORCE MEMORY TYPE TO 1
2855 002562 110437 001326 MOVB R4,$MAMS1 ;STORE UPPER MEMORY ADDRESS
2856 002566 010337 001330 MOV R3,$MADR1 ;STORE LOWER ADDRESS
2857 002572 032737 000010 001664 3$: BIT #SRTINS,OPTFLG ;TEST IF ALREDY REPORTED
2858 002600 001005 BNE 24$ ;YES - SKIP
2859 002602 104401 051766 TYPE ,OPR016 ;TYPE STARTUP INSTRUCTIONS
2860 002606 052737 000010 001664 BIS #SRTINS,OPTFLG ;SET REPORTED FLAG
2861 002614 24$:
2862 002614 022737 000001 001624 CMP #1,SRTFLG ;CHECK IF PARAMETER START
2863 002622 001122 BNE 15$ ;NO, START TESTING
2864 002624 104401 051056 5$: TYPE ,OPR001 ;TYPE 'RK611 BUS ADDRESS ( ) ='
2865 002630 013746 001352 MOV $BASE,-(SP) ;:SAVE $BASE FOR TYPEOUT
2866 002634 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2867 002636 104401 051105 TYPE ,OPR002
2868 002642 104412 RDOCT ;GET VALUE
2869 002644 012637 001222 MOV (SP)+,$TMP0
2870 002650 001407 BEQ 7$ ;CHECK IF <CR>
2871 002652 022737 160000 001222 CMP #160000,$TMP0 ;CHECK IF IN I/O PAGE
2872 002660 101361 BHI 5$
2873 002662 013737 001222 001352 MOV $TMP0,$BASE ;LOAD NEW BUS ADDRESS
2874 002670 104401 051113 7$: TYPE ,OPR003 ;TYPE 'RK611 VECTOR ADDRESS ( ) ='
2875 002674 013746 001346 MOV $VECT1,-(SP) ;GET $VECT1 FOR TYPOUT
2876 002700 042716 160000 BIC #160000,(SP) ;CLEAR PRIORITY BITS
2877 002704 104402 TYPOC
2878 002706 104401 051105 TYPE ,OPR002
2879 002712 104412 RDOCT ;GET VALUE
2880 002714 012637 001222 MOV (SP)+,$TMP0
2881 002720 001412 BEQ 10$ ;CHECK IF <CR>
```



```
288 002722 022737 001000 001222      CMP      #1000,$TMP0
2883 002730 101757          BLOS     7$           ;CHECK IF LEGAL
2884 002732 042737 017777 001346      BIC      #17777,$VECT1 ;CLEAR OLD VECTOR
2885 002740 053737 001222 001346      BIS      $TMP0,$VECT1 ;LOAD NEW VECTOR ADDRESS
2886 002746 104401 051143          TYPE     ,OPR004      ;TYPE 'RK611 PRIORITY ( ) ='
2887 002752 005046          CLR      -(SP)
2888 002754 113716 001347          MOVB    $VECT1+1,(SP)
2889 002760 006216          ASR     (SP)          ;SHIFT 5 BITS RIGHT
2890 002762 006216          ASR     (SP)
2891 002764 006216          ASR     (SP)
2892 002766 006216          ASR     (SP)
2893 002770 006216          ASR     (SP)
2894 002772 104402          TYP0C
2895 002774 104401 051105          TYPE     ,OPR002
2896 003000 104412          RDOCT
2897 003002 012637 001222          MOV     (SP)+,$TMP0 ;GET VALUE
2898 003006 001430          BEQ     15$          ;CHECK IF <CR>
2899 003010 022737 000007 001222          CMP     #7,$TMP0    ;CHECK IF LEGAL
2900 003016 103753          BLO     10$
2901 003020 022737 000004 001222          CMP     #4,$TMP0
2902 003026 101347          BHI     10$
2903 003030 006337 001222          ASL     $TMP0        ;SHIFT 5 BITS LEFT
2904 003034 006337 001222          ASL     $TMP0
2905 003040 006337 001222          ASL     $TMP0
2906 003044 006337 001222          ASL     $TMP0
2907 003050 006337 001222          ASL     $TMP0
2908 003054 042737 160000 001347          BIC     #160000,$VECT1+1 ;CLEAR OLD PRIORITY
2909 003062 053737 001222 001347          BIS     $TMP0,$VECT1+1 ;LOAD RK611 PRIORITY
2910 003070 005037 001304          CLR     $PASS        ;SET PASS COUNT TO ZERO
2911 003074 005037 001666          CLR     DRVDRP      ;CLEAR DROPPED DRIVES LIST
2912 003100 042737 004000 001664          BIC     #DRVDRPT,OPTFLG ;CLEAR DRIVE #'S REPORTED FLAG
2913 003106 004737 034522          JSR     PC,OPTTST    ;SETUP PARITY CHECK & CLOCK
2914 003112 013700 001346          MOV     $VECT1,R0    ;STORE VECTOR FOR USE
2915 003116 042700 160000          BIC     #160000,R0   ;CLEAR PRIORITY BITS
2916 003122 010037 001620          MOV     R0,RKVEC
2917 003126 012710 034442          MOV     #INTHLR,(R0) ;SETUP INTERRUPT ADDRESS
2918 003132 113737 001347 001622          MOVB   $VECT1+1,RKPRI ;STORE PRIORITY FOR USE
2919 003140 013702 001352          MOV     $BASE,R2    ;SET BASE ADDRESS
2920 003144 005037 001264          CLR     $ESCAPE     ;CLEAR ESCAPE
2921 003150 012746 000000          NEWPAS: MOV    #PRO,-(SP) ;SET PRIORITY
2922 003154 012746 003162          MOV    #16$,-(SP)
2923 003160 000002          RTI
2924 003162
2925
2926          .SBTTL **BASIC INTERFACE AND OPTION TESTS
2927          ;*****
2928          ;*TEST 1          RK611 BASE ADDRESS TEST
2929          ;*          CHECK THAT READING THE RK611 BASE ADDRESS (RKCS1) DOES NOT
2930          ;*          CAUSE A NON-EXISTANT MEMORY TRAP. IF A TRAP OCCURS
2931          ;*          THE PROGRAM IS HALTED.
2932          ;*****
2933          ;*****
2934 003162 000004          TST1:  SCOPE
2935 003164 012737 000100 001262          MOV     #100,$TIMES ;DO 100 ITERATIONS
2936 003172 012706 001100          MOV     #STACK,SP  ;CLEAN OFF STACK
2937 003176 012701 000004          MOV     #4,R1      ;SET POINTER TO VECTOR
```

```
2938 003202 012146      MOV      (R1)+,-(SP)      ;STORE OLD VECTOR CONTENTS
2939 003204 011146      MOV      (R1)-,(SP)
2940 003206 012701 000004  MOV      #4,R1           ;RESET POINTER
2941 003212 012721 034434  MOV      #NEXINT,(R1)+    ;SET VECTOR TO NEM TEST HANDLER
2942 003216 012711 000340  MOV      #PR7,(R1)       ;SET PRIORITY
2943 003222 013702 001352  MOV      $BASE,R2       ;SET POINTER TO RK611 BASE ADDRESS
2944 003226 005037 001662  CLR      INTSET         ;CLEAR INTERRUPT COUNTER
2945 003232 012762 000000 000000  MOV      #0,RKCS1(R2)    ;WRITE CS1 TO SEE IN NEM WILL SET
2946 003240 000240      NOP
2947 003242 000240      NOP
2948 003244 000240      NOP
2949 003246 005737 001662  TST      INTSET         ;TEST IF COUNTER IS 0
2950 003252 001406      BEQ      1$            ;YES - SKIP ERROR REPORT
2951 003254 012737 053405 001360  MOV      #EM1,EM1N      ;MESSAGE (NON-EXISTANT MEMORY TRAP ERR)
2952 003262 104001      ERROR 1
2953 003264 000137 043724  JMP      CTRHLT        ;GO TO CONTROLLED HALT
2954 003270 012701 000006 1$:      MOV      #6,R1           ;RESTORE VECTOR
2955 003274 012611      MOV      (SP)+,(R1)
2956 003276 012641      MOV      (SP)+,-(R1)
```

```
2957
2958
2959 *****
2960 :*TEST 2      INTERRUPT VECTOR ADDRESS TEST
2961 :*          CHECK THAT THE INTERRUPT VECTOR FOR THE RK611 IS SET TO THE
2962 :*          EXPECTED ADDRESS. IF INTERRUPT VECTOR IS IN ERROR,
2963 :*          THE PROGRAM IS HALTED.
```

```
2964 *****
2965 TST2:      SCOPE
2966 003302 012737 000100 001262  MOV      #100,$TIMES    ;;DO 100 ITERATIONS
2967 003310 012762 005000 000010  MOV      #CLR,RKCS2(R2) ;CLEAR SUBSYSTEM, SPECIFICALLY TO
2968                                ;CLEAR ANY OLD INTERRUPTS
2969 003316 005037 001662  CLR      INTSET         ;CLEAR INTERRUPT COUNTER
2970 003322 012762 000300 000000  MOV      #RDY!IE,RKCS1(R2) ;WRITE CS1 TO FORCE INTERRUPT
2971 003330 000240      NOP
2972 003332 000240      NOP
2973 003334 000240      NOP
2974 003336 005737 001662  TST      INTSET         ;TEST IF INTERRUPT OCCURRED
2975 003342 001011      BNE      3$            ;YES - SKIP ERROR REPORT
2976 003344 105737 001103  TSTB    $ERFLG        ;TEST IF ERFLG ALREADY SET. IF SO THE
2977                                ;INTERRUPT WENT TO THE WRONG VECTOR
2978                                ;AND MESSAGE HAS BEEN REPORTED.
2979                                ;THEREFORE - EXIT
2979 003350 001004      BNE      2$            ;MESSAGE (NO INTERRUPT)
2980 003352 012737 053405 001360  MOV      #EM1,EM1N
2981 003360 104001      ERROR 1
2982 003362 000137 043724 2$:      JMP      CTRHLT        ;GO TO CONTROLLED HALT
2983 003366 3$:
```

```
2984
2985 .SBTTL  **STATUS VALID TESTS
```

```
2986
2987 *****
2988 :*TEST 3      SELECT ALL DRIVES
2989 :*
2990 :*          IF NOT RUNNING IN APT AUTOMATIC ENVIRONMENT,
2991 :*          DETERMINE WHAT DRIVES ARE ON-LINE BY
2992 :*          SELECTING ALL DRIVES. IF NON-EXISTENT DRIVE REPORTED
2993
```

2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030 003366 000004
3031 003370 012737 000062 001262
3032 003376 005037 001720
3033 003402 104416
3034 003404 104003
3035
3036 003406 012746 000000
3037 003412 012746 003420
3038 003416 000002
3039
3040 003420 013701 001620
3041 003424 012721 034442
3042 003430 012711 00C340
3043 003434 012703 001354
3044 003440 132737 000200 001317
3045 003446 001007
3046 003450 005737 001304
3047 003454 001004
3048 003456 032737 004000 001664
3049 003464 001402

```

:* MAKE SURE STATUS VALID IS RESET. IF DRIVE
:* PRESENT MAKE SURE NO ERROR EXISTS, DRIVE
:* IS CYCLED UP, AND STATUS VALID SET, AND DSC RESET.
:*
:* IF RUNNING IN APT AUTOMATIC ENVIRONMENT, THE DRIVES
:* IDENTIFIED IN ETABLE ARE TESTED FOR NO ERROR, DRIVE
:* CYCLED UP, AND STATUS VALID SET.
:*
:* IF LOCATION 41 INDICATES THE XXDP MEDIA IS ON
:* THE RK06, DRIVE 0 WILL ONLY BE TESTED IF THE PARAM
:* START (214) WAS USED. IF THE AUTOMATIC START (200)
:* IS USED, DRIVE 0 IS NOT TESTED. THE RESTART (204)
:* WILL RETAIN THE TEST STATUS OF DRIVE 0.
:*
:* IF THE PARAM START IS USED, THE OPERATOR MUST
:* EITHER PLACE DRIVE 0 OFF LINE IF IT IS NOT TO BE TESTED
:* OR UNLOADED AND A SCRATCH MEDIA MOUNTED IF IT IS TO
:* BE TESTED. THE PROGRAM WILL MONITOR OFF LINE AND VOLUME
:* VALID TO DETERMINE THE TEST STATUS OF DRIVE 0.
:*
:* ALL DRIVES TO BE TESTED MUST BE ON-LINE, CYCLED UP, AND
:* WRITE LOCK RESET. ADDRESSES OF DRIVES THAT ARE NON-
:* EXISTANT EITHER BECAUSE THE DRIVE DOES NOT EXIST OR IS OFF-
:* LINE ARE USED TO VERIFY NON-EXISTANT DRIVE ERROR DETECTION.
:* DRIVES THAT ARE ON-LINE BUT NOT CYCLED UP OR ARE WRITE
:* LOCKED ARE NOT TESTED.
:*
:* AT COMPLETION OF THE TEST A MESSAGE WILL BE GIVEN TO
:* IDENTIFY THE DRIVES TO BE USED IN TESTING.
:*
:* NOTE: THIS TEST MUST BE RUN AT LEAST ONCE BEFORE
:* ANY OTHER TEST THAT FOLLOWS.
:* THE DETERMINATION OF WHETHER EACH DRIVE IS AN
:* RK06 OR AN RK07 IS MADE IN THIS TEST, AND A
:* TABLE IS MADE ACCORDINGLY.

```

```

*****
TST3: SCOPE
MOV #50,$TIMES ;:DO 50. ITERATIONS
CLR DTYPE ;:ASSUME RK06
TSSINIT ;:CALL SUBSYSTEM CLEAR AND TEST
ERROR 3
MOV #PRO,-(SP) ;:SET PROCESSOR PRIORITY TO ALLOW
MOV #1$,-(SP) ;:RK611 INTERRUPTS
RTI
1$: MOV RKVEC,R1 ;:GET VECTOR
MOV #INTHLR,(R1)+ ;:LOAD INTERRUPT VECTOR
MOV #PR7,(R1)
MOV #$DEVN,R3 ;:GET ADDRESS OF DEVICE MAP
BITB #BIT7,$ENVN ;:TEST IF SHOULD SIZE
BNE 50$ ;:NO - SKIP DRIVE SIZING.
TST $PASS ;:TEST IF FIRST PASS
BNE 50$ ;:NO - SKIP TO DRIVE SELECT TEST
BIT #DRVRPT,OPTFLG ;:TEST IF DRIVE #'S REPORTED
BEQ 92$ ;:NO - GOTO SIZING

```

3050	003466	000137	004142		50\$:	JMP	11\$		
3051	003472	005013			92\$:	CLR	(R3)		:CLEAR DEVICE MAP
3052	003474	123727	000041	000013		CMPB	@#41,#13		:TEST IF RK06 IS LOAD DEVICE
3053	003502	001111				BNE	77\$:NO - SKIP
3054	003504	022737	000001	001624		CMP	#1,SRTFLG		:WAS START AT PARAM?
3055	003512	001411				BEQ	2\$:YES - SKIP
3056	003514	104401	051236			TYPE	,OPR007		:NO TEST OF DRIVE 0
3057	003520	052737	000001	001666		BIS	#BIT0,DRVDRP		:SET DRIVE 0 DROPPED
3058	003526	042737	000001	001664		BIC	#DOTST,OPTFLG		:DR FLAG - NO TEST DRIVE 0
3059	003534	000477				BR	7\$		
3060	003536	104401	051236		2\$:	TYPE	,OPR006		:MESSAGE - SWAP PACK ON DRIVE OFF LINE.
3061	003542	005037	001610			CLR	L.CS2		:SET TO DRIVE 0
3062	003546	005037	001232			CLR	\$TMP4		:CLEAR FOR USE AS A SWITCH
3063	003552	012737	000101	001600	3\$:	MOV	#SELDLV,L.CS1		:LOAD FOR SELECT
3064	003560	012737	003646	001264		MOV	#4,\$ESCAPE		:SET UP ESCAPE IN CASE OF ERR
3065	003566	104417				TLOADRK			:LOAD RK & DO SELECT
3066									
3067	003570	104423				TWAT16			:WAIT 16MS FOR COMPLETION
3068	003572	104002				ERROR	2		:NOT DONE ON TIME
3069									
3070	003574	104420				TGETRK			:GET RK REGISTER
3071	003576	032737	100000	001540		BIT	#CERR,T.CS1		:TEST IF CERR
3072	003604	001431				BEQ	5\$:NO - SKIP
3073	003606	032737	010000	001550		BIT	#NED,T.CS2		:TEST IF NED
3074	003614	001014				BNE	4\$:YES - SKIP
3075	003616	032737	000040	001554		BIT	#DTYE,T.ER		:ELSE SEE IF DRIVE TYPE ERROR
3076	003624	001406				BEQ	26\$:BR IF NO
3077									
3078	003626	012737	002000	001720		MOV	#CDT,DTYPE		:ELSE SETUP FOR RK07
3079									
3080	003634	104416				TSSINIT			:INIT SUBSYS
3081	003636	104003				ERROR	3		:NOT SUCCESSFUL
3082	003640	000744				BR	3\$:AND TRY AGAIN
3083									
3084	003642	104421			26\$:	TCHKOP			:CHECK THE OPERATION AND REPORT THE ERROR
3085	003644	104004				ERROR	4 ;OR5,6,7		:AFTER THE ERROR IS REPORTED THE TEST
3086									:IS ABORTED
3087	003646	104401	051236		4\$:	TYPE	,OPR007		:TYPE NO TEST OF DRIVE 0
3088	003652	042737	000001	001664		BIC	#DOTST,OPTFLG		:DR FLAG - NO TEST OF DRIVE 0
3089	003660	052737	000001	001666		BIS	#BITC,DRVDRP		:SET DRV 0 AS DROPPED
3090	003666	000422				BR	7\$:SKIP OVER WAIT FOR PACK MOUNT
3091	003670	005737	001232		5\$:	TST	\$TMP4		:TEST FLAG DRIVE READY HAS RESET
3092	003674	001010				BNE	6\$:YES - SKIP TO CHECK IF IT IS SET AGAIN
3093	003676	032737	000200	001552		BIT	#DRDY,T.DS		:ELSE CHECK READY
3094	003704	001322				BNE	3\$:STILL SET - GET STATUS AGAIN
3095	003706	012737	177777	001232		MOV	#-1,\$TMP4		:ELSE SET FLAG TO INDICATE READY WENT LOW
3096	003714	000716				BR	3\$:GO GET STATUS AGAIN
3097									
3098	003716	032737	000200	001552	6\$:	BIT	#DRDY,T.DS		:TEST IF READY SET AGAIN
3099	003724	001712				BEQ	3\$:NO - GO GET STATUS AGAIN
3100	003726	052737	000001	001664	77\$:	BIS	#DOTST,OPTFLG		:ELSE SET DRV 0 TEST FLAG.
3101									
3102	003734	012737	004042	001264	7\$:	MOV	#35,\$ESCAPE		:SET UP ESCAPE IN CASE OF ERR
3103	003742	005000				CLR	R0		:CLEAR FOR DRIVE NUMBER COUNTER
3104	003744	012701	000001			MOV	#1,R1		:SET BIT 0 AS DRIVE SELECTOR
3105	003750	005037	001720			CLR	DTYPE		:INIT TO RK06

3106										
3107	003754	032737	000001	001664		BIT	#DOTST,OPTFLG		:TEST DRIVE 0?	
3108	003762	001440				BEQ	9\$:NO - SKIP	
3109										
3110	003764	104416			8\$:	TSSINIT			:INITIALZE SUBSYSTEM	
3111	003766	104003				ERROR	3		:ERROR IF NOT SUCCESSFUL	
3112										
3113	003770	010037	001610			MOV	R0,L,CS2		:LOAD DRIVE NUMBER	
3114	003774	012737	000101	001600		MOV	#SELDRV,L,CS1		:LOAD DRIVE SELECT	
3115	004002	104417				TLOADRK			:LOAD RK REGS	
3116										
3117	004004	104423				TWAT16			:WAIT FOR INTERRUPT	
3118	004006	104002				ERROR	2		:TO SLOW/NOT COMPLETE ERROR	
3119										
3120	004010	104420				TGETRK			:GET RK REGS	
3121	004012	032737	100000	001540		BIT	#CERR,T,CS1		:ERROR?	
3122	004020	001031				BNE	10\$:YES - SKIP	
3123	004022	032737	000200	001552		BIT	#DRDY,T,DS		:ELSE TEST IF DRIVE READY	
3124	004030	001404				BEQ	35\$:NO - SKIP	
3125	004032	032737	004000	001552		BIT	#WRL,T,DS		:ELSE TEST IF WRITE LOCKED	
3126	004040	001403				BEQ	36\$:NO - SKIP	
3127	004042	050137	001666		35\$:	BIS	R1,DRVDRP		:SET THIS BIT AS DROPPED DRIVE	
3128	004046	000406				BR	9\$			
3129	004050				36\$:					
3130	004050	050113				BIS	R1,(R3)		:SET BIT - DRIVE PRESET IN MAP	
3131	004052	006300				ASL	R0			
3132	004054	013760	001720	001722		MOV	DTYPE,TYPTBL(R0)		:SET TABLE WITH DRIVE TYPE INFO	
3133	004062	006200				ASR	R0			
3134										
3135	004064	005200			9\$:	INC	R0		:BUMPS TO NEXT DRIVE	
3136	004066	006301				ASL	R1		:SHIFT DRIVE SELECTOR TO NEXT DRIVE.	
3137	004070	032701	000400			BIT	#BIT8,R1		:WAS LAST DRIVE DONE?	
3138	004074	001022				BNE	11\$:BR IF YES	
3139	004076	005037	001720			CLR	DTYPE		:ELSE INIT FOR RK06 TO TEST NEXT DRV	
3140	004102	000730				BR	8\$			
3141										
3142	004104	032737	010000	001550	10\$:	BIT	#NED,T,CS2		:WAS CERR DUE TO NED?	
3143	004112	001364				BNE	9\$:YES - BUMP TO NEXT DRIVE	
3144	004114	032737	000040	001554		BIT	#DTYE,T,ER		:ELSE SEE IF DRIVE TYPE ERR	
3145	004122	001404				BEQ	24\$:BR IF NO	
3146										
3147	004124	012737	002000	001720		MOV	#CDT,DTYPE		:ELSE SET UP FOR RK07	
3148	004132	000714				BR	8\$:AND TRY AGAIN	
3149										
3150	004134	104421			24\$:	TCHKOP			:ELSE REPORT THE ERRORS	
3151	004136	104004				ERROR	4 ;OR5,6,7			
3152	004140	000000			101\$:	.WORD	0			
3153	004142	032737	004000	001664	11\$:	BIT	#DRVRPT,OPTFLG		:TEST IF SHOULD REPORT	
3154	004150	001037				BNE	16\$:NO - SKIP	
3155										
3156	004152	005713				TST	(R3)		:ANY DRIVE AVAILABLE?	
3157	004154	001004				BNE	12\$:BR IF NOT ZERO	
3158	004156	104401	051317			TYPE	,OPR008		:ELSE REPORT NO DRIVES AVAILABLE	
3159	004162	000137	043724			JMP	CTRHLT		:GO TO CONTROLLED HALT	
3160										
3161	004166	012701	000200		12\$:	MOV	#BIT7,R1		:SET DRIVE SELECTOR FOR DRIVE 7	

3162	004172	012700	000007			MOV	#7,R0	:SET DRIVE NUMBER TO 7
3163	004176	104401	051402			TYPE	.OPR009	:TYPE PREFIX TO DRIVE TEST MESSAGE
3164								
3165	004202	030113			13\$:	BIT	R1,(R3)	:TEST IF THIS DRIVE TO BE TESTED
3166	004204	001007				BNE	15\$:YES - SKIP
3167								
3168	004206	005300			14\$:	DEC	R0	:ELSE DECREMENT DRIVE NUMBER
3169	004210	006201				ASR	R1	:SHIFT BIT SELECTOR TO NEXT DRIVE DOWN
3170	004212	001373				BNE	13\$:IF NOT SHIFTED OUT - LOOP
3171	004214	052737	004000	001664		BIS	#DRV RPT,OPTFLG	:SET DRIVE #'S REPORTED FLAG
3172	004222	000412				BR	16\$:ELSE GO TO STATUS VALID TEST
3173								
3174	004224	010037	004140		15\$:	MOV	R0,101\$:PUT DRIVE NUMBER IN TYPE LOCATION
3175	004230	052737	000060	004140		BIS	#BIT4!BIT5,101\$:MAKE IT ASCIZ
3176	004236	104401				TYPE		:TYPE IT
3177	004240	004140				101\$		
3178	004242	104401	051053			TYPE	.SPACE2	:TYPE SOME SPACES
3179	004246	000757				BR	14\$:LOOP
3180								
3181	004250	005000			16\$:	CLR	R0	:CLEAR DRIVE NUMBER COUNTER
3182	004252	012701	000001			MOV	#1,R1	:SET DRIVE SELECTOR TO DRIVE 0
3183	004256	012737	177777	001240		MOV	#-1,\$TMP7	:SET \$TMP7 NEGATIVE
3184	004264	012737	177777	001630		MOV	#-1,DRVBIT	:SET DRIVE SELECT BIT NEGATIVE
3185	004272	012737	004400	001264		MOV	#18\$,\$ESCAPE	:SET ESCAPE IN CASE OF ERR
3186	004300	030137	001666			BIT	R1,DRVDRP	:HAS DRIVE 0 BEEN DROPPED?
3187	004304	001035				BNE	18\$:YES - SKIP TO DRIVE INC.
3188								
3189	004306	104416			17\$:	TSSINIT		:CLEAR SUBSYSTEM
3190	004310	104003				ERROR	3	:ERROR FOR BAD CLEAR
3191								
3192	004312	010037	001610			MOV	R0,L.CS2	:SET DRIVE SELECT
3193	004316	010037	001626			MOV	R0,DRVNUM	:SET DRIVE NUMBER
3194	004322	004737	036134			JSR	PC,GTYP0	:GET DRV TYPE
3195	004326	012737	000101	001600		MOV	#SELDRV,L.CS1	:SET FOR DRIVE SELECT
3196								
3197	004334	104417				TLOADRK		:LOAD RK REGS
3198	004336	104423				TWAT16		:WAIT FOR INTERRUPT
3199	004340	104002				ERROR	2	:ERROR TO SLOW/NOT COMPLETE
3200	004342	030113				BIT	R1,(R3)	:WAS THAT DRIVE AVAILABLE
3201	004344	001026				BNE	19\$:YES - SKIP
3202								
3203	004346	104422				TCHKWE		:CHECK THAT ERROR OCCURRED
3204	004350	000000				.WORD	0	:NONE OF GROUP 1
3205	004352	000000				.WORD	0	:NONE OF GROUP 2
3206	004354	000001				.WORD	1	:NED IN GROUP 3
3207	004356	104004				ERROR	4 ;OR5,6,7	:ERROR IF NO ERROR OR WRONG ERROR
3208								
3209	004360	032737	100000	001552		BIT	#SVAL,T.DS	:DID STATUS VALID RESET?
3210	004366	001404				BEQ	18\$:YES - SKIP
3211	004370	012737	055527	001400		MOV	#EM47,EM3N	:ELSE MESSAGE (SVAL NOT RESET WITH NED)
3212	004376	104003				ERROR	3	
3213	004400	005200			18\$:	INC	R0	:BUMP TO NEXT DRIVE
3214	004402	006301				ASL	R1	:SHIFT DRIVE SELECT BIT
3215	004404	032701	000400			BIT	#BIT8,R1	:ALL DRIVES CHECKED
3216	004410	001063				BNE	21\$:YES - EXIT
3217	004412	030137	001666			BIT	R1,DRVDRP	:TEST IF DRIVE HAS BEEN DROPPED

```
3218 004416 001370          BNE 18$          ;YES - GET NEXT DRIVE
3219 004420 000732          BR 17$          ;ELSE CHECK THIS DRIVE
3220 004422 104421          19$: TCHKOP      ;CHECK NO ERRORS SET
3221 004424 104004          ERROR 4 ;OR5,6,7 ;REPORT ALL ERRORS
3222
3223 004426 032737 100000 001552 BIT #SVAL,T.DS   ;CHECK SVAL SET
3224 004434 001004          BNE 20$          ;YES - SKIP
3225 004436 012737 055610 001400 MOV #EM48,EM3N  ;MESSAGE (NO SVAL FROM EXISTANT DR)
3226 004444 104003          ERROR 3
3227
3228 004446 012737 000103 001600 20$: MOV #PACK,L.CS1 ;ELSE SET TO DO PACK ACK
3229 004454 104417          TLOADRK        ;LOAD RK
3230
3231 004456 104423          TWAT16         ;WAIT FOR INTERRUPT
3232 004460 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
3233
3234 004462 104421          TCHKOP        ;CHECK FOR ANY ERRORS
3235 004464 104004          ERROR 4 ;OR5,6,7 ;YES - REPORT & ABORT TEST
3236
3237 004466 032737 000100 001552 BIT #VV,T.DS    ;DID VV SET
3238 004474 001005          BNE 22$          ;YES - SKIP
3239 004476 012737 055472 001400 MOV #EM46,EM3N ;MESSAGE (VV DID NOT SET/)
3240 004504 104003          ERROR 3
3241 004506 000734          BR 18$
3242
3243 004510 032737 040000 001552 22$: BIT #DSC,T.DS ;TEST IF DSC RESET
3244 004516 001410          BEQ 23$          ;YES - SKIP
3245 004520 012737 056604 001470 MOV #EMDSC,EM12N
3246 004526 012737 057044 061112 MOV #EMSCLR,DF011A
3247 004534 104003          ERROR 3        ;'DSC NOT RESET RESULT OF SUBSYS CLEAR''
3248 004536 000720          BR 18$
3249
3250 004540 005737 001630          23$: TST DRVBIT   ;TEST IF DRVBIT IS NEGATIVE
3251 004544 100315          BPL 18$          ;NO - SKIP
3252 004546 010137 001630          MOV R1,DRVBIT   ;STORE DRIVE SELECT BIT
3253 004552 010037 001240          MOV R0,$TMP7    ;STORE DRIVE NUMBER TO BE TESTED
3254 004556 000710          BR 18$
3255
3256 004560          21$:
3257 004560 005037 001264          CLR $ESCAPE     ;REV 006
3258 004564 013737 001240 001626 MOV $TMP7,DRVNUM ;LOAD LOWEST # DRIVE PRESENT INTO DRVNUM
3259 004572 013701 001626          MOV DRVNUM,R1
3260 004576 004737 036150          JSR PC,GTYP1    ;GET DRV TYP
3261
3262 004602 023727 001624 000002 CMP SRTFLG,#2    ;TEST IF CLOCK ADJUST START
3263 004610 001002          BNE 25$          ;NO - SKIP
3264 004612 000137 043614          JMP ADJCLK      ;GO TO ADJUST CLOCK ROUTINE
3265
3266 004616          25$:
3267
3268 .....
3269 ;*TEST 4          RELEASE ALL DRIVES
3270 ;*
3271 ;*          RELEASE ALL DRIVES. MAKE SURE NO ERROR
3272 ;*          SETS AND STATUS VALID IS RESET.
3273 ;*
```

```

3274
3275 004616 000004
3276 004620 012737 000062 001262
3277 004626 104416
3278 004630 104003
3279
3280 004632 013737 001626 001610
3281 004640 012737 000101 001600
3282
3283 004646 104417
3284 004650 104423
3285 004652 104002
3286
3287 004654 104421
3288 004656 104004
3289
3290 004660 012737 000010 001610
3291
3292 004666 104417
3293 004670 104423
3294 004672 104002
3295
3296 004674 104421
3297 004676 104004
3298 004700 032737 100000 001552
3299 004706 001404
3300 004710 012737 055667 001400
3301 004716 104003
3302 004720
3303
3304 004720 004737 036034
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315 004724 000004
3316 004726 012737 000062 001262
3317 004734 104416
3318 004736 104003
3319 004740 012701 000001
3320 004744 013737 001626 001610
3321 004752 012737 000101 001600
3322 004760 005037 001616
3323 004764 104417
3324 004766 104423
3325 004770 104002
3326 004772 104421
3327 004774 104004
3328
3329 004776 032737 100000 001552

```

```

*****
TST4: SCOPE
MOV #50.,$TIMES ;;DO 50. ITERATIONS
TSSINIT ;;INITIALIZE SUBSYSTEM
ERROR 3 ;BAD INIT
MOV DRVNUM,L.CS2 ;SET DRIVE NUMBER
MOV #SELDRV,L.CS1 ;SET DRIVE SELECT
TLOADRK ;LOAD RK REGS
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK FOR ANY ERRORS
ERROR 4 ;OR5,6,7 ;REPORT ANY ERRORS
MOV #RLS,L.CS2 ;SET DRIVE RELEASE,STILL SET FOR SELECT
TLOADRK ;LOAD RK REGS
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK FOR ANY ERRORS
ERROR 4 ;OR 5, 6, OR 7 ;REPORT ALL ERRORS
BIT #SVAL,T.DS ;DID SVAL RESET?
BEQ 1$ ;YES - SKIP
MOV #EM49,EM3N ;MESSAGE (SVAL NOT RESET W/RELEASE)
ERROR 3
1$:
TSTLUP: JSR PC,LDCON ;LOAD RK06-RK07 CONSTANTS
*****
*TEST 5 NON-STANDARD MESSAGES AND SVAL
*
* PICK ONE OF THE AVAILABLE DRIVES AND GET
* NON-STANDARD MESSAGES. MAKE SURE NO
* ERROR OCCURS AND STATUS VALID DOES NOT SET
* AND THAT NON-STANDARD MESSAGES CAUSE STATUS
* VALID TO RESET.
*****
TST5: SCOPE
MOV #50.,$TIMES ;;DO 50. ITERATIONS
TSSINIT ;;CLEAR SUBSYSTEM
ERROR 3 ;BAD CLEAR MESSAGE
MOV #1,R1 ;PRESET R1 FOR MESSAGE PAIR 1
MOV DRVNUM,L.CS2 ;LOAD DRV NUMBER
MOV #SELDRV,L.CS1 ;LOAD SELECT COMMAND
1$: CLR L.MR1 ;LOAD FOR STANDARD STATUS
TLOADRK ;LOAD RK
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK OPERATION
ERROR 4 ;5,6 OR 7 ;REPORT ALL ERRORS
BIT #SVAL,T.DS ;TEST STATUS VALID SET

```



```
3330 005004 001007 BNE 2$ ;YES-SKIP
3331
3332 005006 012737 056444 001460 MOV #EMSVAL,EM11N
3333 005014 012737 050250 061112 MOV #EMSELD,DF011A
3334 005022 104011 ERROR 11 ;"SVAL NOT SET RESULT OF DRIVE SELECT"
3335
3336 005024 010137 001616 2$: MOV R1,L.MR1 ;LOAD MESSAGE PAIR SELECT
3337
3338 005030 104417 TLOADRK ;LOAD RK
3339 005032 104423 TWAT16 ;WAIT FOR INTERRUPT
3340 005034 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3341
3342 005036 104421 TCHKOP ;CHECK OPERATION
3343 005040 104004 ERROR 4 ;5,6, OR 7 ;REPORT ALL ERRORS
3344
3345 005042 032737 100000 001552 BIT #SVAL,T.DS ;TEST STATUS VALID RESET
3346 005050 001407 BEQ 3$ ;YES-SKIP
3347
3348 005052 012737 056444 001470 MOV #EMSVAL,EM12N
3349 005060 012737 056461 061112 MOV #EMNZPR,DF011A
3350 005066 104012 ERROR 12 ;"SVAL NOT RESET RESULT OF SEL W/ NON-O PAIR"
3351
3352 005070 022701 000003 3$: CMP #3,R1 ;WAS PAIR 3 SELECTED?
3353 005074 001402 BEQ 4$ ;YES-SKIP
3354 005076 005201 INC R1 ;BUMP TO NEXT PAIR
3355 005100 000727 BR 1$ ;SKIP TO DO IT.
3356 005102
3357
3358 :*****
3359 :*TEST 6 WRITING CS2 AND STATUS VALID
3360 :*
3361 :* SELECT AN AVAILABLE DRIVE. MAKE SURE STATUS
3362 :* VALID IS SET. WRITE COMMAND AND STATUS REGISTER 2.
3363 :* MAKE SURE STATUS VALID RESETS.
3364 :*****
3365 005102 000004 TST6: SCOPE
3366 005104 012737 000062 001262 MOV #50, $TIMES ;:DO 50. ITERATIONS
3367 005112 104416 TSSINIT ;CLEAR SUBSYSTEM
3368 005114 104003 ERROR 3 ;BAD INIT ERROR
3369
3370 005116 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3371 005124 012737 000101 001600 MOV #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3372
3373 005132 104417 TLOADRK ;LOAD RK
3374 005134 104423 TWAT16 ;WAIT FOR INTERRUPT
3375 005136 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3376
3377 005140 104421 TCHKOP ;CHECK OPERATION
3378 005142 104004 ERROR 4 ;5,6, OR 7 ;REPORT ALL ERRORS
3379
3380 005144 032737 100000 001552 BIT #SVAL,T.DS ;TEST STATUS VALID SET
3381 005152 001007 BNE 1$ ;YES-SKIP
3382
3383 005154 012737 056444 001460 MOV #EMSVAL,EM11N
3384 005162 012737 050250 061112 MOV #EMSELD,DF011A
3385 005170 104011 ERROR 11 ;"SVAL NOT SET RESULT OF DRV SELECT"
```

```
3386
3387 005172 013762 001626 000010 1$: MOV DRVNUM,RKCS2(R2) ;WRITE CS2 TO RESET SVAL
3388
3389 005200 104420 TGETRK ;GET RK REGS.
3390
3391 005202 032737 100000 001552 BIT #SVAL,T.DS ;TEST SVAL RESET
3392 005210 001407 BEQ 2$ ;YES-SKIP
3393
3394 005212 012737 056444 001470 MOV #EMSVAL,EM12N
3395 005220 012737 056522 061112 MOV #EMWCS2,DF011A
3396 005226 104012 ERROR 12 ;"SVAL NOT RESET BY WRITING CS2"
3397 005230 2$:
3398
3399 .SBTTL **CONTROLLER ERROR TESTS
3400
3401 :*****
3402 :*TEST 7 DRIVE TYPE ERROR
3403 :*
3404 :* CREATE A DRIVE TYPE ERROR. MAKE SURE DRIVE
3405 :* TYPE ERROR SETS AND STATUS VALID SETS.
3406 :*
3407 :*****
3408 005230 000004 TST7: SCOPE
3409 005232 012737 000062 001262 MOV #50.,$TIMES ;;DO 50. ITERATIONS
3410 005240 104416 TSSINIT ;CLEAR SUBSYSTEM
3411 005242 104003 ERROR 3 ;BAD INIT ERROR
3412
3413 005244 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3414 005252 012737 000101 001600 MOV #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3415 005260 053737 036130 001600 BIS HOLD2,L.CS1 ;LOAD DRIVE TYPE
3416
3417 005266 104417 TLOADRK ;LOAD RK
3418 005270 104423 TWAT16 ;WAIT FOR INTERRUPT
3419 005272 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3420
3421 005274 104422 TCHKWE ;CHECK OPERATION WITH EXPECTED ERROR
3422 005276 000040 .WORD 000040 ;DRIVE TYPE ERROR
3423 005300 000000 .WORD 0
3424 005302 000000 .WORD 0
3425 005304 104004 ERROR 4 ; OR 5,6,7 ;REPORT ANY DIFFERENCES (NO ERRORS,
3426 ;ADDITIONAL ERRORS, DIFFERENT ERRORS)
3427 005306 032737 100000 001552 BIT #SVAL, T.DS ;TEST IF SVAL SET
3428 005314 001007 BNE 1$ ;YES-SKIP
3429
3430 005316 012737 056444 001460 MOV #EMSVAL,EM11N
3431 005324 012737 055140 061112 MOV #EMDTPE,DF011A
3432 005332 104011 ERROR 11 ;"SVAL NOT SET RESULT OF DRV TYPE ERR"
3433 005334 1$:
3434 :*****
3435 :*TEST 10 STATUS VALID AND PARITY ERROR
3436 :*
3437 :* ISSUE A SELECT TO AN AVAILIABLE DRIVE WITH BAD PARITY.
3438 :* MAKE SURE SPAR, CONTROLLER ERROR, ATTENTION,
3439 :* DRIVE STATUS CHANGES, DRPAR, DRIVE INTERRUPT,
3440 :* AND STATUS VALID SET, ISSUE A CONTROLLER
3441 :* CLEAR. MAKE SURE DRIVE INTERRUPT AND ATTENTION
```

```
3442 :* ARE STILL SET. SELECT DRIVE AGAIN WITH GOOD
3443 :* PARITY. MAKE SURE ATTENTION, DRIVE STATUS
3444 :* CHANGE, DRPAR, CONTROLLER ERROR, DRIVE INTERRUPT,
3445 :* AND STATUS VALID ARE SET AND SPAR IS RESET.
3446 :* ISSUE A CONTROLLER CLEAR. GET NON-STANDARD MESSAGES
3447 :* AND MAKE SURE ONLY DRIVE INTERRUPT AND ATTENTION
3448 :* ARE SET. CLEAR ATTENTION WITH DRIVE CLEAR. REPEAT
3449 :* FOR ALL AVAILIABLE DRIVES.
3450 :*
3451 :*****
3452 TST10: SCOPE
3453 005334 000004 MOV #50.,$TIMES ;:DO 50. ITERATIONS
3454 005336 012737 000062 001262 TSSINIT ;:CLEAR SUBSYSTEM
3455 005344 104416 ERROR 3 ;:BAD INIT ERROR
3456 005346 104003
3457 005350 013737 001626 001610 MOV DRVNUM,L.CS2 ;:LOAD DRIVE NUMBER
3458 005356 012737 000101 001600 MOV #SELDRV,L.CS1 ;:LOAD DRIVE SELECT
3459 005364 012737 000020 001616 MOV #PAT,L.MR1 ;:LOAD EVEN PARITY BIT
3460
3461 005372 104417 TLOADRK ;:LOAD RK REGS-SELECT W/EVEN PARITY
3462 005374 104423 TWAT16 ;:WAIT FOR INTERRUPT
3463 005376 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
3464
3465 005400 104422 TCHKWE ;:CHECK OPERATION FOR EXPECTED ERROR
3466 005402 000011 DRPARERR!SPARERR ;:DRIVE SELECTED DRIVE BUS PARITY ERROR
3467 005404 000000 .WORD 0 ;:CONTROLLER DETECTED DRIVE BUS PARITY ERROR
3468 005406 000000 .WORD 0
3469 005410 104004 ERROR 4 ; OR 5,6,7 ;:REPORT ANY DIFFERENCES
3470
```

```
3471 005412 012700 000400      MOV      #BIT8,R0      ;ROUTINE TO DETERMINE WHICH BIT
3472
3473 005416 013701 001626      MOV      DRVNUM,R1    ;SHOULD BE SET IN ASOF TO INDICATE
3474 005422 001403              BEQ      3$           ;DRIVE ATTENTION. R0 WILL HAVE THE
3475 005424 006300              2$:    ASL      R0      ;BIT THAT SHOULD BE SET FOR THE DRIVE
3476 005426 005301              DEC      R1          ;IN USE
3477 005430 001375              BNE      2$
3478
3479 005432 030037 001556              3$:    BIT      R0,T.ASOF ;TEST ATTENTION SET
3480 005436 001007              BNE      4$           ;YES-SKIP
3481 005440 012737 056630 001460      MOV      #EMDA,EM11N
3482 005446 012737 055055 061112      MOV      #EMDPAR,DF011A
3483 005454 104011              ERROR   11          ;"DRV ATT NOT SET RESULT OF DRV PARITY ERR"
3484 005456 032737 040000 001540      4$:    BIT      #DI,T.CS1 ;TEST DRIVE INTERRUPT SET
3485 005464 001007              BNE      5$           ;YES-SKIP
3486 005466 012737 056564 001460      MOV      #EMDI,EM11N
3487 005474 012737 055055 061112      MOV      #EMDPAR,DF011A
3488 005502 104011              ERROR   11          ;"DRV INT NOT SET RESULT OF DRV PARITY ERR"
3489
3490 005504 032737 040000 001552      5$:    BIT      #DSC,T.DS  ;TEST DRIVE STATUS CHANGE SET
3491 005512 001007              BNE      6$           ;YES-SKIP
3492 005514 012737 056604 001460      MOV      #EMDSC,EM11N
3493 005522 012737 055055 061112      MOV      #EMDPAR,DF011A
3494 005530 104011              ERROR   11          ;"DSC NOT SET RESULT OF DRV PARITY ERR"
3495
3496 005532 032737 100000 001552      6$:    BIT      #SVAL,T.DS  ;TEST STATUS VALID SET
3497 005540 001007              BNE      7$           ;YES-SKIP
3498 005542 012737 056444 001460      MOV      #EMSVAL,EM11N
3499 005550 012737 055055 061112      MOV      #EMDPAR,DF011A
3500
3501 005556 104011              ERROR   11          ;"SVAL NOT SET RESULT OF DRV PAR ERR"
3502
3503 005560 005037 001616              7$:    CLR      L.MR1    ;CLEAR PAT IN MR1
3504
3505 005564 052737 100000 001600      BIS      #CCLR,L.CS1  ;CLEAR CONTROLLER
3506 005572 104417              TLOADRK ;LOAD RK REGS TO DO CLEAR
3507
3508
3509 005574 104421              TCHKOP
3510 005576 104004              ERROR   4 ; OR 5,6,7 ;CHECK NO ERRORS SET
;REPORT ALL ERRORS STILL SET
3511
3512 005600 030037 001556              BIT      R0,T.ASOF    ;TEST ATTENTION STILL SET
3513 005604 001007              BNE      8$           ;YES-SKIP
3514 005606 012737 056630 001510      MOV      #EMDA,EM14N
3515 005614 012737 056650 061112      MOV      #EMCCLR,DF011A
3516 005622 104014              ERROR   14          ;"DRV ATT RESET RESULT OF CONT CLR"
3517
3518 005624 032737 040000 001540      8$:    BIT      #DI,T.CS1 ;TEST DRIVE INTERRUPT STILL SET
3519 005632 001007              BNE      9$           ;YES-SKIP
3520 005634 012737 056564 001510      MOV      #EMDI,EM14N
3521 005642 012737 056650 061112      MOV      #EMCCLR,DF011A
3522 005650 104014              ERROR   14          ;"DRV INT RESET RESULT OF CONTROLLER CLR"
3523
3524 005652 012737 000101 001600      9$:    MOV      #SELDIV,L.CS1 ;LOAD SELECT DRIVE
3525
3526 005660 104417              TLOADRK ;LOAD RK REGS
```

```

3527 005662 104423      TWAT16      ;WAIT FOR INTERRUPT
3528 005664 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
3529
3530 005666 104422      TCHKWE     ;CHECK THAT ERRORS ARE SET
3531 005670 000010      .WORD      000010 ;DPAR SET (NO SPAR 1, REPORT
3532 005672 000000      .WORD      0      ;ANY DISCREPENCIES
3533 005674 000000      .WORD      0
3534 005676 104004      ERROR      4 ; OR 5,6,7
3535 005700 030037 001556      BIT        RO,T.ASOF ;TEST ATTENTION STILL SET.
3536 005704 001007      BNE        10$      ;YES-SKIP
3537 005706 012737 056630 001460      MOV        #EMDA,EM11N
3538 005714 012737 055055 061112      MOV        #EMDPAR,DF011A
3539 009722 104011      ERROR      11      ;'DRV ATT NOT SET RESULT OF DRV PARITY ERR'
3540
3541 005724 032737 040000 001540 10$:      BIT        #DI,T.CS1 ;TEST DRIVE INTERRUPT STILL SET
3542 005732 001007      BNE        11$      ;YES-SKIP
3543 005734 012737 056564 001460      MOV        #EMDI,EM11N
3544 005742 012737 055055 061112      MOV        #EMDPAR,DF011A
3545 005750 104011      ERROR      11      ;'DRV INT NOT SET RESULT OF DRV PARITY ERR'
3546
3547 005752 032737 040000 001552 11$:      BIT        #DSC,T.DS  ;TEST STATUS CHANGE STILL SET
3548 005760 001007      BNE        12$      ;YES-SKIP
3549 005762 012737 056604 001460      MOV        #EMDSC,EM11N
3550 005770 012737 055055 061112      MOV        #EMDPAR,DF011A
3551 005776 104011      ERROR      11      ;'DSC NOT SET RESULT OF DRV PARITY ERR'
3552
3553 006000 032737 100000 001552 12$:      BIT        #SVAL,T.DS ;TEST STATUS VALID STILL SET
3554 006006 001007      BNE        13$      ;YES-SKIP
3555 006010 012737 056444 001460      MOV        #EMSVL,EM11N
3556 006016 012737 055055 061112      MOV        #EMDPAR,DF011A
3557 006024 104011      ERROR      11      ;'SVAL NOT SET RESULT OF DRV PARITY ERR'
3558
3559 006026 052737 100000 001600 13$:      BIS        #CCLR,L.CS1 ;CLEAR CONTROLLER
3560
3561 006034 104417      TLOADRK    ;LOAD RK REGS (DO CLEAR)
3562
3563
3564 006036 012701 000001      MOV        #1,R1      ;SET COUNTER TO SELECT STATUS PAIR
3565 006042 012737 000101 001600      MOV        #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3566
3567 006050 010137 001616      14$:      MOV        R1,L.MR1   ;LOAD STATUS PAIR SELECTION
3568 006054 104417      TLOADRK    ;LOAD RK REGS
3569 006056 104423      TWAT16     ;WAIT FOR INTERRUPT
3570 006060 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
3571
3572 006062 104421      TCHKOP     ;CHECK IF ANY ERRORS SET
3573 006064 104004      ERROR      4 ; OR 5,6,7 ;REPORT ALL ERRORS SET.
3574
3575 006066 030037 001556      BIT        RO,T.ASOF ;TEST ATTENTION STILL SET
3576 006072 001007      BNE        15$      ;YES-SKIP
3577 006074 012737 056630 001410      MOV        #EMDA,EM4N
3578 006102 012737 056461 061112      MOV        #EMNZPR,DF011A
3579 006110 104014      ERROR      14      ;'ATTENTION RESET RESULT OF NON-0 PAIR SEL'
3580
3581 006112 032737 040000 001540 15$:      BIT        #DI,T.CS1
3582 006120 001007      BNE        16$

```

```

3583 006122 012737 056564 001510      MOV    #EMDI,EM14N
3584 006130 012737 056461 061112      MOV    #EMNZPR,DF011A
3585 006136 104014                ERROR  14      ;'DRV INT RESET RESULT OF NON-0 PAIR SELECT''
3586
3587 006140 005201                16$:  INC    R1      ;BUMP PAIR SELECT
3588 006142 022701 000004          CMP    #4,R1   ;ALL PAIRS DONE?
3589 006146 001340                BNE   14$     ;NO-LOOP
3590
3591 006150 005037 001616          CLR   L.MR1   ;CLEAR MR1
3592
3593 006154 012737 000105 001600      MOV    #CLEAR,L.CS1 ;LOAD DRIVE CLEAR
3594
3595 006162 104417                TLOADRK      ;DO DRIVE CLEAR
3596 006164 104423                TWAT16      ;WAIT FOR INTERRUPT
3597 006166 104002                ERROR  2      ;TO SLOW/NOT COMPLETE ERROR
3598
3599 006170 104421                TCHKOP      ;CHECK FOR ANY ERRORS
3600 006172 104004                ERROR  4 ; OR 5,6,7 ;REPORT ALL ERRORS
3601
3602 006174 012701 000020                17$:  MOV    #20,R1   ;SET COUNT FOR SHORT WAIT
3603 006200 005301                DEC    R1     ;TO ALLOW CONTROLLER TIME TO POLL
3604 006202 001376                BNE   17$     ;DRIVES
3605
3606 006204 104420                TGETRK      ;GET RK REGS
3607 006206 030037 001556          BIT    R0,T.ASOF ;TEST ATTENTION RESET
3608 006212 001407                BEQ   18$     ;YES-SKIP
3609 006214 012737 056630 001470      MOV    #EMDA,EM12N
3610 006222 012737 050276 061112      MOV    #EMDCLR,DF011A
3611 006230 104012                ERROR  12     ;'ATTENTION NOT RESET RESULT OF DRV CLEAR
3612
3613 006232 032737 040000 001540      18$:  BIT    #DI,T.CS1 ;TEST DRIVE INTERRUPT RESET
3614 006240 001407                BEQ   19$     ;YES-SKIP
3615 006242 012737 056564 001470      MOV    #EMDI,EM12N
3616 006250 012737 050276 061112      MOV    #EMDCLR,DF011A
3617 006256 104012                ERROR  12     ;'DRV INT NOT RESET RESULT OF DRIVE CLR''
3618
3619 006260                19$:
3620                ;*****
3621                ;TEST 11      UNIT FIELD ERROR ON RELEASE
3622                ;*
3623                ;*
3624                ;*   ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE
3625                ;*   DRIVE. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE
3626                ;*   A RELEASE COMMAND. CLOCK THROUGH PHASE ADDRESS 2.
3627                ;*   TURN OFF DIAGNOSTIC MODE. MAKE SURE UNIT FIELD
3628                ;*   ERROR SETS.
3629                ;*****
3630                TST11: SCOPE
3631 006260 000004                MOV    #50,,$TIMES ;:DO 50. ITERATIONS
3632 006262 012737 000062 001262      TSSINIT      ;CLEAR SUBSYSTEM
3633 006270 104416                ERROR  2      ;BAD INIT ERROR
3634
3635 006274 013737 001626 001610      MOV    DRVNUM,L.CS2 ;SELECT A DRIVE
3636 006302 012737 000101 001600      MOV    #SELDRV,L.CS1 ;DO DRIVE SELECT
3637
3638 006310 104417                TLOADRK      ;LOAD RK
  
```

```

3639 006312 104423          TWAT16          ;WAIT FOR INTERRUPT
3640 006314 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
3641
3642 006316 104421          TCHKOP          ;CHECK FOR ANY ERRORS
3643 006320 104004          ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS.
3644
3645 006322 052737 000010 001610  BIS #RLS,L.CS2   ;LOAD RELEASE
3646 006330 012737 000040 001616  MOV #DMD,L.MR1   ;SET DIAGNOSTIC MODE
3647
3648 006336 104417          TLOADRK         ;LOAD RK
3649
3650 006340 004437 036316  JSR R4,MCLOCK   ;CALL MAINT CLOCK
3651 006344      023          .BYTE ^D19      ;NUMBER OF PHASES
3652 006345      002          .BYTE 2         ;NUMBER OF CLOCK XISTIONS
3653
3654 006346 042762 000040 000026  BIC #DMD,RKMR1(R2) ;CLEAR DIAG MODE
3655
3656 006354 104423          TWAT16          ;WAIT FOR INTERRUPT
3657 006356 104002          ERROR 2         ;TO SLOW/NOT COMPLETED
3658
3659 006360 104422          TCHKWE          ;CHECK OPERATION WITH ERROR
3660 006362 000000          .WORD 0
3661 006364 000000          .WORD 0
3662 006366 000004          .WORD UFERR     ;UNIT FIELD ERROR
3663 006370 104004          ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
3664
3665 006372 104416          TSSINIT         ;CLEAR SUBSYSTEM TO INSURE UFE RESETS
3666 006374 104002          ERROR 2
3667
3668 *****
3669 *TEST 12          UNIT FIELD ERROR ON SELECT
3670 *
3671 *      ISSUE A SUBSYSTEM CLEAR.  SELECT AN AVAILABLE
3672 *      DRIVE.  PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE
3673 *      A SELECT COMMAND WITH MESSAGE ID = 3 AND DRIVE
3674 *      SELECTED = 0.  CLOCK THROUGH PHASE ADDRESS 6.
3675 *      TURN OFF DIAGNOSTIC MODE.  MAKE SURE UNIT FIELD
3676 *      ERROR SETS.
3677 *****
3678 006376 000004          TST12: SCOPE
3679 006400 012737 000062 001262  MOV #50,STIMES  ;;DO 50. ITERATIONS
3680 006406 104416          TSSINIT         ;CLEAR SUBSYSTEM
3681 006410 104003          ERROR 3        ;BAD INIT ERROR
3682
3683 006412 013737 001626 001610  MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3684 006420 012737 000101 001600  MOV #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3685
3686 006426 104417          TLOADRK         ;LOAD RK
3687 006430 104423          TWAT16          ;WAIT FOR INTERRUPT
3688 006432 104002          ERROR 2         ;TO SLOW/NOT COMPLETE
3689
3690 006434 104421          TCHKOP          ;CHECK FOR ANY ERROR
3691 006436 104004          ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
3692
3693 006440 012737 000043 001616  MOV #DMD!BIT1!BIT0,L.MR1 ;LOAD DIAG MODE & MSG PAIR 3
3694 006446 005037 001610          CLR L.CS2       ;LOAD FOR DRIVE 0

```

```
3695
3696 006452 104417          TLOADRK          ;LOAD RK
3697
3698 006454 004437 036316   JSR      R4,MCLKCK  ;CALL MAINTENANCE CLOCK
3699 006460      026        .BYTE    ^D22      ;THROUGH PHASE 6
3700 006461      002        .BYTE    2              ;PLUS 2 TRANSITIONS
3701
3702 006462 042762 000040 000026 BIC      #DMD,RKMR1(R2) ;CLEAR DIAG MODE
3703
3704 006470 104423          TWAT16          ;WAIT FOR INTERRUPT
3705 006472 104002          ERROR    2          ;TO SLOW/NOT COMPLETED ERROR
3706
3707 006474 013737 036132 006504 MOV      HOLD3,1$    ;EXPECTED ERROR
3708
3709 006502 104422          TCHKWE          ;CHECK OPERATION WITH ERROR
3710 006504 000000          1$: .WORD    0
3711 006506 000000          .WORD    0
3712 006510 000004          .WORD    UFERR      ;UNIT FIELD ERROR SHOULD SET
3713 006512 104004          ERROR    4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
3714
3715          .SBTTL  **ATTENTION HANDLING BY CONTROLLER
3716
3717          ;*****
3718          ;*TEST 13      DOUBLE INTERRUPT
3719          ;*
3720          ;*      ISSUE A SUBSYSTEM CLEAR.  ISSUE A RECALIBRATE.
3721          ;*      MAKE SURE STATUS VALID IS SET.  CHECK THAT SECOND
3722          ;*      INTERRUPT OCCURS.  AFTER SECOND INTERRUPT
3723          ;*      CHECK THAT STATUS VALID IS RESET.  ISSUE SELECT
3724          ;*      AND MAKE SURE STATUS VALID SETS.  CLEAR DRIVE.
3725          ;*      CHECK THAT DRIVE STATUS CHANGE SETS
3726          ;*      (BIT 14 OF DRIVE STATUS
3727          ;*      REGISTER)
3728          ;*
3729          ;*****
3730 006514 000004          TST13: SCOPE
3731 006516 012737 000062 001262 MOV      #50.,$TIMES ;:DO 50. ITERATIONS
3732 006524 104416          TSSINIT        ;:CLEAR SUBSYSTEM
3733 006526 104003          ERROR    3          ;:BAD INIT ERROR
3734
3735 006530 013737 001626 001610 MOV      DRVNUM,L.CS2 ;:LOAD DRIVE NUMBER
3736 006536 012737 000113 001600 MOV      #RECAL,L.CS1 ;:LOAD RECAL
3737
3738 006544 104417          TLOADRK          ;LOAD RK
3739 006546 104423          TWAT16          ;WAIT FOR 1ST INTERRUPT
3740 006550 104002          ERROR    2          ;TO SLOW/NOT COMPLETE
3741 006552 005037 001662 CLR      INTSET      ;:CLEAR INTERRUPT FLAG
3742 006556 104420          TGETRK          ;:GET RK REGS
3743 006560 032737 100000 001552 BIT      #SVAL,T.DS  ;:TEST SVAL SET
3744 006566 001010          BNE      1$        ;:YES-SKIP
3745 006570 012737 056444 001460 MOV      #EMSVL,EM11N
3746 006576 012737 050337 061112 MOV      #EMRCAL,DF011A
3747 006604 104011          ERROR    11        ;:"SVAL NOT SET RESULT OF RECAL"
3748 006606 000463          BR      50$        ;:ABORT TEST
3749
3750 006610 104437          1$: TWAT8S          ;WAIT FOR INTERRUPT
```



```
3751 006612 000401 BR 2$ ;NO INTERRUPT RETURN
3752 006614 000404 BR 3$ ;INTERRUPT RETURN
3753
3754 006616 012737 055737 001370 2$: MOV #EM50,EM2N ;ALTER MESSAGE 'NO 2ND INTERRUPT OR IT WAS LATE'
3755 006624 104002 ERROR 2
3756
3757 006626 104420 3$: TGETRK ;GET RK REGS
3758 006630 032737 100000 001552 BIT #SVAL,T.DS ;TEST SVAC SET NOW
3759 006636 001410 REQ 4$ ;NO-SKIP
3760 006640 012737 056444 001470 MOV #EMSVL,EM12N
3761 006646 012737 056671 061112 MOV #EM2INT,DF011A
3762 006654 104012 ERROR 12 ;'SVAL NOT RESET RESULT OF SECOND TEST'
3763 006656 000437 BR 50$
3764
3765 006660 032737 040000 001552 4$: BIT #DSC,T.DS ;TEST DSC SET BY ATTENTION
3766 006666 001010 BNE 5$ ;YES-SKIP
3767 006670 012737 056604 001460 MOV #EMDSC,EM11N
3768 006676 012737 056671 061112 MOV #EM2INT,DF011A
3769 006704 104011 ERROR 11 ;'DSC NOT SET RESULT OF SECOND INTERRUPT'
3770 006706 000423 BR 50$
3771
3772 006710 012737 000101 001600 5$: MOV #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3773
3774 006716 104417 TLOADRK ;LOAD RK REGS
3775 006720 104423 TWAT16 ;WAIT FOR INTERRUPT
3776 006722 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3777
3778 006724 104421 TCHKOP ;CHECK FOR ANY ERRORS
3779
3780 006726 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
3781
3782 006730 032737 100000 001552 BIT #SVAL,T.DS ;TEST SVAC SET
3783 006736 001007 BNE 50$ ;YES-SKIP
3784 006740 012737 056444 001460 MOV #EMSVL,EM11N
3785 006746 012737 050250 061112 MOV #EMSELD,DF011A
3786 006754 104011 ERROR 11 ;'SVAL NOT SET RESULT OF DRV SEL.
3787 006756 50$:
3788 :*****
3789 :*TEST 14 SINGLE INTERRUPT FROM ATTENTION
3790 :*
3791 :* DO A SEEK TO CYLINDER 0. WAIT FOR INTERRUPT FROM
3792 :* DRIVE ATTENTION. LOWER PRIORITY AGAIN AND MAKE
3793 :* SURE ANOTHER INTERRUPT DOES NOT OCCUR. CLEAR DRIVE.
3794 :*
3795 :*****
3796 006756 000004 TST14: SCOPE
3797 006760 012737 000062 001262 MOV #50,STIMES ;DO 50. ITERATIONS
3798 006766 104416 TSSINIT ;CLEAR SUBSYSTEM
3799 006770 104003 ERROR 3 ;BAD INIT ERROR
3800
3801 006772 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3802 007000 012737 000117 001600 MOV #SEEK,L.CS1 ;LOAD SEEK DCYL LEFT AT 0.
3803
3804 007006 104417 TLOADRK ;LOAD RK REGS
3805 007010 104423 TWAT16 ;WAIT FOR INTERRUPT
3806 007012 104002 ERROR 2 ;TO SLOW/NOT COMPLETED ERROR
```

```
3807
3808
3809 007014 104420          TGETRK          ;GET RK REGS
3810
3811 007016 032737 040000 001540  BIT #DI,T.CS1    ;TEST DI SET
3812 007024 001010          BNE 2$          ;YES-SKIP
3813 007026 012737 056564 001460  MOV #EMDI,EM11N
3814 007034 012737 056712 061112  MOV #EMSKSF,DF011A
3815 007042 104011          ERROR 11        ;"DI NOT SET RESULT OF SEEK TO SELF"
3816 007044 000417          BR 50$
3817
3818
3819 007046 012700 000031 2$: MOV #25.,R0      ;LOAD AND DECREMENT A COUNT TO
3820 007052 005300          3$: DEC R0        ;ZERO. GIVE CONTROLLER A CHANCE TO
3821 007054 001376          BNE 3$          ;INTERRUPT AGAIN. ERROR IF IT DOES.
3822
3823 007056 022737 000001 001662  CMP #1,INTSET    ;CHECK ONLY ONE INTERRUPT OCCURRED
3824 007064 001407          BEQ 50$         ;YES-SKIP
3825 007066 012737 057064 001500  MOV #EMMI,EM13N
3826 007074 012737 056712 061112  MOV #EMSKSF,DF011A
3827 007102 104013          ERROR 13        ;"MULTIPLE INTERRUPTS RESULT OF SEEK TO SELF"
3828
3829 007104 104421          50$: TCHKOP        ;CHECK FOR ANY ERRORS
3830 007106 104004          ERROR 4 ;OR 5,6,7 ;REPORT ALL ERRORS
3831
3832 *****
3833 :*TEST 15 RESET ATTENTIONS WITH UNIBUS INIT
3834
3835 :* DO A SEEK TO CYLINDER 0 ON ALL AVAILIABLE DRIVES.
3836 :* ISSUE A RESET. MAKE SURE ALL ATTENTION RESET.
3837 *****
3838
3839 007110 000004          TST15: SCOPE
3840 007112 012737 000005 001262  MOV #5.,$TIMES  ;;DO 5. ITERATIONS
3841 007120 104416          TSSINIT
3842 007122 104003          ERROR 3         ;BAD INIT ERROR
3843
3844 007124 013737 001626 001610  MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3845 007132 012737 000117 001600  MOV #SEEK,L.CS1  ;LOAD SEEK (TO SELF=0)
3846
3847 007140 104417          TLOADRK        ;LOAD RK REGS
3848 007142 104423          TWAT16        ;WAIT FOR INTERRUPT
3849 007144 104002          ERROR 2        ;TO SLOW/NOT COMPLETE
3850
3851 007146 104420          TGETRK          ;GET RK REGS
3852
3853 007150 032737 040000 001540  BIT #DI,T.CS1    ;TEST DI SET
3854 007156 001010          BNE 1$          ;YES-EXIT
3855 007160 012737 056564 001460  MOV #EMDI,EM11N
3856 007166 012737 056712 061112  MOV #EMSKSF,DF011A
3857 007174 104011          ERROR 11        ;"DI NOT SET RESULT OF SEEK TO SELF"
3858 007176 000450          BR 50$
3859
3859 007200 005037 001662 1$: CLR INTSET      ;CLEAR INTERRUPT COUNTER
3860 007204 000005          RESET          ;DO UNIBUS RESET
3861 007206 004737 045220          JSR PC,$TKINT  ;RESET KEYBOARD INTERRUPT
3862
```

```
3863 007212 005037 001674          CLR      LCLKTK          :CLEAR TICK COUNTER
3864 007216 004737 035314          JSR      PC,MYTIME      :CALL TIMER
3865 007222 022737 000012 001674 5$: CMP      #10.,LCLKTK    :COUNT 10 TICKS (MILLISECONDS)?
3866 007230 001372          BNE      5$             :NO - LOOP
3867
3868 007232 012762 000100 000000  MOV      #IE,RKCS1(R2)  :SET IE FOR ANY STRAY INTERRUPTS
3869 007240 004737 034522          JSR      PC,OPTTSI      :SET UP OPTIONS AGAIN
3870
3871 007244 104423          TWAT16          :WAIT 16 MS FOR AN INTERRUPT
3872 007246 000410          BR       2$           :NONE IS EXPECTED SO RETURN SHOULD BE
3873                          :HERE-BR TO CONTINUE TEST.
3874 007250 012737 056727 001500  MOV      #EMUXIT,EM13N  :INT OCCURRED ON RESET
3875 007256 012737 056771 061112  MOV      #EMRSET,DF011A
3876 007264 104013          ERROR     13          :'UNEXECUTED INTERRUPT RESULT OF RESET'
3877 007266 000414          BR       50$
3878 007270 104420          TGETRK          :GET RK REGS
3879 007272 032737 040000 001540 2$: BIT      #DI,T.CS1    :TEST DI RESET
3880 007300 001407          BEQ      50$          :YES-SKIP
3881 007302 012737 056564 001470  MOV      #EMDI,EM12N
3882 007310 012737 056771 061112  MOV      #EMRSET,DF011A
3883 007316 104012          ERROR     12          :'DI NOT RESET RESULT OF RESET'
3884
3885 007320          50$:
3886
3887
3888 .SBTTL **ILLEGAL DISK ADDRESS ERROR TESTS
3889
3890 :*****
3891 :*TEST 16      ILLEGAL DISK ADDRESS (PART 1)
3892 :*
3893 :*      ISSUE A SEEK TO CYLINDER 0, HEAD 3. MAKE SURE
3894 :*      ILLEGAL ADDRESS ERROR AND SEEK INCOMPLETE SETS.
3895 :*      CLEAR CONTROLLER AND CLEAR DRIVE. REPEAT FOR HEADS 4-7,
3896 :*      CHECKING THAT BOTH IDAE AND SEEK INCOMPLETE SET FOR
3897 :*      HEAD 7 AND IDAE ALONE SETS FOR HEADS 4, 5, AND 6.
3898 :*****
3899 TST16: SCOPE
3900 007322 012737 000012 001262  MOV      #10.,$TIMES  ;;DO 10. ITERATIONS
3901 007330 012701 000003          MOV      #3,R1        :PRESET FOR SELECTING TRACK 3
3902
3903 007334 104416          TSSINIT          :CLEAR SUBSYSTEM
3904 007336 104003          ERROR     3
3905
3906 007340 012737 000113 001600  MOV      #RECAL,L.CS1  :SET UP TO RECAL
3907 007346 013737 001626 001610  MOV      DRVNUM,L.CS2  :LOAD DRIVE
3908
3909 007354 104417          TLOADRK          :LOAD RK REGS
3910
3911 007356 104423          TWAT16          :WAIT FOR 1ST INTERRUPT
3912 007360 104002          ERROR     2          :TO SLOW/NOT COMPLETE ERROR
3913
3914 007362 005037 001662          CLR      INTSET      :CLEAR INTERRUPT FLAG
3915
3916 007366 104437          TWAT8S          :WAIT FOR INTERRUPT
3917 007370 104002          ERROR     2
3918
```

```

3919 007372 012737 007400 001110      MOV    #1$, $LPERR      ;SET LOCAL LOOP ON ERROR
3920
3921 007400 104416      1$:    TSSINIT            ;CLEAR SUBSYSTEM
3922 007402 104003      ERROR  3                ;BAD INIT ERROR
3923
3924 007404 013737 001626 001610      MOV    DRVNUM, L.CS2    ;LOAD DRIVE NUMBER
3925 007412 012737 000117 001600      MOV    #SEEK, L.CS1    ;LOAD SEEK
3926 007420 110137 001607      MOVB   R1, L.DT        ;LOAD TRACK
3927
3928 007424 104417      TLOADRK                ;LOAD RK REGS
3929 007426 104423      TWAT16                 ;WAIT FOR INTERRUPT
3930 007430 104002      ERROR  2                ;TO SLOW/NOT COMPLETE
3931
3932 007432 032701 000001      BIT    #BIT0, R1        ;TEST IF HEAD ADDRESS HAS BIT 0
3933 007436 001403      BEQ    2$                ;NO - SKIP
3934 007440 032701 000002      BIT    #BIT1, R1        ;TEST IF HEAD ADDRESS HAS BOTH 0 AND 1
3935 007444 001007      BNE    3$                ;YES-GO CHECK BOTH IDAE AND SKI SET
3936
3937 007446 104422      2$:    TCHKWE                ;CHECK OPERATION WITH ERROR
3938 007450 002000      IDAERR                 ;ILLEGAL DISK ADDRESS ERROR
3939 007452 000000      0
3940 007454 000000      0
3941 007456 104004      ERROR  4 ; OR 5,6,7    ;REPORT ALL DISCREPANCIES
3942 007460 104415      SCOP1                  ;LOCAL LGOP ON ERROR
3943 007462 000406      BR     4$
3944
3945 007464 104422      3$:    TCHKWE                ;CHECK OPERATION WITH ERROR
3946 007466 002002      !DAERR!SKIERR         ;ILLEGAL DISK ADDRESS ERROR
3947 007470 000000      0                       ;SEEK INCOMPLETE
3948 007472 000000      0
3949 007474 104004      ERROR  4 ;OR 5,6,7    ;REPORT ANY DISCREPANCIES
3950 007476 104415      SCOP1                  ;LOCAL LOOP ON ERROR TO 1$
3951
3952 007500 005201      4$:    INC    R1                ;ELSE BUMP TO NEXT ILLEGAL TRACK
3953 007502 022701 000010      CMP    #8., R1         ;ALL ILLEGAL TRACKS SELECTED?
3954 007506 001334      BNE    1$                ;NO-LOOP
3955
3956 *****
3957 :*TEST 17      ILLEGAL DISK ADDRESS (PART 2)
3958 :*
3959 :*      ISSUE A SEEK TO CYLINDER 1000, HEAD 0. MAKE SURE
3960 :*      ILLEGAL DISK ADDRESS ERROR SETS. CLEAR CONTROLLER AND DRIVE
3961 :*      THIS TEST IS BYPASSED FOR THE RK07 BECAUSE THE RK07
3962 :*      CONTROLLER DOES NOT RECOGNIZE THE ILLEGAL DISK ADDRESS
3963 :*
3964 *****
3964 007510 000004      TEST17: SCOPE
3965 007512 012737 000012 001262      MOV    #10., $TIMES    ;:DO 10. ITERATIONS
3966 007520 104416      TSSINIT                ;:CLEAR SUBSYSTEM
3967 007522 104003      ERROR  3                ;:BAD INIT ERROR
3968
3969 007524 012737 000113 001600      MOV    #RECAL, L.CS1   ;:LOAD RECALIBRATE
3970 007532 013737 001626 001610      MOV    DRVNUM, L.CS2   ;:LOAD DRIVE
3971
3972 007540 104417      TLOADRK                ;:LOAD RK REGS
3973
3974 007542 104423      TWAT16                 ;:WAIT FOR 1ST INTERRUPT
  
```

```
3975 007544 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
3976
3977 007546 005037 001662    CLR  INTSET      ;CLEAR INTERRUPT FLAG
3978
3979 007552 104437          TWAT8S          ;WAIT FOR INTERRUPT
3980 007554 104002          ERROR 2
3981
3982 007556 012737 007564 001110  MOV  #1$, $LPERR ;SET LOOP TO BYPASS RECAL
3983
3984 007564 104416          1$: TSSINIT        ;CLEAR SUBSYSTEM
3985 007566 104003          ERROR 3
3986
3987 007570 013737 001626 001610  MOV  DRVNUM, L.CS2 ;LOAD DRIVE NUMBER
3988 007576 012737 000117 001600  MOV  #SEEK, L.CS1 ;LOAD SFEK
3989 007604 005737 001720          TST  DTYPE      ;SEE IF RK06
3990 007610 001402          BEQ  2$         ;BR IF YES
3991
3992          :        MOV  #1777, L.DCYL ;ELSE LOAD RK07 ILL CYL
3993          :        MOV  #<IDAERR!SKIERR>, 4$ ;EXP RK07 ERR
3994 007612 000137 007644          BR   3$
3995          :        JMP  TST20        ;RK07 CONTROLLER
3996          :        ;DOES NOT RECOGNIZE
3997          :        ;ILLEGAL DISK ADDRESS
3998          :        ;REV 006 THE ABOVE THREE LINES
3999 007616 012737 001000 001614  2$: MOV  #1000, L.DCYL ;LOAD DK06 ILL CYL
4000          :        MOV  #IDAERR, 4$ ;EXP RK06 ERR
4001
4002 007624 104417          3$: TLOADRK      ;LOAD RK REGS
4003 007626 104423          TWAT16         ;WAIT FOR INTERRUPT
4004 007630 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
4005
4006 007632 104422          TCHKWE        ;CHECK OPERATION WITH ERROR
4007 007634 002000          4$: .WORD  IDAERR ;DISK ADDRESS ERROR
4008 007636 000000          .WORD  0
4009 007640 000000          .WORD  0
4010 007642 104004          ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPANCIES
```

.SBTTL **WRITE HEADER TESTS

```
4011
4012
4013
4014
4015 :*****
4016 :*TEST 20 READ BAD SECTOR INFORMATION
4017 :*
4018 :* ISSUE A READ DATA OF 400 WORDS TO CYLINDER 632,
4019 :* TRACK 2 TO GET THE FACTORY DETECTED BAD
4020 :* SECTOR FILE, 26 SECTOR MODE.
4021 :*
4022 :* CYLINDER 1456 IS USED FOR THE RK07.
4023 :*
4024 :* IF AN ERROR OCCURS, READ SECTOR 2, 4, 6, OR 10(8) UNTIL
4025 :* A SUCCESSFUL READ IS DONE. IF NONE READ SUCCESSFULLY
4026 :* REMOVE THIS DRIVE FROM TEST. WHEN A READ IS SUCCESSFUL,
4027 :* TEST THAT THE PACK IS NOT AN ALIGNMENT PADK AND
4028 :* STORE THE ENTRIES FOR LATER USE.
4029 :*
4030 :* REPEAT THIS SERIES OF OPERATIONS FOR FACTORY DETECTED
BAD SECTORS 24 SECTOR MODE, SOFTWARE DETECTED
```

```
4031 :* BAD SECTORS 26 SECTOR MODE, AND SOFTWARE DETECTED BAD
4032 :* SECTORS 24 SECTOR MODE. IF THE NUMBER OF BAD SECTORS FOR
4033 :* 24 OR 26 SECTOR MODE EXCEED 20(10) THE DRIVE IS REMOVED
4034 :* FROM TESTING.
4035 :*
4036 :*
4037 :*****
4038 007644 000004 TST20: SCOPE
4039 007646 012737 000001 001262 MOV #1,$TIMES ;;DO 1 ITERATION
4040 :
4041 : TSSINIT ;CLEAR SUBSYS FROM LAST IDAERR
4042 : ERROR 3 ;BAD INIT ERROR
4043 :
4044 : MOV #RECAL,L.CS1 ;DO RECAL CMD
4045 : MOV DRVNUM,L.CS2
4046 :
4047 : TLOADRK ;LOAD 'L' REGS INTO RK REGS
4048 : TWAT16
4049 : ERROR 2 ;NO INTR
4050 :
4051 : CLR INTSET ;CLR INIT FLAG
4052 :
4053 : TWAT8S ;WAIT FOR 2'N INTR
4054 : ERROR 2 ;NOT THERE
4055 :
4056 :
4057 007654 105037 007722 CLR 2$ ;CLEAR SECTOR POINTER
4058 007660 005000 CLR R0
4059 007662 005005 CLR R5 ;CLEAR R5 FOR BAD SECTOR COUNTING
4060 007664 013703 001640 MOV BSF26P,R3 ;SET POINT IN TO STORE BS 26 SECT FORMAT
4061 007670 012737 007676 001110 MOV #1$,$LPERR ;SET ERROR RETURN ADDRESS FOR INTERNAL LOOP
4062 007676 104416 1$: TSSINIT ;CLEAR SUBSYSTEM
4063 007700 104003 ERROR 3 ;BAD INIT ERROR
4064 :
4065 007702 013737 036120 007724 MOV LSTCYL,15$
4066 :
4067 007710 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS WITH
4068 007714 000121 RDDATA ; READ DATA
4069 007716 177400 -400 ; WORD COUNT
4070 007720 070414 Ibuff ; BUFFER ADDRESS
4071 007722 000 2$: .BYTE 0 ; SECTOR ADDRESS
4072 007723 002 .BYTE 2 ; TRACK ADDRESS
4073 007724 000000 15$: 0 ; CYLINDER ADDRESS 632/1456
4074 :
4075 007726 104417 TLOADRK ;LOAD 'L' REGS INTO RK
4076 007730 104431 TWAT112 ;WAIT FOR INTERRUPT
4077 007732 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4078 007734 104421 TCHKOP ;CHECK FOR ANY ERRORS
4079 007736 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4080 007740 104415 SCOP1 ;LOOP TO 1$ IF SW 9 SET
4081 :
4082 007742 105737 001103 TSTB $ERFLG ;TEST FOR ERROR IN OPERATION
4083 007746 001502 BEQ 7$ ;NO-SKIP
4084 007750 005700 TST R0 ;GETTING A BS FACTORY SECTOR 26 SECT FORMAT?
4085 007752 001023 BNE 3$ ;NO-SKIP
4086 007754 062737 000002 007722 ADD #2,2$ ;NEXT SECTOR ADDRESS
```

4087	007762	122737	000012	007722		CMPB	#10.,2\$:PAST APPLICABLE SECTORS?
4088	007770	001066				BNE	6\$:NO-SKIP
4089	007772	012737	056036	001360		MOV	#EM51,EM1N	
4090	010000	104001				ERROR	1	: "CANNOT READ BS FILES
4091	010002	043737	001630	001354	25\$:	BIC	DRVBIT,\$DEV	: CLEAR DRIVE FROM DRIVE MAP
4092	010010	053737	001630	001666		BIS	DRVBIT,DRVDRP	: SET DRIVE DROPPED
4093	010016	000137	027716			JMP	NEWDRV	: ABORT TEST PASS.
4094								
4095	010022	022700	000001		3\$:	CMP	#1,R0	: GETTING A BS SOFT SECTOR 26 SECT FORMAT?
4096	010026	001014				BNE	4\$: NO-SKIP
4097	010030	062737	000002	007722		ADD	#2,2\$: NEXT SECTOR ADDRESS
4098	010036	122737	000026	007722		CMPB	#22.,2\$: PAST APPLICABLE SECTORS?
4099	010044	001040				BNE	6\$: NO-SKIP
4100	010046	012737	056036	001360		MOV	#EM51,EM1N	
4101	010054	104001				ERROR	1	: "CANNOT READ BS FILES"
4102	010056	000751				BR	25\$	
4103								
4104	010060	022700	000002		4\$:	CMP	#2,R0	: GETTING A BS FACT SECTOR 24 SECTOR FORMAT?
4105	010064	001014				BNE	5\$: NO-SKIP
4106	010066	062737	000002	007722		ADD	#2,2\$: NEXT SECTOR ADDRESS
4107	010074	122737	000013	007722		CMPB	#11.,2\$: PAST APPLICABLE SECTORS?
4108	010102	001021				BNE	6\$: NO-SKIP
4109								
4110	010104	012737	056036	001360		MOV	#EM51,EM1N	
4111	010112	104001				ERROR	1	: "CANNOT READ BS FILES"
4112	010114	000732				BR	25\$	
4113								
4114	010116	062737	000002	007722	5\$:	ADD	#2,2\$: NEXT SECTOR (BS SOFT 24 SECT MODE)
4115	010124	122737	000027	007722		CMPB	#23.,2\$: PAST APPLICABLE SECTORS?
4116	010132	001005				BNE	6\$: NO-SKIP
4117	010134	012737	056036	001360		MOV	#EM51,EM1N	
4118	010142	104001				ERROR	1	: "CANNOT READ BS FILES
4119	010144	000716				BR	25\$	
4120								
4121	010146	105037	001103		6\$:	CLRB	\$ERFLG	: CLEAR ERROR FLAG
4122	010152	000651				BR	1\$: DO NEXT READ
4123								
4124	010154	005737	070422		7\$:	TST	IBUFF+6	: CHECK FOR ALIGNMENT PACK
4125	010160	001405				BEQ	8\$: NO-SKIP
4126	010162	012737	056131	001360		MOV	#EM52,EM1N	
4127	010170	104001				ERROR	1	: "ALIGNMENT PACK. DRIVE ABORTING"
4128	010172	000703				BR	25\$	
4129								
4130	010174	012701	070424		8\$:	MOV	#IBUFF+10,R1	: SET TO START OF BAD SECTOR DATA
4131								
4132	010200	022711	177777		9\$:	CMP	#-1,(R1)	: TEST IF WORD ALL ONES (END OF DATA)
4133	010204	001422				BEQ	11\$: YES-SKIP
4134	010206	012123				MOV	(R1)+,(R3)+	: STORE CYLINDER
4135	010210	012123				MOV	(R1)+,(R3)+	: TRACK AND SECTOR
4136	010212	005205				INC	R5	: BUMP ERROR COUNTER
4137	010214	022705	000025			CMP	#21.,R5	: DOES IT TOTAL 20 FOR THIS FORMAT?
4138	010220	001367				BNE	9\$: NO-TEST AND MORE NEXT ADDRESS
4139	010222	012737	056227	001360		MOV	#EM53,EM1N	
4140	010230	104001				ERROR	1	: TO MANY BAD SECTORS
4141	010232	043737	001630	001354	10\$:	BIC	DRVBIT,\$DEV	: CLEAR DRIVE FROM TESTING
4142	010240	053737	001630	001666		BIS	DRVBIT,DRVDRP	: SET DRIVE DROPPED

```
4143 010246 000137 027716      JMP      NEWDRV      ;ABORT PASS
4144
4145 010252 005200      11$:    INC      R0      ;BUMP TO NEXT
4146 010254 022700 000001      CMP      #1,R0      ;NOW TESTING BS SOFT 26 SECTOR MODE?
4147 010260 001011      BNE     12$         ;NO-SKIP
4148 010262 012723 177777      MOV     #-1,(R3)+   ;INSERT END OF FIELD FLAG
4149 010266 010337 001644      MOV     R3,BSS26P   ;SET POINTER TO BAD SECTOR SOFTWARE FIELD
4150 010272 112737 000012 007722      MOVB   #12,2$      ;SET TO FIRST SECTOR THIS MODE
4151 010300 000137 007676      JMP     1$         ;GO READ IT.
4152 010304 022700 000002      12$:    CMP      #2,R0      ;NOW TESTING BS FACT 24 SECTOR MODE?
4153 010310 001014      BNE     13$         ;NO-SKIP
4154 010312 012723 177777      MOV     #-1,(R3)+   ;INSERT END OF FIELD FLAG
4155 010316 112737 000001 007722      MOVB   #1,2$        ;SET TO FIRST SECTOR THIS MODE
4156 010324 010537 001646      MOV     R5,BS26CT   ;STORE TOTAL BS COUNT 26 SECTOR MODE
4157 010330 005005      CLR     R5          ;CLEAR COUNTER FOR COUNTING 24 SECT BS
4158 010332 013703 001636      MOV     BSF24P,R3   ;SET POINTER FOR STORING BS
4159 010336 000137 007676      JMP     1$         ;GO READ
4160
4161 010342 022700 000003      13$:    CMP      #3,R0      ;NOW TESTING BS SOFT 24 SECTOR MODE
4162 010346 001011      BNE     14$         ;NO-SKIP
4163 010350 012723 177777      MOV     #-1,(R3)+   ;INSERT END OF FIELD FLAG
4164 010354 010337 001642      MOV     R3,BSS24P   ;STORE POINTER TO BSS 24 SECTOR MODE
4165 010360 112737 000013 007722      MOVB   #13,2$      ;GET START OF FIELDS BSS 24 SECT MODE
4166 010366 000137 007676      JMP     1$         ;GO READ IT
4167
4168 010372 012723 177777      14$:    MOV     #-1,(R3)+   ;INSERT END OF FIELD FLAG
4169 010376 010537 001650      MOV     R5,BS24CT   ;STORE COUNT BSS 24 SECTOR MODE
4170
4171
4172 010402 012700 061144      MOV     #BS26,R0    ;GET START OF BAD SECTOR BUFFER
4173      MOV     #BS24+52.,R3 ;GET END OF BUFFER
4174      :
4175      :27$:   CMP     #312,(R0)+  ;TEST IF ANY SECTORS BAD IN CYL 312
4176      BEQ     26$        ;YES - GET OUT
4177      :
4178      :
4179      :
4180      :
4181 010406 005003      CLR     R3          ;COUNT
4182 010410 022710 177777      27$:    CMP     #-1,(R0)   ;AT THE END OF TABLE ?
4183 010414 001404      BEQ     16$        ;BRANCH IF SO
4184 010416 022720 000312      CMP     #312,(R0)+ ;CYL 312 BAD ?
4185 010422 001372      BNE     27$        ;BRANCH IF NOT TRY AGAIN
4186 010424 000424      BR      26$        ;ELSE ERROR
4187 010426 005203      16$:    INC     R3         ;
4188 010430 020327 000001      CMP     R3,#1      ;SEE IF 1 ST HALF OF BS 26 TABLE
4189 010434 001003      BNE     17$        ;BRANCH IF NOT
4190 010436 013700 001644      MOV     BSS26P,R0  ;MOVE UP THE TABLE
4191 010442 000762      BR      27$        ;
4192 010444 020327 000002      17$:    CMP     R3,#2      ;SEE IF DID 2 ND HALF OF BS26 TABLE
4193 010450 001003      BNE     18$        ;BRANCH IF NOT
4194 010452 012700 061270      MOV     #BS24,R0   ;ELSE MOVE UP 1 ST HALF
4195 010456 000754      BR      27$        ;
4196 010460 020327 000003      18$:    CMP     R3,#3      ;SEE IF DID 1 ST HALF OF BS24 TABLE
4197 010464 001003      BNE     19$        ;BRANCH IF NOT
4198 010466 013700 001642      MOV     BSS24P,R0  ;ELSE MOVE UP 2ND HALF OF BS24
```



```
4199 010472 000746 BR 27$
4200 010474 000406 19$: BR 28$ ;ALL DONE
4201
4202 010476 012737 057110 001360 26$: MOV #DRVABT,EM1N ;'BAD SECTOR IN AREA FOR TEST''
4203 010504 104001 ERROR 1
4204 010506 000137 010002 JMP 25$
4205
4206 010512 28$:
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217 010512 000004
4218 010514 012737 000012 001262
4219 010522 012737 000312 001676
4220 010530 105037 010653
4221 010534 105037 010557
4222 010540 104416
4223 010542 104003
4224
4225 010544 004437 035674 9$: JSR R4,LRLOAD ;LOAD 'L' REGS
4226 010550 000127 WRHEAD ;WRITE HEADER
4227 010552 177676 -102 ;WORD CNT FOR 26 SECTOR MODE
4228 010554 072414 OBUFF ;BUFFER
4229 010556 000 .BYTE 0 ;SECTOR 0
4230 010557 000 11$: .BYTE 0 ;TRACK 0
4231 010560 000312 312 ;CYL 0
4232
4233 010562 004437 042260 JSR R4,GENCOM ;GENERATE DATA
4234 010566 000600 000600 ;BUILD HEADERS-NO BAD SECTORS
4235 010570 012737 010600 001110 MOV #111$,$LPERR ;SET LOCAL LOOP ON ERROR
4236 010576 000402 BR 1$ ;SKIP INIT
4237 010600 111$:
4238 010600 104416 TSSINIT ;CLEAR SUBSYSTEM
4239 010602 104003 ERROR 3 ;BAD INIT ERROR
4240 010604 104417 1$: TLOADRK ;LOAD RK REGS
4241 010606 104431 TWAT112 ;WAIT FOR INTERRUPT
4242 010610 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4243
4244 010612 104421 TCHKOP ;CHECK FOR ANY ERRORS
4245 010614 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4246
4247 010616 104415 SCOP1 ;INTERNAL LOOP TO 111$
4248
4249 010620 012737 010660 001110 MOV #112$,$LPERR ;SET LOCAL LOOP ON ERROR
4250 010626 010203 MOV R2,R3 ;BUILD POINTER TO RKDB
4251 010630 062703 000024 ADD #RKDB,R3
4252 010634 012701 070414 MOV #IBUFF,R1 ;SET POINTER TO BUFFER
4253 010640 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4254 010644 000125 RDHEAD ;READ HEADER
```

4255	010646	000000				0		:NO WORDS COUNT
4256	010650	000000				0		:NO BUFFER
4257	010652	000				.BYTE 0		:SECTOR 0
4258	010653	000			10\$:	.BYTE 0		: TRACK 0
4259	010654	000312				312		: CYL 312
4260								
4261	010656	000402				BR 2\$:SKIP INIT
4262	010660				112\$:			
4263	010660	104416				TSSINIT		:CLEAR SUBSYSTEM
4264	010662	104003				ERROR 3		:BAD INIT ERROR
4265	010664	104417			2\$:	TLOADRK		:LOAD RK REGS
4266	010666	104423				TWAT16		:WAIT FOR INTERRUPT
4267	010670	104002				ERROR 2		:TO SLOW/NOT COMPLETE ERROR
4268								
4269	010672	104421				TCHKOP		:CHECK FOR ANY ERRORS
4270	010674	104004				ERROR 4; OR 5,6,7		:REPORT ALL ERRORS
4271	010676	012700	000003			MOV #3,R0		:SET COUNT
4272	010702	011321			5\$:	MOV (R3),(R1)+		:GET RKDB
4273	010704	104420				TGETRK		:GET RK REGS
4274	010706	032737	100000	001550		BIT #DLT,T.CS2		:TEST IF DATA LATE
4275	010714	001410				BEQ 3\$:NO-SKIP
4276	010716	012737	054631	054260		MOV #EMDLT,EM13		
4277	010724	012737	056777	061112		MOV #EMRDB,DF011A		
4278	010732	104013				ERROR 13		: 'DATA LATE SET RESULT OF DB READ
4279	010734	104415				SCOPI		: LOCAL LOOP TO 112\$
4280								
4281	010736	032737	100000	001540	3\$:	BIT #CERR,T.CS1		:TEST IF CONT ERROR SET
4282	010744	001410				BEQ 4\$:NO-SKIP
4283	010746	012737	057023	001500		MOV #EMCERR,EM13N		
4284	010754	012737	056777	061112		MOV #EMRDB,DF011A		
4285	010762	104013				ERROR 13		: 'CERR SET RESULT OF READ DB
4286	010764	104415				SCOPI		: LOCAL LOOP TO 112\$
4287	010766	005300			4\$:	DEC R0		:DEC COUNT
4288	010770	001344				BNE 5\$:LOOP IF NOT ZERO
4289	010772	012737	011002	001110		MOV #117\$,\$LPERR		:SET LOCAL LOOP 117\$
4290	011000	000402				BR 7\$:SKIP INIT
4291	011002				117\$:			
4292	011002	104416				TSSINIT		:CLEAR SUBSYSTEM
4293	011004	104003				ERROR 3		:BAD INIT ERROR
4294	011006	004437	036362		7\$:	JSR R4,RDSTHD		:GO READ & SEQUENCE HEADERS
4295	011012	104421				TCHKOP		:CONTROLLER ERROR RETURN
4296	011014	104004				ERROR 4; OR 5,6,7		:REPORT ALL ERRORS
4297	011016	104013				ERROR 13		: 'DATA LATE SET RESULT OF DATA BUFFER READ'
4298	011020	104002				ERROR 2		: 'OPERATION TO SLOW' MESSAGE
4299								:OR 'HEADER 0 NOT FOUND' MESSAGE
4300								
4301	011022	004437	042260			JSR R4,GENCOM		
4302	011026	100200				100200		:COMPARE IBUF & OBUF (HEADERS)
4303	011030	000414				BR 6\$:GOOD RETURN-NO MISCOMPARES
4304	011032	104015				ERROR 15		:REPORT 1ST MISCOMPARES
4305								
4306	011034	013700	001634			MOV ERRLMT,R0		:GET ERROR LIMIT
4307	011040	005300			12\$:	DEC R0		:DECREMENT IT
4308	011042	001407				BEQ 6\$:EXIT IF ZERO
4309	011044	004437	042260			JSR R4,GENCOM		
4310								

4311	011050	040000			040000		:RESUME COMPARE
4312	011052	000403			BR 6\$:GOOD RETURN-NO MORE ERRORS
4313	011054	104016			ERROR 16		:REPORT NEXT ERROR LINE
4314	011056	000770			BR 12\$:LOOP
4315	011060	104415			SCOP1		:LOCAL ERROR LOOP TO 117\$
4316							
4317	011062	105737	001607		6\$: TSTB L.DT		:WAS TRACK 1 JUST DONE?
4318	011066	001010			BNE 8\$:YES-SKIP
4319							
4320	011070	112737	000001	010557	MOVB #1,11\$:CHANGE PARAM TO LOAD "L" WITH
4321	011076	112737	000001	010653	MOVB #1,10\$:TRACK 2
4322	011104	000137	010544		JMP 9\$:JUMP TO DO ENTIRE TEST ON TRK 1
4323							
4324	011110				8\$:		
4325							
4326					.SBTTL **HEADER RECOGNITION TESTS		
4327							
4328					*****		
4329					:TEST 22 BAD SECTOR ERROR		
4330							
4331					:*		
4332					:* FORMAT CYLINDER 312, TRACK 0, ON SCRATCH PACK TO HAVE		
4333					:* SECTOR 0 (BIT 15 OR WORD 2 OF HEADER RESET) AND SECTOR 1		
4334					:* (BIT 14 OR WORD 2 OF HEADER RESET) TO BE BAD SECTORS		
4335					:* AND ALL OTHER SECTORS GOOD.		
4336					:*		
4337					:* ISSUE A WRITE DATA OR 400 WORDS TO CYLINDER 312, TRACK 0,		
4338					:* SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS. ISSUE A		
4339					:* WRITE DATA TO CYLINDER 0, TRACK 0, SECTOR 1 OF 400 WORDS.		
4340					:* MAKE SURE BAD SECTOR ERROR SET. ISSUE A WRITE DATA		
4341					:* OF 400 WORDS TO CYLINDER 0, TRACK 0, SECTOR 2. MAKE		
4342					:* SURE NO ERROR SETS.		
4343					:*		
4344	011110	000004			TST22: SCOPE		
4345	011112	012737	000012	001262	MOV #10, \$TIMES		:DO 10. ITERATIONS
4346	011120	012737	000312	001676	MOV #312, REFMT		:SET REFORMAT SWITCH
4347	011126	104416			TSSINIT		:CLEAR SUBSYSTEM
4348	011130	104003			ERROR 3		:BAD INIT ERROR
4349							
4350	011132	004437	035674		JSR R4, LRLOAD		:LOAD "L" REGS.
4351	011136	000127			WRHEAD		:WRITE HEADER
4352	011140	177676			-102		:WORD COUNT FOR 26 SECTOR MODE
4353	011142	072414			OBUFF		:BUFFER ADDRESS
4354	011144	000			.BYTE 0		:SECTOR
4355	011145	000			.BYTE 0		:TRACK
4356	011146	000312			312		:CYLINDER
4357							
4358							
4359	011150	004437	042260		JSR R4, GENCOM		:GENERATE HEADERS
4360	011154	000600			600		:WITH NO BS BITS
4361							
4362	011156	012700	072416		MOV #OBUFF+2, R0		:RESET BIT 15 IN WORD 2 OF
4363	011162	042720	100000		BIC #BIT15, (R0)+		:SECTOR 0 HEADER AND BIT 14
4364	011166	042720	100000		BIC #BIT15, (R0)+		:IN WORD 2 OF SECTOR 1 HEADER.
4365	011172	005720			TST (R0)+		:ALSO CORRECT THE VRC
4366	011174	042720	040000		BIC #BIT14, (R0)+		

```
4367 011200 042710 040000 BIC #BIT14,(R0)
4368
4369 011204 104417 TLOADRK ;LOAD RK REGS
4370 011206 104431 TWAT112 ;WAIT FOR INTERRUPT
4371 011210 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4372
4373 011212 104421 TCHKOP ;CHECK IF ANY ERRORS
4374 011214 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4375 011216 012737 011224 001110 MOV #4$, $LPERR ;SET LOCAL LOOP ON ERROR
4376 011224 104416 4$: TSSINIT
4377 011226 104003 ERROR 3
4378 011230 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
4379 011234 000123 WRDATA ;WRITE DATA
4380 011236 177400 -400 ;WORD COUNT
4381 011240 072414 OBUFF ;BUS ADDRESS
4382 011242 000 5$: .BYTE 0 ;SECT 0
4383 011243 000 .BYTE 0 ;TRACK 0
4384 011244 000312 312 ;CYL 312
4385
4386 011246 104417 1$: TLOADRK ;LOAD RK REGS
4387 011250 104424 TWAT32 ;WAIT FOR INTERRUPT
4388 011252 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4389
4390 011254 022737 000002 011242 CMP #2,5$ ;JUST READ SECTOR 2?
4391 011262 001415 BEQ 6$ ;YES - SKIP
4392
4393 011264 104422 TCHKWE ;CHECK OPERATION WITH ERROR
4394 011266 000000 0
4395 011270 000100 100 ;EXPECTED BSE
4396 011272 000000 0
4397 011274 104004 ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
4398
4399 011276 104415 SCOPI ;LOCAL ERROR LOOP TO 4$
4400
4401 011300 122737 000002 011242 CMPB #2,5$ ;WAS SECTOR SET TO 2
4402 011306 001405 BEQ 7$ ;YES-SKIP
4403 011310 105237 011242 INCB 5$ ;BUMP TO NEXT SECTOR
4404 011314 000743 BR 4$ ;LOOP
4405
4406 011316 104421 6$: TCHKOP ;CHECK FOR GOOD OPERATION
4407 011320 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4408
4409 011322 7$:
4410 ;*****
4411 ;*TEST 23 HEADER VRC ERROR
4412 ;*
4413 ;* FORMAT CYLINDER 312, TRACK 0, ON SCRATCH PACK TO HAVE
4414 ;* 16 SECTORS WITH BAD HEADER VRC. ISSUE A WRITE DATA
4415 ;* OF EACH OF THE SECTORS WITH A BAD HEADER VRC. MAKE
4416 ;* SURE HEADER VRC ERROR SETS. ISSUE A WRITE DATA TO
4417 ;* A GOOD HEADER AND MAKE SURE NO ERROR OCCURS.
4418 ;*
4419 ;*****
4420 011322 000004 TST23: SCOPE
4421 011324 012737 000012 001262 MOV #10, $TIMES ;DO 10. ITERATIONS
4422 011332 012737 000312 001676 MOV #312,REFMT ;SET REFORMAT SWITCH
```

4423	011340	104416			TSSINIT				:CLEAR SUBSYSTEM
4424	011342	104003			ERROR	3			:BAD INIT ERROR
4425									
4426	011344	004437	035674		JSR	R4,LRLOAD			:LOAD "L" REGS
4427	011350	000127			WRHEAD				:WRITE HEADER
4428	011352	177676			-102				:WORD COUNT
4429	011354	072414			OBUFF				:BUFF ADD
4430	011356	000			.BYTE	0			:SECT
4431	011357	000			.BYTE	0			:TRACK
4432	011360	000312			312				:CYL
4433									
4434	011362	004437	042260		JSR	R4,GENCOM			
4435	011366	000600			600				:BUILD HEADERS NO BSE
4436									
4437	011370	012700	072420		MOV	#OBUFF+4,R0			:GET ADDRESS OF VRC HDRO
4438	011374	012703	000001		MOV	#BIT0,R3			:SET FOR BIT CHANGE SELECT
4439	011400	030310		1\$:	BIT	R3,(R0)			:CHECK A VRC BIT
4440	011402	001402			BEQ	2\$:SKIP IF ZERO
4441	011404	040310			BIC	R3,(R0)			:ELSE CLEAR IT
4442	011406	000401			BR	3\$:SKIP
4443	011410	050310		2\$:	BIS	R3,(R0)			:IF ZERO SET IT
4444	011412	062700	000006	3\$:	ADD	#6,R0			:BUMP TO NEXT VRC WORD
4445									
4446	011416	006303			ASL	R3			:SHIFT THE SELECT
4447	011420	001367			BNE	1\$:IF BIT NOT SHIFTED OUT-LOOP
4448									
4449	011422	104417			TLOADRK				:LOAD RK REGS
4450	011424	104431			TWAT112				:WAIT FOR INTERRUPT
4451	011426	104002			ERROR	2			:TO SLOW/NOT COMPLETE ERROR
4452	011430	104421			TCHKOP				:CHECK OPERATION COMPLETE
4453	011432	104004			ERROR	4 ; OR 5,6,7			:REPORT ALL ERRORS
4454									
4455	011434	012737	011442 001110		MOV	#4\$,\$LPERR			:SET LOCAL LOOP
4456	011442	104416		4\$:	TSSINIT				:CLEAR SUBSYSTEM
4457	011444	104003			ERROR	3			:BAD INIT ERROR
4458									
4459	011446	004437	035674		JSR	R4,LRLOAD			:LOAD "L" REGS
4460	011452	000123			WRDATA				:WRITE DATA
4461	011454	177400			-400				:WORD COUNT
4462	011456	072414			OBUFF				:BUFFER ADD
4463	011460	000		5\$:	.BYTE	0			:SECT
4464	011461	000			.BYTE	0			:TRACK
4465	011462	000312			312				:CYL
4466									
4467	011464	104417			TLOADRK				:LOAD RK REG
4468	011466	104424			TWAT32				:WAIT FOR INTERRUPT
4469	011470	104002			ERROR	2			:TO SLOW/NOT COMPLETE ERROR
4470									
4471	011472	022737	000020 011460		CMP	#16.,5\$:WAS THIS WRITE SECTOR 16?
4472	011500	001415			BEQ	6\$:YES-SKIP
4473									:ELSE
4474	011502	104422			TCHKWE				:CHECK OPERATION WITH ERROR
4475	011504	000000			0				
4476	011506	000040			40				:HVRC EM EXPECTED
4477	011510	000000			0				
4478	011512	104004			ERROR	4 ; OR 5,6,7			:REPORT ANY DISCREPENCIES

```
4479
4480 011514 104415          SCOP1          ;LOCAL LOOP TO 4$
4481
4482 011516 105237 011460    INCB          5$          ;BUMP SECTOR IN "L" REG
4483 011522 022737 000016 011460    CMP          #16,5$      ;IF SECTOR IS 16 OR LESS
4484 011530 003744          BLE          4$          ;LOOP
4485 011532 000402          BR           7$          ;ELSE EXIT
4486 011534 104421          TCHKOP        ;CHECK LAST OPERATION NO ERRORS
4487 011536 104004          ERROR       4 ; OR 5,6,7 ;REPORT ALL ERRORS
4488
4489 011540          7$:
```

4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545

011540 000004
011542 012737 000012 001262
011550 012737 000312 001676
011556 104416
011560 104003

011562 004437 035674
011566 000127
011570 177676
011572 072414
011574 000
011575 000
011576 000312

011600 004437 042260
011604 000600

011606 042737 100000 072416

011614 104417
011616 104431
011620 104002

011622 104421
011624 104004

011626 004437 035674
011632 000123
011634 177400
011636 072414
011640 000
011641 000
011642 000312

011644 104417
011646 104424
011650 104002

011652 104422
011654 000000
011656 000040
011660 000000
011662 104004

```
*****  
:TEST 24 BAD SECTOR ERROR AND HVRC ERROR  
:*****  
:      FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR ZERO HAS  
:      BOTH A BAD SECTOR ERROR AND HEADER VRC.  ISSUE A WRITE DATA  
:      TO CYLINDER 0, TRACK 0, SECTOR 0.  MAKE SURE ONLY HEADER VRC  
:      ERROR SETS.  
:*****  
TST24: SCOPE  
MOV #10, $TIMES ;DO 10. ITERATIONS  
MOV #312, REFMT ;SET REFORMAT SWITCH  
TSSINIT ;CLEAR SUBSYSTEM  
ERROR 3 ;BAD INIT ERROR  
  
JSR R4, LRLOAD ;LOAD "L" REG  
WRHEAD ;WRITE HEADER  
-102 ;WORD CNT FOR 26 SECTOR MODE  
OBUFF ;BUFF ADD  
.BYTE 0 ;SECTOR  
.BYTE 0 ;TRACK  
312 ;CYLINDER  
  
JSR R4, GENCOM  
600 ;BUILD HEADERS-NO BSE  
  
BIC #BIT15, OBUFF+2 ;CLEAR BIT TO SET BSE, LEAVE VRC BAD.  
  
TLOADRK ;LOAD RK REGS  
TWAT112 ;WAIT FOR INTERRUPT  
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR  
  
TCHKOP ;CHECK FOR ANY ERRORS  
ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS  
  
JSR R4, LRLOAD ;LOAD "L" REGS  
WRDATA ;WRITE DATA  
-400 ;WORD COUNT  
OBUFF ;BUFF ADD  
.BYTE 0 ;SECTOR  
.BYTE 0 ;TRACK  
312 ;CYLINDER  
  
TLOADRK ;LOAD RK REGS  
TWAT32 ;WAIT FOR INTERRUPT  
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR  
  
TCHKWE ;CHECK OPERATION WITH EXPECTED ERR  
0  
40 ;HVRC ERR EXPECTED  
0  
ERROR 4 ; OR 5,6,7 ;REPORT ALL DISCREPENCIES
```

:TEST 25 OPERATION INCOMPLETE
:*****

```
4546
4547
4548
4549
4550
4551 011664 000004
4552 011666 012737 000012 001262
4553 011674 012737 000312 001676
4554 011702 104416
4555 011704 104003
4556
4557 011706 004437 035674
4558 011712 000127
4559 011714 177676
4560 011716 072414
4561 011720 000
4562 011721 000
4563 011722 000312
4564
4565 011724 004437 042260
4566 011730 000600
4567
4568 011732 052737 001000 072614
4569 011740 052737 001000 072616
4570
4571 011746 104417
4572 011750 104431
4573 011752 104002
4574
4575 011754 104421
4576 011756 104004
4577
4578 011760 004437 035674
4579 011764 000123
4580 011766 177400
4581 011770 072414
4582 011772 025
4583 011773 000
4584 011774 000312
4585
4586 011776 104417
4587 012000 104425
4588 012002 104002
4589
4590 012004 104422
4591 012006 000000
4592 012010 000020
4593 012012 000000
4594 012014 104004
4595
4596
4597
4598
4599
4600
4601
```

TST25: SCOPE
MOV #10, \$TIMES ;DO 10. ITERATIONS
MOV #312, REFMT ;SET REFORMAT SWITCH
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
JSR R4, LRLOAD ;LOAD "L" REGS
WRHEAD ;WRITE HEADER
-102 ;WORD COUNT FOR 26 SECT MODE
OBUFF ;BUFF ADD
.BYTE 0 ;SECTOR
.BYTE 0 ;TRACK
312 ;CYLINDER
JSR R4, GENCOM
600 ;BUILD HEADERS-NO BSE ERRORS
BIS #BIT9, OBUFF+200 ;CHANGE FORMAT IN SECTOR 25
BIS #BIT9, OBUFF+202 ;CORRECT THE VRC
TLOADRK ;LOAD RK REGS
TWAT112 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE
TCHKOP ;CHECK FOR ANY ERRORS
ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
JSR R4, LRLOAD ;LOAD "L" REGS
WRDATA ;WRITE DATA
-400 ;400 WORDS
OBUFF ;BUFF ADD
.BYTE 25 ;SECTOR 25
.BYTE 0 ;TRACK 0
312 ;CYL 312
TLOADRK ;LOAD RK REGS
TWAT48 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE
TCHKWE ;CHECK OPERATION EXPECTED ERROR
0
20 ;OPI EXPECTED
0
ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES

TST 26 OPI WITH HVRC ERROR

FORMAT CYLINDER 312, TRACK 0 SUCH THAT A HEADER VRC
ERROR IS PRESENT AND SECTOR 17 HAS THE WRONG FORMAT.
ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 312,
TRACK 0, SECTOR 17. THAT BOTH OPERATION INCOMPLETE


```
4602
4603
4604
4605 012016 000004
4606 012020 012737 000012 001262
4607 012026 012737 000312 001676
4608 012034 104416
4609 012036 104003
4610
4611 012040 004437 035674
4612 012044 000127
4613 012046 177676
4614 012050 072414
4615 012052 000
4616 012053 000
4617 012054 000312
4618
4619 012056 004437 042260
4620 012062 000600
4621
4622 012064 012700 072550
4623 012070 052720 001000
4624 012074 052720 001000
4625 012100 062700 000004
4626 012104 032710 000001
4627 012110 001403
4628 012112 042710 000001
4629 012116 000402
4630 012120 052710 000001
4631
4632 012124 104417
4633 012126 104431
4634 012130 104002
4635
4636 012132 104421
4637 012134 104004
4638
4639 012136 004437 035674
4640 012142 000123
4641 012144 177400
4642 012146 072414
4643 012150 017
4644 012151 000
4645 012152 000312
4646
4647 012154 104417
4648 012156 104425
4649 012160 104002
4650
4651 012162 104422
4652 012164 000000
4653 012166 000060
4654 012170 000000
4655 012172 104004
4656
4657
```

AND HEADER VRC SET.

TST26: SCOPE

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

MOV #312, REFMT ;SET REFORMAT SWITCH

TSSINIT ;CLEAR SUBSYSTEM

ERROR 3 ;BAD INIT ERROR

JSR R4, LRLOAD ;LOAD "L" REGS

WRHEAD ;WRITE HEADER

-102 ;WORD COUNT FOR 26 SECT MODE

OBUFF ;BUS ADDRESS

.BYTE 0 ;SECTOR

.BYTE 0 ;TRACK

312 ;CYLINDER

JSR R4, GENCOM

600 ;BUILD HEADER- NO BSE ERRORS

MOV #OBUFF+134, R0 ;GET ADDRESS 2ND WORD HDR 17(8)

BIS #BIT9, (R0)+ ;SET FORMAT 24 SECT PER TRACK

BIS #BIT9, (R0)+ ;SET VRC BIT

ADD #4, R0 ;BUMP TO HVRC WORD HDR 20(8)

BIT #BIT0, (R0) ;TEST BIT 0

BEO 1\$;RESET-SKIP

BIC #BIT0, (R0) ;CLEAR BIT

BR 2\$

1\$: BIS #BIT0, (R0) ;SET BIT

2\$: TLOADRK ;FORCE OPI AND HVRC ERROR

TWAT112 ;LOAD RK REGS

ERROR 2 ;WAIT FOR INTERRUPT

2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK FOR ANY ERRORS

ERROR 4 ; OR 5,6,7 ;YES-REPORT ALL ERRORS

JSR R4, LRLOAD ;LOAD "L" REGS

WRDATA ;WRITE DATA

-400 ;400 WORDS

OBUFF ;BUFF ADDRESS

.BYTE 17 ;SECT 17

.BYTE 0 ;TRACK 0

312 ;CYLINDER 312

TLOADRK ;LOAD RK REGS

TWAT48 ;WAIT FOR INTERRUPT

ERROR 2 ;TO SLOW/NOT COMPLETE

TCHKWE ;CHECK WITH EXPECTED ERROR

0

60 ;HVRC ERR & OPI EXPECTED

0

ERROR 4 ;OR 5,6,7

*TEST 27 HVRC IGNORE ON NON-ADDRESSED SECTOR

```

4658
4659
4660
4661
4662
4663
4664
4665 012174 000004
4666 012176 012737 000012 001262
4667 012204 012737 000312 001676
4668
4669 012212 104416
4670 012214 104003
4671
4672 012216 004437 035674
4673 012222 000127
4674 012224 177676
4675 012226 072414
4676 012230 000
4677 012231 000
4678 012232 000312
4679
4680 012234 004437 042260
4681 012240 000600
4682
4683 012242 012700 072560
4684 012246 012701 000002
4685 012252 030110
4686 012254 001402
4687 012256 040110
4688 012260 000401
4689 012262 050110
4690
4691 012264 104417
4692 012266 104431
4693 012270 104002
4694
4695 012272 104421
4696 012274 104004
4697
4698 012276 004437 035674
4699 012302 000123
4700 012304 177400
4701 012306 072414
4702 012310 021
4703 012311 000
4704 012312 000312
4705 012314 104417
4706 012316 104424
4707 012320 104002
4708
4709 012322 104421
4710 012324 104004
4711
4712
4713

```

:*
 :* FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR 20 HAS AN HVRC
 :* ERROR. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 312, TRACK 0,
 :* AND SECTOR 21. MAKE SURE HVRC IS NOT SET AT THE
 :* END OF THE OPERATION
 :*
 :*****
 TST27: SCOPE
 MOV #10, \$TIMES ;DO 10. ITERATIONS
 MOV #312, REFMT ;SET REFORMAT SWITCH
 TSSINIT ;CLEAR SUBSYSTEM
 ERROR 3 ;BAD INIT ERROR
 JSR R4, LRLOAD ;LOAD "L" REGISTERS
 WRHEAD ;WRITE HEADER
 -102 ;WORD COUNT FOR 26 SECTOR MODE
 OBUFF ;BUFF ADD
 .BYTE 0 ;SECTOR
 .BYTE 0 ;TRACK
 312 ;CYLINDER
 JSR R4, GENCOM
 600 ;BUILD HEADERS-NO BSE ERRORS
 MOV #OBUFF+144, R0 ;ADDRESS OF HEAD 20 HVRC WORD
 MOV #BIT1, R1 ;BIT 1 CONSTANT
 BIT R1, (R0) ;TEST BIT 1 SET
 BEQ 1\$;RESET-SKIP
 BIC R1, (R0) ;ELSE CLEAR BIT 1
 BR 2\$;SKIP
 1\$: BIS R1, (R0) ;SET BIT 1
 2\$: TLOADRK ;LOAD RK REGS
 TWAT112 ;WAIT FOR INTERRUPT
 ERROR 2 ;TO SLOW/NOT COMPLETE
 TCHKOP ;CHECK FOR ANY ERROR
 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
 JSR R4, LRLOAD ;LOAD "L" REGISTER
 WRDATA ;WRITE DATA
 -400 ;WORD COUNT
 OBUFF ;BUFF ADD
 .BYTE 21 ;SECTOR
 .BYTE 0 ;TRACK
 312 ;CYLINDER
 TLOADRK ;LOAD RK REGS
 TWAT32 ;WAIT FOR INTERRUPT
 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
 TCHKOP ;CHECK FOR ANY ERROR
 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS.

.SBTTL **DATA TRANSFER TESTS

```
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724 012326 000004
4725 012330 012737 000012 001262
4726 012336 012737 000312 001676
4727 012344 104416
4728 012346 104003
4729
4730 012350 012737 000312 012376
4731 012356 105037 012375
4732
4733 012362 004437 035674 1$: JSR R4,LRLOAD ;LOAD 'L' REG
4734 012366 000127 WRHEAD ;WRITE HEADER
4735 012370 177676 -102 ;WORD COUNT FOR 26 SECTOR MODE
4736 012372 072414 OBUFF ;BUFF ADDRESS
4737 012374 000 .BYTE 0 ;SECTOR
4738 012375 000 2$: .BYTE 0 ;TRACK
4739 012376 000312 7$: 312 ;CYLINDER
4740
4741 012400 004437 042260 JSR R4,GENCOM
4742 012404 001200 1200 ;BUILD HDRS-INCLUDE BAD SECTORS
4743
4744 012406 104417 TLOADRK ;LOAD RK REGS
4745 012410 104431 TWAT112 ;WAIT FOR INTERRUPT
4746 012412 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4747
4748 012414 104421 TCHKOP ;CHECK FOR ANY ERRORS
4749 012416 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4750
4751 012420 022737 000313 012376 CMP #313,7$ ;TEST IF DONE 313 TK 0
4752 012426 001414 BEQ 3$ ;YES - SKIP
4753 012430 123727 012375 000002 CMPB 2$,#2 ;DID WE JUST FORMAT TRACK 2
4754 012436 001403 BEQ 8$ ;YES-SKIP
4755 012440 105237 012375 INCB 2$ ;BUMP TO NEXT TRACK
4756 012444 000746 BR 1$ ;GO FORMAT NEXT TRACK
4757
4758 012446 105037 012375 8$: CLRB 2$ ;CLEAR TRACK POINTER
4759 012452 005237 012376 INC 7$ ;BUMP CYL TO 313
4760 012456 000741 BR 1$ ;GO FORMAT 313 TK 0
4761
4762 012460 004437 035674 3$: JSR R4,LRLOAD ;LOAD 'L' REGS
4763 012464 000123 WRDATA ;WRITE DATA
4764 012466 177400 -400 ;ONE SECTOR WORD COUNT
4765 012470 072414 OBUFF ;BUFF ADDRESS
4766 012472 012 .BYTE 12 ;SECTOR 12
4767 012473 000 .BYTE 0 ;TRACK 0
4768 012474 000312 312 ;CYLINDER 312
4769
```

```

4770 012476 004437 042260 JSR R4,GENCOM
4771 012502 000001 1 ;BUILD DATA PATTERN 1
4772 012504 000400 400 ;400 WORDS LONG
4773 012506 012737 012514 001110 MOV #4$, $LPERR ;SET FOR LOCAL LOOP
4774 012514 104417 4$: TLOADRK ;LOAD RK REGS
4775 012516 104431 TWAT112 ;WAIT FOR INTERRUPT
4776 012520 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4777
4778 012522 104421 TCHKOP ;CHECK FOR ANY ERRORS
4779 012524 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4780
4781 012526 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
4782 012532 000121 RDDATA ;READ DATA
4783 012534 177400 -400 ;400 WORDS
4784 012536 070414 IBUFF ;BUFF ADD
4785 012540 012 .BYTE 12 ;SECTOR 12
4786 012541 000 .BYTE 0 ;TRACK 0
4787 012542 000312 312 ;CYL 312
4788
4789 012544 104417 TLOADRK ;LOAD RK
4790 012546 104424 TWAT32 ;WAIT FOR INTERRUPT
4791 012550 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
4792
4793 012552 104421 TCHKOP ;CHECK FOR ANY ERRORS
4794 012554 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4795
4796 012556 004437 042260 JSR R4,GENCOM
4797 012562 100001 100001 ;GO COMPARE DATA TO PATTERN 1
4798 012564 000400 400 ;400 WORDS LONG
4799 012566 000413 BR 6$ ;GOOD RETURN-NO DATA ERRORS
4800 012570 104015 ERROR 15 ;ERROR RETURN
4801
4802 012572 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
4803 012576 005300 5$: DEC R0 ;DEC LIMIT
4804 012600 001406 BEQ 6$ ;EXIT IF 0
4805 012602 004437 042260 JSR R4,GENCOM
4806 012606 040000 040000 ;CONTINUE COMPARE
4807 012610 000402 BR 6$ ;EXIT IF NO MORE ERRORS
4808 012612 104016 ERROR 16 ;ELSE REPORT MISCOMPARE
4809 012614 000770 BR 5$ ;LOOP
4810 012616 005037 001676 6$: CLR REFMT ;CLEAR REFORMAT SWITCH
4811
4812
4813 :*****
4814 :*TEST 31 WRITE DATA WITH BUS ADDRESS INCREMENT INHIBIT
4815 :*
4816 :* ISSUE A WRITE DATA OF ONE SECTOR TO CYLINDER 312,
4817 :* TRACK 2, SECTOR 12 WITH INHIBIT BUS
4818 :* ADDRESS INCREMENT. READ DATA BACK TO MAKE SURE
4819 :* EVERY WORD IS THE SAME AND CORRECT.
4820 :*****
4821 012622 000004 TST31: SCOPE
4822 012624 012737 000012 001262 MOV #10., $TIMES ;;DO 10. ITERATIONS
4823
4824 012632 104416 TSSINIT ;CLEAR SUBSYSTEM
4825 012634 104003 ERROR 3 ;BAD INIT ERROR
  
```

4826									
4827	012636	004437	035674		JSR	R4,LRLOAD		:LOAD 'L' REGS	
4828	012642	000123			WRDATA			:WRDATA	
4829	012644	177400			-400			: -400 WORDS	
4830	012646	072414			OBUFF			:OBUFF IS BUFF ADDRESS	
4831	012650	012			.BYTE	12		:SECTOR 12	
4832	012651	001			.BYTE	1		:TRACK 1	
4833	012652	000312			312			:CYLINDER 312	
4834									
4835	012654	052737	000020	001610	BIS	#BAI,L.CS2		:SET INCREMENT INHIBIT	
4836	012662	004437	042260		JSR	R4,GENCOM		:BUILD PATTERN	
4837	012666	000016			16			:PATTERN 16	
4838	012670	000400			400			:400 WORDS	
4839									
4840	012672	104417			TLOADRK			:LOAD RK REGS	
4841	012674	104430			TWAT96			:WAIT FOR INTERRUPT	
4842	012676	104002			ERROR	2		:TO SLOW/NOT COMPLETE ERROR	
4843									
4844	012700	104421			TCHKOP			:CHECK OPERATION FOR ANY ERRORS	
4845	012702	104004			ERROR	4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS	
4846									
4847	012704	004437	035674		JSR	R4,LRLOAD		:LOAD 'L' REGS	
4848	012710	000121			RDDATA			:RDDATA	
4849	012712	177400			-400			: -400 WORDS	
4850	012714	070414			IBUFF			:IBUFF IS BUFF ADDRESS	
4851	012716	012			.BYTE	12		:SECTOR 12	
4852	012717	001			.BYTE	1		:TRACK 1	
4853	012720	000312			312			:CYLINDER 312	
4854									
4855	012722	012700	000377		MOV	#377,R0		:SET COUNT TO SET OBUFF TO BE	
4856	012726	012701	072416		MOV	#OBUFF+2,R1		:ALL THE FIRST WORD OF PATTERN	
4857	012732	012703	072414		MOV	#OBUFF,R3			
4858									
4859	012736	011321		1\$:	MOV	(R3),(R1)+		:MOV THE WORD	
4860	012740	005300			DEC	R0			
4861	012742	001375			BNE	1\$:LOOP UNTIL ALL WORDS SET	
4862									
4863	012744	104417			TLOADRK			:LOAD RK REGS	
4864	012746	104424			TWAT32			:WAIT FOR INTERRUPT	
4865	012750	104002			ERROR	2		:TO SLOW/NOT COMPLETE ERROR	
4866									
4867	012752	104421			TCHKOP			:CHECK OPERATION FOR ANY ERRORS	
4868	012754	104004			ERROR	4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS	
4869									
4870	012756	004437	042260		JSR	R4,GENCOM		:COMPARE THE DATA	
4871	012762	100000			100000				
4872	012764	000400			400				
4873	012766	000413			BR	2\$			
4874	012770	104015			ERROR	15			
4875	012772	013700	001634		MOV	ERRLMT,R0		:GET ERROR LIMIT	
4876	012776	005300		64\$:	DEC	R0		:DECREMENT COUNT	
4877	013000	001406			BEQ	65\$:IF ZERO - EXIT	
4878	013002	004437	042260		JSR	R4,GENCOM		:CONTINUE DATA COMPARE	
4879	013006	040000			40000				
4880	013010	000402			BR	65\$:NO MORE ERRORS - EXIT	
4881	013012	104016			ERROR	16		:REPORT NEXT ERROR	

4882 013014 000770
4883 013016
4884
4885 013016
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896 013016 000004
4897 013020 012737 000012 001262
4898 013026 123727 001326 000001
4899 013034 002016
4900
4901 013036 032737 000002 001664
4902 013044 001011
4903
4904 013046 104401 051573
4905 013052 104401 051736
4906 013056 052737 000002 001664
4907 013064 104401 051752
4908 013070 000470
4909 013072 012737 013100 001110
4910 013100
4911 013100 104416
4912 013102 104003
4913
4914 013104 004437 035674
4915 013110 000123
4916 013112 177400
4917 013114 177770
4918 013116 016
4919 013117 000
4920 013120 000312
4921 013122 004437 042260
4922 013126 010010
4923 013130 001777
4924 013132 000400
4925
4926 013134 104417
4927 013136 104430
4928 013140 104002
4929
4930 013142 104421
4931 013144 104004
4932 013146 104415
4933
4934 013150 004437 042260
4935 013154 002007
4936 013156 001000
4937

BR 64\$;LOOP
65\$:
2\$:

:TEST 32 WRITE DATA ADDRESS GREATER THAN 32K
:
: ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 177770.
: MAKE SURE CORRECT DATA IS ON DISK.
:
: NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
: OF MEMORY IS PRESENT.
:

TST32: SCOPE
MOV #10,\$TIMES ;DO 10. ITERATIONS
CMPB \$MAMS1,#1 ;TEST IF >32K MEM
BGE 2\$;YES-SKIP

BIT #MEMSZB,OPTFLG ;TEST IF REPORT ALREADY MADE
BNE 1\$;YES -SKIP

TYPE ,OPR011 ;'INSUFFICIENT MEMORY DATA TRANSFER WITH
TYPE ,OPR012 ;ADDRESS >32K
BIS #MEMSZB,OPTFLG ;SET FLAG
TYPE ,OPR015 ;BYPASSED'
BR 4\$;EXIT
MOV #5\$,\$LPERR ;SET LOCAL LOOP ON ERROR ADDRESS

1\$:
2\$:
5\$:

TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR

JSR R4,LRLOAD ;LOAD 'L' REGS
WRDATA ;WRITE DATA
-400 ;400 WORDS
177770 ;BUS ADDRESS IN 32K -10 BYTES
.BYTE 16 ;SECTOR 16
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312
JSR R4,GENCOM ;GENERATE DATA
10010 ;PATTERN 10, MEM. MANAGEMENT FOR DEST.
1777 ;RELOCATION ARGUMENT
400 ;400 WORDS

TLOADRK ;LOAD RK REGS
TWAT96 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
SCOP1 ;LOCAL LOOP ON ERROR TO 5\$

JSR R4,GENCOM ;CLEAR Ibuff TO 1'S.
2007
1000

```
4938 013160 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
4939 013164 000121 RDDATA ;RDDATA
4940 013166 177400 -400 ;-400 WORDS
4941 013170 070414 Ibuff ;IBUFF IS BUFF ADDRESS
4942 013172 016 .BYTE 16 ;SECTOR 16
4943 013173 000 .BYTE 0 ;TRACK 0
4944 013174 000312 312 ;CYLINDER 312
4945 013176 104417 TLOADRK ;LOAD RK REGS
4946 013200 104424 TWAT32 ;WAIT FOR INTERRUPT
4947 013202 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4948 013204 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
4949 013206 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
4950 013210 004437 042260 JSR R4,GENCOM ;COMPARE
4951 013214 110000 110000 ;MEMORY MANAGEMENT FOR DESTINATION
4952 013216 001777 1777 ;RELOCATION ARGUMENT
4953 013220 000400 400 ;400 WORDS
4954 013222 000413 BR 4$ ;NO ERROR-SKIP
4955 013224 104015 ERROR 15 ;REPORT FIRST MISCOMPARE
4956 013226 013700 001634 MOV ERRMT,R0 ;GET ERROR LIMIT
4957 013232 005300 64$: DEC R0 ;DECREMENT COUNT
4958 013234 001406 BEQ 65$ ;IF ZERO - EXIT
4959 013236 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
4960 013242 050000 50000
4961 013244 000402 BR 65$ ;NO MORE ERRORS - EXIT
4962 013246 104016 ERROR 16 ;REPORT NEXT ERROR
4963 013250 000770 BR 64$ ;LOOP
4964 013252
4965
4966 013252 4$:
4967 *****
4968 :*TEST 33 READ DATA ADDRESS GREATER THAN 32K
4969 :*
4970 :* ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 177770.
4971 :* CHECK MEMORY FOR CORRECT TRANSFER.
4972 :*
4973 :* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
4974 :* OF MEMORY IS PRESENT.
4975 :*
4976 :*****
4977 TST33: SCOPE
4978 013254 012737 000012 001262 MOV #10,$TIMES ;DO 10. ITERATIONS
4979 013262 123727 001326 000001 CMPB $MAMS1,#1 ;CHECK IF >32K MEMORY
4980 013270 002001 BGE 2$ ;YES-SKIP
4981
4982 013272 000462 1$: BR 5$ ;EXIT
4983
4984 013274 012737 013302 001110 2$: MOV #3$,$LPERR ;SET LOCAL ERROR LOOP
4985
4986 013302 3$:
4987 013302 104416 TSSINIT ;CLEAR SUBSYSTEM
4988 013304 104003 ERROR 3 ;BAD INIT ERROR
4989 013306 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
4990 013312 000123 WRDATA ;WRDATA
4991 013314 177400 -400 ;-400 WORDS
4992 013316 072414 OBUFF ;OBUFF IS BUFF ADDRESS
4993 013320 017 .BYTE 17 ;SECTOR 17
```

```
4994 013321 000 .BYTE 0 ;TRACK 0
4995 013322 000312 312 ;CYLINDER 312
4996 013324 004437 042260 JSR R4,GENCOM ;GENERATE DATA IN OBUFF
4997 013330 000011 11 ;PATTERN 11
4998 013332 000400 400 ;400 WORDS
4999
5000 013334 104417 TLOADRK ;LOAD RK REGS
5001 013336 104430 TWAT96 ;WAIT FOR INTERRUPT
5002 013340 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5003
5004 013342 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5005 013344 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5006 013346 004437 035674 JSR R4,LRLoad ;LOAD 'L' REG
5007 013352 000121 RDDATA ;READ DATA
5008 013354 177400 -400 ;400 WORDS
5009 013356 177770 177770 ;ACROSS 32K BOUNDARY
5010 013360 017 .BYTE 17 ;SECTOR 17
5011 013361 000 .BYTE 0 ;TRACK 0
5012 013362 000312 312 ;CYL 312
5013
5014 013364 104417 TLOADRK ;LOAD RK REGS
5015 013366 104424 TWAT32 ;WAIT FOR INTERRUPT
5016 013370 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5017 013372 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5018 013374 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5019 013376 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5020 013402 120000 120000 ;MEMORY MANAGEMENT WITH SOURCE
5021 013404 001777 1777 ;RELOCATION ARGUMENT
5022 013406 000400 400 ;COMPARE 400 WORDS
5023 013410 000413 BR 5$ ;NO MISCOMPARE-EXIT
5024 013412 104015 ERROR 15 ;REPORT FIRST MISCOMPARE
5025 013414 013700 001634 MOV ERRMT,R0 ;GET ERROR LIMIT
5026 013420 005300 64$: DEC R0 ;DECREMENT COUNT
5027 013422 001406 BEQ 65$ ;IF ZERO - EXIT
5028 013424 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5029 013430 060000 60000 BR 65$ ;NO MORE ERRORS - EXIT
5030 013432 000402 ERROR 16 ;REPORT NEXT ERROR
5031 013434 104016 BR 64$ ;LOOP
5032 013436 000770
5033 013440
5034 013440
5035
5036 *****
5037 *TEST 34 WRITE DATA ADDRESS GREATER THAN 64K
5038
5039 * ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 377770.
5040 * MAKE SURE CORRECT DATA IS ON DISK.
5041
5042 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
5043 * OF MEMORY IS PRESENT.
5044 *****
5045 TST34: SCOPE
5046 013442 012737 000012 001262 MOV #10,$TIMES ;DO 10. ITERATIONS
5047 013450 123727 001326 000002 CMPB $MAMS1,#2 ;CHECK IF >64K MEMORY
5048 013456 002016 BGE 2$ ;YES-SKIP
5049 013460 032737 000002 001664 6$: BIT #MEMSZB,OPTFLG ;TEST IF REPORT FLAG SET
```



```

5050 013466 001011 BNE 1$ ;NO-SKIP
5051
5052 013470 104401 051573 TYPE ,OPR011 ;"INSUFFICIENT MEMORY-DATA XFER WITH
5053 013474 104401 051742 TYPE ,OPR013 ;ADDRESS >64K
5054 013500 104401 051752 TYPE ,OPR015 ;"BYPASSED"
5055 013504 052737 000002 001664 BIS #MEMSZB,OPTFLG ;SET FLAG
5056 013512 000467 BR 5$
5057
5058 013514 012737 013522 001110 2$: MOV #3$, $LPERR ;SET LOCAL LOOP ON ERROR
5059
5060 013522 3$:
5061 013522 104416 TSSINIT ;CLEAR SUBSYSTEM
5062 013524 104003 ERROR 3 ;BAD INIT ERROR
5063 013526 004437 042260 JSR R4,GENCOM ;GENERATE DATA, PATTERN 11
5064 013532 010011 10011 ;MEM MANAGEMENT ON DESTINATION
5065 013534 003777 3777 ;RELOCATION ARGUMENT
5066 013536 000400 400 ;400 WORDS
5067
5068 013540 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5069 013544 000523 WRDATA!BA16 ;WRITE DATA AND SET BA16
5070 013546 177400 -400 ;400 WORDS
5071 013550 177770 177770 ;ACROSS 64K BOUNDARY
5072 013552 020 .BYTE 20 ;SECTOR 20
5073 013553 000 .BYTE 0 ;TRACK 0
5074 013554 000312 312 ;CYLINDER 312
5075
5076 013556 104417 TLOADRK ;LOAD RK REGS
5077 013560 104430 TWAT96 ;WAIT FOR INTERRUPT
5078 013562 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5079
5080 013564 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5081 013566 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5082 013570 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO 1'S
5083 013574 002007 2007
5084 013576 001000 1000
5085
5086 013600 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5087 013604 000121 RDDATA ;RDDATA
5088 013606 177400 -400 ;-400 WORDS
5089 013610 070414 Ibuff ;IBUFF IS BUFF ADDRESS
5090 013612 020 .BYTE 20 ;SECTOR 20
5091 013613 000 .BYTE 0 ;TRACK 0
5092 013614 000312 312 ;CYLINDER 312
5093 013616 104417 TLOADRK ;LOAD RK REGS
5094 013620 104424 TWAT32 ;WAIT FOR INTERRUPT
5095 013622 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5096
5097 013624 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5098 013626 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5099 013630 004437 042260 JSR R4,GENCOM ;CHECK DATA
5100 013634 110000 110000 ;MEMORY MANAGEMENT WITH DESTINATION
5101 013636 003777 3777 ;RELOCATION ARGUMENT
5102 013640 000400 400 ;400 WORDS
5103 013642 000413 BR 5$ ;NO MISCOMPARES-SKIP
5104 013644 104015 ERROR 15 ;REPORT FIRST ERROR
5105

```

```
5106 013646 013700 001634      MOV      ERRMT,R0      ;GET ERROR LIMIT
5107 013652 005300      64$: DEC      R0        ;DECREMENT COUNT
5108 013654 001406      BEQ      65$          ;IF ZERO - EXIT
5109 013656 004437 042260      JSR      R4,GENCOM    ;CONTINUE DATA COMPARE
5110 013662 050000      50000
5111 013664 000402      BR       65$          ;NO MORE ERRORS - EXIT
5112 013666 104016      ERROR   16           ;REPORT NEXT ERROR
5113 013670 000770      BR       64$          ;LOOP
5114 013672
5115 013672
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126 013672 000004      TST35: SCOPE
5127 013674 012737 000012 001262      MOV      #10, $TIMES  ;;DO 10. ITERATIONS
5128 013702 123727 001326 000002      CMPB    $MAMS1,#2    ;CHECK IF >64K MEMORY
5129 013710 002001      BGE     2$           ;YES-SKIP
5130 013712 000462      1$: BR       5$      ;EXIT
5131
5132 013714 012737 000032 001110      2$: MOV      #32,$LPERR ;SET LOCAL LOOP ON ERROR
5133
5134 013722
5135 013722 104416      3$: TSSINIT      ;CLEAR SUBSYSTEM
5136 013724 104003      ERROR   3           ;BAD INIT ERROR
5137 013726 004437 035674      JSR      R4,LRLOAD   ;LOAD 'L' REGS
5138 013732 000123      WRDATA      ;WRDATA
5139 013734 177400      -400        ;-400 WORDS
5140 013736 072414      OBUF      ;OBUF IS BUFF ADDRESS
5141 013740 021          .BYTE 21          ;SECTOR 21
5142 013741 000          .BYTE 0           ;TRACK 0
5143 013742 000312      312         ;CYLINDER 312
5144 013744 004437 042260      JSR      R4,GENCOM    ;GENERATE DATA
5145 013750 000012      12          ;PATTERN 12
5146 013752 000400      400         ;400 WORDS
5147
5148 013754 104417      TLOADRK      ;LOAD RK REGS
5149 013756 104430      TWAT96      ;WAIT FOR INTERRUPT
5150 013760 104002      ERROR   2           ;TO SLOW/NOT COMPLETE ERROR
5151
5152 013762 104421      TCHKOP      ;CHECK OPERATION FOR ANY ERRORS
5153 013764 104004      ERROR   4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5154 013766 004437 035674      JSR      R4,LRLOAD   ;LOAD 'L' REGS
5155 013772 000521      RDDATA!BA16 ;READ DATA AND SET BA16
5156 013774 177400      -400        ;400 WORDS
5157 013776 177770      177770      ;ACROSS 64K BOUNDARY
5158 014000 021          .BYTE 21          ;FROM SECTOR 21
5159 014001 000          .BYTE 0           ;TRACK 0
5160 014002 000312      312         ;CYLINDER 312
5161
```

```
5162 014004 104417 TLOADRK ;LOAD RK REGS
5163 014006 104424 TWAT32 ;WAIT FOR INTERRUPT
5164 014010 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5165
5166 014012 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5167 014014 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5168 014016 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5169 014022 120000 120000 ;MEM MANAGEMENT WITH SOURCE
5170 014024 003777 3777 ;RELOCATION ARGUMENT
5171 014026 000400 400 ;400 WORDS
5172 014030 000413 BR 5$ ;NO MISCOMPARES-SKIP
5173 014032 104015 ERROR 15 ;REPORT FIRST ERROR
5174
5175 014034 013700 001634 MOV ERRMT,R0 ;GET ERROR LIMIT
5176 014040 005300 64$: DEC R0 ;DECREMENT COUNT
5177 014042 001406 BEQ 65$ ;IF ZERO - EXIT
5178 014044 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5179 014050 060000 60000
5180 014052 000402 BR 65$ ;NO MORE ERRORS - EXIT
5181 014054 104016 ERROR 16 ;REPORT NEXT ERROR
5182 014056 000770 BR 64$ ;LOOP
5183 014060 65$:
5184
5185 014060 5$:
5186 :*****
5187 :*TEST 36 WRITE DATA ADDRESS GREATER THAN 96K
5188 :*
5189 :* ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 577770.
5190 :* MAKE SURE CORRECT DATA IS ON DISK.
5191 :*
5192 :* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K
5193 :* OF MEMORY IS PRESENT.
5194 :*
5195 :*****
5196 014060 000004 TST36: SCOPE
5197 014062 012737 000012 001262 MOV #10,$TIMES ;DO 10. ITERATIONS
5198 014070 123727 001326 000003 CMPB $MAMS1,#3 ;CHECK IF >96K MEMORY
5199 014076 002016 BGE 3$ ;YES-SKIP
5200 014100 032737 000002 001664 1$: BIT #MEMSZB,OPTFLG ;TEST IF REPORT FLAG SET
5201 014106 001011 BNE 2$ ;NO-SKIP
5202
5203 014110 104401 051573 TYPE ,OPR011 ;"INSUFFICIENT MEMORY-DATA TRANSFET WITH
5204 014114 104401 051746 TYPE ,OPR014 ;ADDRESS >96K BYPASSED"
5205 014120 104401 051752 TYPE ,OPR015
5206 014124 052737 000002 001664 BIS #MEMSZB,OPTFLG ;SET REPORT FLAG
5207 014132 000463 BR 6$ 2$:
5208
5209 014134 012737 014142 001110 3$: MOV #4$,$LPERR ;SET LOCAL LOOP ON ERROR
5210
5211 014142 4$:
5212 014142 104416 TSSINIT ;CLEAR SUBSYSTEM
5213 014144 104003 ERROR 3 ;BAD INIT ERROR
5214 014146 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REG
5215 014152 001123 WRDATA:BA17 ;WRITE DATA AND BA17
5216 014154 177400 -400 ;400 WORDS FROM
5217 014156 177770 177770 ;ACROSS 96K BOUNDARY
```

```
5218 014160 022 .BYTE 22 ;TO SECTOR 22
5219 014161 000 .BYTE 0 ;TRACK 0
5220 014162 000312 312 ;CYL 312
5221 014164 004437 042260 JSR R4,GENCOM ;GENERATE DATA
5222 014170 010013 10013 ;PATTERN 13 MEM MAN WITH DEST.
5223 014172 005777 5777 ;RELOCATION ARGUMENT
5224 014174 000400 400 ;400 WORDS
5225
5226 014176 104417 TLOADRK ;LOAD RK REGS
5227 014200 104430 TWAT96 ;WAIT FOR INTERRUPT
5228 014202 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5229
5230 014204 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5231 014206 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5232
5233 014210 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5234 014214 000121 RDDATA ;RDDATA
5235 014216 177400 -400 ;-400 WORDS
5236 014220 070414 Ibuff ;IBUFF IS BUFF ADDRESS
5237 014222 022 .BYTE 22 ;SECTOR 22
5238 014223 000 .BYTE 0 ;TRACK 0
5239 014224 000312 312 ;CYLINDER 312
5240
5241 014226 104417 TLOADRK ;LOAD RK REGS
5242 014230 104424 TWAT32 ;WAIT FOR INTERRUPT
5243 014232 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5244
5245 014234 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5246 014236 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5247
5248 014240 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5249 014244 110000 110000 ;MEM MANAGEMENT WITH DESTINATION
5250 014246 005777 5777 ;RELOCATION ARGUMENT
5251 014250 000400 400 ;400 WORDS
5252 014252 000413 BR 6$ ;NO MISCOMPARES-BRANCH
5253 014254 104015 ERROR 15 ;REPORT 1ST ERROR
5254
5255 014256 013700 001634 MOV ERRLMT,RO ;GET ERROR LIMIT
5256 014262 005300 64$: DEC R0 ;DECREMENT COUNT
5257 014264 001406 BEQ 65$ ;IF ZERO - EXIT
5258 014266 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5259 014272 050000 50000
5260 014274 000402 BR 65$ ;NO MORE ERRORS - EXIT
5261 014276 104016 ERROR 16 ;REPORT NEXT ERROR
5262 014300 000770 BR 64$ ;LOOP
5263 014302 65$:
5264
5265 014302 6$:
5266
5267 :*****
5268 :*TEST 37 READ DATA ADDRESS GREATER THAN 96K
5269 :*
5270 :* ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 577770.
5271 :* CHECK MEMORY FOR CORRECT TRANSFER.
5272 :*
5273 :* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K
: OF MEMORY IS PRESENT.
```

```
5274  
5275  
5276 014302 000004  
5277 014304 012737 000012 001262  
5278 014312 123727 001326 000003  
5279 014320 002001  
5280 014322 000462  
5281  
5282 014324 012737 014332 001110  
5283  
5284 014332  
5285 014332 104416  
5286 014334 104003  
5287  
5288 014336 004437 035674  
5289 014342 000123  
5290 014344 177400  
5291 014346 072414  
5292 014350 005  
5293 014351 000  
5294 014352 000312  
5295 014354 004437 042260  
5296 014360 000014  
5297 014362 000400  
5298  
5299 014364 104417  
5300 014366 104430  
5301 014370 104002  
5302  
5303 014372 104421  
5304 014374 104004  
5305 014376 004437 035674  
5306 014402 001121  
5307 014404 177400  
5308 014406 177770  
5309 014410 005  
5310 014411 000  
5311 014412 000312  
5312  
5313 014414 104417  
5314 014416 104424  
5315 014420 104002  
5316  
5317 014422 104421  
5318 014424 104004  
5319 014426 004437 042260  
5320 014432 120000  
5321 014434 005777  
5322 014436 000400  
5323 014440 000413  
5324 014442 104015  
5325  
5326 014444 013700 001634  
5327 014450 005300  
5328 014452 001406  
5329 014454 004437 042260
```

TST37: SCOPE
MOV #10, \$TIMES ;DO 10. ITERATIONS
CMPB \$MAMS1, #3 ;CHECK IF >96K MEMORY
BGE 3\$;YES-SKIP
BR 6\$
2\$:
3\$: MOV #4\$, \$LPERR ;SET LOCAL LOOP ON ERROR
4\$:
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
JSR R4, LRLOAD ;LOAD "L" REGS
WRDATA ;WRDATA
-400 ;-400 WORDS
OBUFF ;OBUFF IS BUFF ADDRESS
.BYTE 5 ;SECTOR 5
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312
JSR R4, GENCOM ;GENERATE DATA
14 ;PATTERN 14
400 ;400 WORDS
TLOADRK ;LOAD RK REGS
TWAT96 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
JSR R4, LRLOAD ;LOAD "L" REGS
RDDATA!BA17 ;READ DATA WITH BA17 SET
-400 ;400 WORDS
177770 ;ACROSS 96K BOUNDARY
.BYTE 5 ;FROM SECTOR 5
.BYTE 0 ;TRACK 0
312 ;CYL 312
TLOADRK ;LOAD RK REGS
TWAT32 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
JSR R4, GENCOM ;COMPARE DATA
120000 ;MEM MANAGEMENT WITH SOURCE
5777 ;RELOCATION ARGUMENT
400 ;400 WORDS
BR 6\$;NO MISCOMPARES-SKIP
ERROR 15 ;REPORT FIRST ERROR
64\$: MOV ERRLMT, R0 ;GET ERROR LIMIT
DEC R0 ;DECREMENT COUNT
BEQ 65\$;IF ZERO - EXIT
JSR R4, GENCOM ;CONTINUE DATA COMPARE

5330	014460	060000			60000		
5331	014462	000402			BR	65\$:NO MORE ERRORS - EXIT
5332	014464	104016			ERROR	16	:REPORT NEXT ERROR
5333	014466	000770			BR	64\$:LOOP
5334	014470				65\$:		
5335							
5336	014470				6\$:		
5337					*****		
5338					:TEST 40	PARTIAL SECTOR WRITE DATA	
5339					*****		
5340					:*	ISSUE A WRITE DATA OF 103 WORDS TO CYLINDER 312,	
5341					:*	HEAD 0, SECTOR 0. MAKE SURE THE SECTOR WAS	
5342					:*	ZERO FILLED CORRECTLY.	
5343					*****		
5344					TST40:	SCOPE	
5345	014470	000004			MOV	#10.,\$TIMES	::DO 10. ITERATIONS
5346	014472	012737	000012	001262	TSSINIT		:CLEAR SUBSYSTEM
5347	014500	104416			ERROR	3	:BAD INIT ERROR
5348	014502	104003					
5349							
5350	014504	004437	035674		JSR	R4,LRLOAD	:LOAD "L" REG
5351	014510	000123			WRDATA		:WRITE DATA
5352	014512	177675			-103		:WORD COUNT PARTIAL SECTOR
5353	014514	072414			IBUFF		:BUFF ADDRESS
5354	014516	007			.BYTE	7	:SECTOR 7
5355	014517	000			.BYTE	0	:TRACK 0
5356	014520	000312			312		:CYLINDER 312
5357							
5358	014522	004437	042260		JSR	R4,GENCOM	:GENERATE DATA
5359	014526	000003			3		:PATTERN 3
5360	014530	000400			400		:400 WORDS
5361							
5362	014532	104417			TLOADRK		:LOAD RK REGS
5363	014534	104430			TWAT96		:WAIT FOR INTERRUPT
5364	014536	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
5365							
5366	014540	104421			TCHKOP		:CHECK FOR ANY ERROR
5367	014542	104004			ERROR	4 ; OR 5,6,7	:REPORT ALL ERROR
5368							
5369	014544	004437	035674		JSR	R4,LRLOAD	:LOAD "L" REGS
5370							
5371	014550	000121			RDDATA		:READ DATA
5372	014552	177400			-400		:ONE FULL SECTOR
5373	014554	070414			IBUFF		:BUFF ADDRESS
5374	014556	007			.BYTE	7	:SECTOR 7
5375	014557	000			.BYTE	0	:TRACK 0
5376	014560	000312			312		:CYLINDER 312
5377							
5378	014562	004437	042260		JSR	R4,GENCOM	
5379	014566	002007			2007		:CLEAR IBUFF TO ALL ONES
5380	014570	000400			400		
5381							
5382	014572	104417			TLOADRK		:LOAD RK REGS
5383	014574	104424			TWAT32		:WAIT FOR INTERRUPT
5384	014576	104002			ERROR	2	:TO SLOW/NOT COMPLETE
5385							

```
5386 014600 104421 TCHKOP ;CHECK FOR ANY ERRORS
5387 014602 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
5388
5389 014604 012701 072622 MOV #OBUFF+206,R1 ;CLEAR THE LAST 205 WORDS
5390 014610 012700 000275 MOV #275,R0 ;OF THE OUTPUT BUFFER TO ZERO
5391 014614 005021 1$: CLR (R1)+ ;TO VERIFY THE PARTIAL SECTOR
5392 014616 005300 DEC R0 ;WRITE 0 FILLED THE SECTOR
5393 014620 001375 BNE 1$
5394 014622 004437 042260 JSR R4,GENCOM
5395 014626 100000 100000 ;COMPARE OBUFF & IBUFF.
5396 014630 000400 400 ;ALL 400 WORDS
5397 014632 000413 BR 3$ ;NO ERRORS-EXIT
5398 014634 104015 ERROR 15 ;REPORT FIRST COMPARE ERROR
5399
5400 014636 013700 001634 2$: MOV ERRLMT,R0 ;GET ERROR LIMIT
5401 014642 005300 DEC R0 ;DECREMENT IT
5402 014644 001406 BEQ 3$ ;IF ZERO-EXIT
5403 014646 004437 042260 JSR R4,GENCOM
5404 014652 040000 40000 ;CONTINUE COMPARE
5405 014654 000402 BR 3$ ;NO MORE ERRORS-EXIT
5406 014656 104016 ERROR 16 ;REPORT NEXT COMPARE ERROR
5407 014660 000770 BR 2$ ;LOOP
5408
5409 014662 3$:
5410 *****
5411 *TEST 41 PARTIAL SECTOR READ DATA
5412 *
5413 * WRITE CYLINDER 312, TRACK 0, SECTOR ZERO WITH A
5414 * KNOWN CONFIGURATION. ISSUE A READ DATA OF
5415 * 103 WORDS TO CYLINDER 312, TRACK 0, SECTOR 0.
5416 * MAKE SURE ONLY 103 WORDS GET TRANSFERRED
5417 * TO MEMORY.
5418 *
5419 *****
5420 014662 000004 TST41: SCOPE
5421 014664 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
5422 014672 104416 TSSINIT ;CLEAR SUBSYSTEM
5423 014674 104003 ERROR 3 ;BAD INIT ERROR
5424
5425 014676 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5426 014702 000123 WRDATA ;WRDATA
5427 014704 177400 -400 ;-400 WORDS
5428 014706 072414 OBUFF ;OBUFF IS BUFF ADDRESS
5429 014710 017 .BYTE 17 ;SECTOR 17
5430 014711 000 .BYTE 0 ;TRACK 0
5431 014712 000312 312 ;CYLINDER 312
5432 014714 004437 042260 JSR R4,GENCOM ;GENERATE DATA
5433 014720 000004 4 ;PATTERN 4
5434 014722 000400 400 ;400 WORDS
5435
5436 014724 104417 TLOADRK ;LOAD RK REGS
5437 014726 104430 TWAT96 ;WAIT FOR INTERRUPT
5438 014730 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5439
5440 014732 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5441 014734 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
```

```
5442
5443 014736 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5444 014742 000121 RDDATA ;RDDATA
5445 014744 177675 -103 ;-103 WORDS
5446 014746 070414 Ibuff ;IBUFF IS BUFF ADDRESS
5447 014750 017 .BYTE 17 ;SECTOR 17
5448 014751 000 .BYTE 0 ;TRACK 0
5449 014752 000312 312 ;CYLINDER 312
5450 014754 004437 042260 JSR R4,GENCOM
5451 014760 002007 2007 ;CLEAR Ibuff
5452 014762 000400 400
5453
5454 014764 104417 TLOADRK ;LOAD RK REGS
5455 014766 104424 TWAT32 ;WAIT FOR INTERRUPT
5456 014770 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5457 014772 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5458 014774 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5459
5460 014776 012700 072622 MOV #OBUFF+206,R0 ;AFTER THE LAST 205 WORDS OF
5461 015002 012701 000275 MOV #275,R1 ;THE OUTPUT BUFFER TO ALL ONES.
5462 015006 012720 177777 1$: MOV #-1,(R0)+ ;THESE SHOULD ALL BE ONES IN
5463 015012 005301 DEC R1 ;IBUFF BECAUSE THE PARTIAL
5464 015014 001374 BNE 1$ ;READ FILLED ONLY 103 WORDS.
5465 015016 004437 042260 JSR R4,GENCOM ;GO COMPARE Ibuff & OBUFF
5466 015022 100000 100000
5467 015024 000400 400 ;ALL 400 WORDS
5468 015026 000413 BR 3$ ;NO ERRORS-EXIT
5469 015030 104015 ERROR 15 ;REPORT FIRST COMPARE ERROR
5470
5471 015032 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5472 015036 005300 64$: DEC R0 ;DECREMENT COUNT
5473 015040 001406 BEQ 65$ ;IF ZERO - EXIT
5474 015042 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5475 015046 040000 40000
5476 015050 000402 BR 65$ ;NO MORE ERRORS - EXIT
5477 015052 104016 ERROR 16 ;REPORT NEXT ERROR
5478 015054 000770 BR 64$ ;LOOP
5479 015056 65$:
5480
5481 015056 3$:
5482 ;*****
5483 ;*TEST 42 WRITE DATA WITH NON-EXISTENT MEMORY
5484 ;*
5485 ;* ISSUE A WRITE DATA OF 1 WORD USING ADDRESS 776000.
5486 ;* MAKE SURE NON-EXISTENT MEMORY SETS.
5487 ;*
5488 ;*****
5489 015056 000004 TST42: SCOPE
5490 015060 012737 000012 001262 MOV #10.,$TIMES ;:DO 10. ITERATIONS
5491 015066 104416 TSSINIT ;:CLEAR SUBSYSTEM
5492 015070 104003 ERROR 3 ;:BAD INIT ERROR
5493
5494 015072 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REG
5495 015076 001523 BA16!BA17!WRDATA ;BA16 & 17 SET WITH WRITE DATA
5496 015100 177777 -1 ;WORD COUNT OF 1
5497 015102 176000 176000 ;BUFF ADDRESS=IO PAGE BASE
```



```
5498 015104 013 .BYTE 13 :SECT 13
5499 015105 000 .BYTE 0 :TRACK 0
5500 015106 000312 312 :CYLINDER 312
5501
5502 015110 104417 TLOADRK :LOAD RK REGS
5503 015112 104430 TWAT96 :WAIT FOR INTERRUPT
5504 015114 104002 ERROR 2 :TO SLOW/NOT COMPLETE ERROR
5505 015116 104422 TCHKWE :CHECK OPERATION WITH ERROR
5506 015120 000000 0
5507 015122 000000 0
5508 015124 000040 NEMERR :NON-EXISTENT MEMORY ERROR
5509 015126 104004 ERROR 4 :OR5,6,7 :REPORT ANY DISCREPENCIES
5510 015130 012737 054151 001450 MOV #EM11A,EM10N :SET UP ERROR MESSAGE
5511 015136 012737 050650 061072 MOV #OPER42,DF010A :WITH SUPPORT MESSAGE
5512 015144 113700 001541 MOV T,CS1+1,R0 :GET UPPER CS1
5513 015150 042700 177774 BIC #177774,R0 :CLEAR UNUSED BITS
5514 015154 022700 000003 CMP #3,R0 :TEST IF BOTH UPPER BUS BITS SET
5515 015160 001406 BEQ 1$ :YES - SKIP
5516 015162 010037 001204 MOV R0,$REG11 :SET UP FOR ERROR REPORT
5517 015166 012737 000003 001202 MOV #3,$REG10
5518 015174 104010 ERROR 10
5519 015176 022737 176002 001544 1$: CMP #176002,T.BA :TEST IF BUSS ADDRESS LOW OKAY
5520 015204 001412 BEQ 2$ :YES - SKIP
5521 015206 012737 054123 001450 MOV #EM11,EM10N :SET UP MESSAGE
5522 015214 012737 176002 001202 MOV #176002,$REG10 :STORE VALUE FOR REPORT
5523 015222 013737 001544 001204 MOV T.BA,$REG11
5524 015230 104010 ERROR 10
5525
5526 015232 005737 001542 2$: TST T.WC :TEST IF WORD COUNT CORRECT
5527 015236 001411 BEQ 3$ :YES - SKIP
5528 015240 012737 054076 001450 MOV #EM10,EM10N :SET UP MESSAGE
5529 015246 005037 001202 CLR $REG10
5530 015252 013737 001542 001204 MOV T.WC,$REG11
5531 015260 104010 ERROR 10
5532
5533 015262 3$:
5534
5535
5536 :*****
5537 :*TEST 43 READ DATA WITH NON-EXISTENT MEMORY
5538 :*
5539 :* ISSUE A READ DATA OF 1 WORD USING ADDRESS 776000.
5540 :* MAKE SURE NON-EXISTENT MEMORY SETS.
5541 :*****
5542 015262 000004 TST43: SCOPE
5543 015264 012737 000012 001262 MOV #10,,$TIMES ;;DO 10. ITERATIONS
5544 015272 104416 TSSINIT :CLEAR SUBSYSTEM
5545 015274 104003 ERROR 3 :BAD INIT ERROR
5546
5547 015276 004437 035674 JSR R4,LRLOAD :LOAD 'L' REG
5548 015302 001521 BA16!BA17!RDDATA :BA16 & 17 WITH READ DATA
5549 015304 177777 -1 :WORD COUNT OF 1
5550 015306 176000 176000 :BUFF ADDRESS=IO PAGE BASE
5551 015310 013 .BYTE 13 :SECTOR 13
5552 015311 000 .BYTE 0 :TRACK 0
5553 015312 000312 312 :CYL 312
```

```
5554
5555 015314 104417 TLOADRK ;LOAD RK REGS
5556 015316 104430 TWAT96 ;WAIT FOR INTERRUPT
5557 015320 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5558 015322 104422 TCHKWF ;CHECK OPERATION WITH ERRORS
5559 015324 000000 0
5560 015326 000000 0
5561 015330 000040 NEMERR ;NON-EXISTENT MEMORY ERROR
5562 015332 104004 ERROR 4: OR 5,6,7 ;REPORT ALL DISCREPANCIES
5563 015334 012737 054151 001450 MOV #EM11A,EM10N ;SET MESSAGE
5564 015342 012737 050724 061072 MOV #OPER43,DF010A ;SET SUPPORT MESSAGE
5565 015350 113700 001541 MOV T.CS1+1,R0 ;GET UPPER CS1
5566 015354 042700 177774 BIC #177774,R0 ;CLEAR UNWANTED BITS
5567 015360 022700 000003 CMP #3,R0 ;TEST BOTH BUS 16 & 17 SET
5568 015364 001406 BEQ 1$ ;YES - SKIP
5569 015366 012737 000003 001202 MOV #3,$REG10 ;SET VALUES FOR REPORT
5570 015374 010037 001204 MOV R0,$REG11
5571 015400 104010 ERROR 10
5572
5573 015402 022737 176002 001544 1$: CMP #176002,T.BA ;TEST IF BUS ADDRESS CORRECT
5574 015410 001412 BEQ 2$ ;YES - SKIP
5575 015412 012737 054123 001450 MOV #EM11,EM10N ;SET MESSAGE
5576 015420 012737 176002 001202 MOV #176002,$REG10 ;SET VALUES FOR REPORT
5577 015426 013737 001544 001204 MOV T.BA,$REG11
5578 015434 104010 ERROR 10
5579
5580 015436 005737 001542 2$: TST T.WC ;TEST IF WORD COUNT CORRECT
5581 015442 001411 BEQ 3$ ;YES - SKIP
5582 015444 012737 054076 001450 MOV #EM10,EM10N ;SET MESSAGE
5583 015452 005037 001202 CLR $REG10 ;SET VALUES
5584 015456 013737 001542 001204 MOV T.WC,$REG11
5585 015464 104010 ERROR 10
5586
5587 015466 3$:
5588
5589
5590 *****
5591 :*TEST 44 UNIBUS PARITY ERROR
5592 :*
5593 :* INITIALIZE A MEMORY LOCATION WITH BAD
5594 :* PARITY. ISSUE A WRITE DATA OF 400 WORDS
5595 :* STARTING AT A LOCATION 112 WORDS BEFORE
5596 :* THE LOCATION WITH BAD PARITY. MAKE SURE
5597 :* THAT UNIBUS PARITY ERROR SETS.
5598 :*
5599 :* NOTE: THIS TEST IS ONLY EXECUTED IF
5600 :* MEMORY PARITY OPTION EXISTS FOR
5601 :* BUFFER. IT WILL NOT BE EXECUTED
5602 :* ON 'ECC' TYPE MEMORY.
5603 :*
5604 :* *****
5605 015466 000004 TST44: SCOPE
5606 015470 012737 000012 001262 MOV #10.,$TIMES ;DO 10. ITERATIONS
5607 015476 104416 TSSINIT ;CLEAR SUBSYSTEM
5608 015500 104003 ERROR 3 ;BAD INIT ERROR
5609
```

```

5610 015502 032737 000100 001664 BIT #PARPRE,OPTFLG ;TEST IF PARITY OPTION PRESENT
5611 015510 001013 BNE 1$ ;YES-SKIP
5612 015512 032737 000004 001664 BIT #MEMPYB,OPTFLG ;TEST IF PARITY OPTION REPORTED
5613 015520 001005 BNE 25$ ;YES-SKIP TO EXIT
5614 015522 104401 051452 * TYPE ,OPR010 ;PRINT BYPASS MESSAGE
5615 015526 052737 000004 001664 BIS #MEMPYB,OPTFLG ;SET OPTION REPORTED BIT
5616 015534 000137 016326 25$: JMP 4$ ;SKIP TO EXIT
5617
5618 015540 1$:
5619 015540 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5620 015544 000123 WRDATA ;WRDATA
5621 015546 177400 -400 ;-400 WORDS
5622 015550 072414 OBUFF ;OBUFF IS BUFF ADDRESS
5623 015552 010 .BYTE 10 ;SECTOR 10
5624 015553 000 .BYTE 0 ;TRACK 0
5625 015554 000312 312 ;CYLINDER 312
5626
5627 015556 032737 002000 001664 BIT #CP1170,OPTFLG ;TEST IF 11/70
5628 015564 001002 BNE 3$ ;YES - SKIP
5629 015566 005077 164100 CLR @CSRPTR ;CLEAR PARITY IE
5630
5631 015572 004437 042260 3$: JSR R4,GENCOM ;GENERATE DATA
5632 015576 000007 7 ;PATTERN 7
5633 015600 000400 400 ;400 WORDS
5634
5635 015602 012746 000340 MOV #PR7,-(SP) ;PUT PRIORITY 7 CN STACK
5636 015606 012746 015614 MOV #10$,-(SP) ;PUT ADDRESS ON STACK
5637 015612 000002 RTI ;SET PRI
5638 015614 10$:
5639
5640 015614 012737 100200 072640 MOV #100200,OBUFF+224 ;SET WORD IN BUFFER
5641 015622 032737 002000 001664 BIT #CP1170,OPTFLG ;TEST IF 11/70
5642 015630 001011 BNE 5$ ;YES - SKIP
5643 015632 012777 000004 164032 MOV #BIT2,@CSRPTR ;SET WRITE WRONG PARITY BIT
5644 015640 012737 100200 072640 MOV #100200,OBUFF+224 ;SET BAD PARITY IN MEMORY
5645 015646 012777 000001 164016 MOV #BIT0,@CSRPTR ;CLEAR CONTROL BIT, SET IE BIT
5646
5647 015654 013746 001622 5$: MOV RKPRI,-(SP) ;SET OLD PRIORITY
5648 015660 012746 015666 MOV #11$,-(SP) ;SET ADDRESS
5649 015664 000002 RTI ;RESTORE PRI
5650 015666 013704 001672 11$: MOV CSRPTR,R4 ;SET R4 WITH CSR POINTER
5651 015672 005000 CLR R0 ;SET R0 FOR COUNTER
5652 015674 012701 001662 MOV #INTSET,R1 ;SET R1 FOR POINTER TO INTERRUPT FLAG
5653 015700 012777 070400 163712 MOV #SPCHLR,@RKVEC ;SET UP INTERRUPT VECTORS FOR
5654 015706 012777 070410 164002 MOV #SPCPAR,@MMVECA ;RK611 AND PARITY ERROR
5655 015714 012737 016302 001264 MOV #2$,$ESCAPE ;SET UP ESCAPE FOR ERROR
5656 015722 104417 TLOADRK ;LOAD RK REGS
5657 015724 032737 002000 001664 BIT #CP1170,OPTFLG ;TEST IF 11/70
5658 015732 001434 BEQ 45$ ;NO - SKIP
5659 015734 012737 000016 177746 MOV #16,177746 ;SET TO DISABLE CACHE AND UNIBUS ERROR
5660 015742 012777 000000 163740 MOV #0,@KWLADD ;TURN OFF CLOCK INTERRUPTS
5661 015750 012777 000000 163166 MOV #0,@$TKS ;TURN OFF KEYBOARD
5662 015756 012737 170000 177750 MOV #170000,177750 ;SET FOR ERROR FORCE
5663
5664
5665 015764 105711 *****
40$: TSTB (R1) ;LOOP TO WAIT FOR INTERRUPT OR ABORT

```

```
5666 015766 003005 BGT 43$ ;WAIT. THE CODE BETWEEN THE STARS IS SET
5667 015770 005300 DEC R0 ;SET UP SO ALL BYTES HAVE PARITY OF 1.
5668 015772 100774 BMI 40$ ;IF THIS CODE IS CHANGED, REMEMBER ALL
5669 015774 000240 NOP ;BYTES MUST HAVE AN EVEN NUMBER OF
5670 015776 003372 BGT 40$ ;BITS.
5671 016000 000240 NOP
5672 016002 005014 43$: CLR (R4) ;CLEAR ERROR FORCE
5673 ;*****
5674
5675 016004 005037 177746 CLR 177746 ;ENABLE CACHE
5676 016010 012777 000100 163672 MOV #BIT6,@KWLADD ;TURN ON CLOCK INTERRUPTS
5677 016016 012777 000100 163120 MOV #100,@$TKS ;TURN ON KEYBOARD INTERRUPTS
5678 016024 104430 45$: TWAT96 ;WAIT FOR INTERRUPT
5679 016026 000414 BR 46$ ;TO SLOW/NOT COMPLETE ERROR
5680 016030 032737 002000 001664 BIT #CP1170,OPTFLG ;TEST IF 11/70
5681 016036 001024 BNE 48$ ;YES - SKIP
5682 016040 005077 163626 CLR @CSRPTR ;ELSE CLEAR CSR
5683 016044 005037 072640 CLR OBUFF+224 ;CLEAR THE BAD PARITY WORD
5684 016050 012777 000001 163614 MOV #1,@CSRPTR ;SET PARITY DETECT AGAIN
5685 016056 000414 BR 48$ ;SKIP
5686
5687 016060 032737 002000 001664 46$: BIT #CP1170,OPTFLG ;TEST IF 11/70
5688 016066 001007 BNE 47$ ;YES - SKIP
5689 016070 005077 163576 CLR @CSRPTR ;CLEAR CSR
5690 016074 005037 072640 CLR OBUFF+224 ;CLEAR BAD PARITY WORD
5691 016100 012777 000001 163564 MOV #1,@CSRPTR ;SET UP PARITY DETECT AGAIN
5692 016106 104002 47$: ERROR 2 ;REPORT TO SLOW ERROR
5693 016110 48$:
5694 016110 104422 TCHKWE ;CHECK OPERATION WITH ERROR
5695 016112 000000 0
5696 016114 000000 0
5697 016116 000100 UPERR ;UNIBUS PARITY ERROR
5698 016120 104004 ERROR 4: OR 5,6,7 ;REPORT ALL DISCREPANCIES
5699
5700 016122 005037 001264 CLR $ESCAPE ;CLEAR ESCAPE
5701 016126 012737 050777 061072 MOV #OPER44,DF010A ;SET MESSAGES FOR REPORT
5702 016134 012737 054123 001450 MOV #EM11,EM10N
5703 016142 023727 001544 072642 CMP T.BA,#OBUFF+226 ;CHECK IF BA IN RANGE
5704 016150 103010 BHIS 14$ ;NOT TO LOW - SKIP
5705 016152 012737 072642 001202 MOV #OBUFF+226,$REG10 ;SET VALUES FOR REPORT
5706 016160 013737 001544 001204 MOV T.BA,$REG11
5707 016166 104010 ERROR 10
5708 016170 000413 BR 16$
5709
5710 016172 023727 001544 072646 14$: CMP T.BA,#OBUFF+232 ;CHECK IF BA IN RANGE
5711 016200 101407 BLOS 16$ ;YES - SKIP
5712 016202 012737 072646 001202 MOV #OBUFF+232,$REG10 ;SET VALUES
5713 016210 012737 001544 001204 MOV #T.BA,$REG11
5714 016216 104010 ERROR 10
5715
5716 016220 012737 054076 001450 16$: MOV #EM10,EM10N ;SET MESSAGE
5717 016226 023727 001542 177513 CMP T.WC,#-265 ;CHECK IF WORD COUNT WITHIN RANGE
5718 016234 103007 BHIS 20$ ;YES - SKIP
5719 016236 012737 177513 001202 MOV #-265,$REG10 ;SET VALUES
5720 016244 013737 001542 001204 MOV T.WC,$REG11
5721 016252 104010 ERROR 10
```

5722									
5723	016254	023727	001542	177515	20\$:	CMP	T.WC,#-263	:	STILL CHECKING IF WC IN RANGE
5724	016262	101407				BLOS	2\$:	YES - SKIP
5725	016264	012737	177515	001202		MOV	#-263,\$REG10	:	SET VALUES
5726	016272	013737	001542	001204		MOV	T.WC,\$REG11		
5727	016300	104010				ERROR	10		
5728									
5729	016302				2\$:				
5730	016302	005037	001264			CLR	\$ESCAPE	:	REV 006

5731 016306 012777 034442 163304 MOV #INTHLR,@RKVEC ;RESET INT. VECTORS FOR RK06
 5732 016314 004737 034522 JSR PC,OPTTST
 5733 016320 012777 000100 162616 MOV #100,@\$TKS ;INSURE KEYBOARD IS ENABLED

5734
 5735 016326 4\$:

5736
 5737
 5738 .SBTTL **MULTIPLE SECTOR OPERATIONS

5739
 5740
 5741
 5742
 5743
 5744
 5745
 5746
 5747

 : *TEST 45 TWO SECTOR WRITE DATA (PART 1)
 : *
 : * ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312,
 : * TRACK 0, SECTOR 0. READ DATA BACK ONE SECTOR
 : * AT A TIME AND MAKE SURE IT IS CORRECT.
 : *
 : *

5748 016326 000004
 5749 016330 012737 000012 001262 TST45: SCOPE

5750 016336 104416 MOV #10.,\$TIMES ;:DO 10. ITERATIONS
 5751 016340 104003 TSSINIT ;:CLEAR SUBSYSTEM
 5752 ERROR 3 ;:BAD INIT ERROR

5753 016342 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
 5754 016346 000123 WRDATA ;:WRDATA
 5755 016350 177000 -1000 ;:-1000 WORDS
 5756 016352 072414 OBUFF ;:OBUFF IS BUFF ADDRESS
 5757 016354 000 .BYTE 0 ;:SECTOR 0
 5758 016355 000 .BYTE 0 ;:TRACK 0
 5759 016356 000312 312 ;:CYLINDER 312

5760
 5761 016360 004437 042260 JSR R4,GENCOM ;:GENERATE DATA
 5762 016364 000015 15 ;:PATTERN 15
 5763 016366 001000 1000 ;:1000 WORDS

5764
 5765 016370 104417 TLOADRK ;:LOAD RK REGS
 5766 016372 104430 TWAT96 ;:WAIT FOR INTERRUPT
 5767 016374 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR

5768
 5769 016376 104421 TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
 5770 016400 104004 ERROR 4 ;OR 5, 6, 7, 10 ;:REPORT ALL ERRORS

5771
 5772 016402 004437 042260 JSR R4,GENCOM ;:CLEAR Ibuff
 5773 016406 002007 2007 ;:TO ALL 1'S
 5774 016410 001000 1000

5775
 5776 016412 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
 5777 016416 000121 RDDATA ;:RDDATA
 5778 016420 177400 -400 ;:-400 WORDS
 5779 016422 070414 Ibuff ;:IBUFF IS BUFF ADDRESS
 5780 016424 000 .BYTE 0 ;:SECTOR 0
 5781 016425 000 .BYTE 0 ;:TRACK 0
 5782 016426 000312 312 ;:CYLINDER 312

5783
 5784 016430 104417 TLOADRK ;:LOAD RK REGS
 5785 016432 104424 TWAT32 ;:WAIT FOR INTERRUPT
 5786 016434 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR

```
5787
5788 016436 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5789 016440 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5790
5791 016442 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5792 016446 000121 RDDATA ;RDDATA
5793 016450 177400 -400 ;-400 WORDS
5794 016452 071414 Ibuff+1000 ;IBUFF+1000 IS BUFF ADDRESS
5795 016454 001 .BYTE 1 ;SECTOR 1
5796 016455 000 .BYTE 0 ;TRACK 0
5797 016456 000312 312 ;CYLINDER 312
5798
5799 016460 104417 TLOADRK ;LOAD RK REGS
5800 016462 104424 TWAT32 ;WAIT FOR INTERRUPT
5801 016464 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5802
5803 016466 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5804 016470 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5805
5806 016472 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5807 016476 100000 100000 ;
5808 016500 001000 1000 ;1000 WORDS
5809 016502 000413 BR 2$ ;NO MISCOMPARES-EXIT
5810 016504 104015 ERROR 15 ;REPORT FIRST ERROR
5811
5812 016506 013700 001634 MOV ERR_LMT,R0 ;GET ERROR LIMIT
5813 016512 005300 64$: DEC R0 ;DECREMENT COUNT
5814 016514 001406 BEQ 65$ ;IF ZERO - EXIT
5815 016516 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5816 016522 040000 40000
5817 016524 000402 BR 65$ ;NO MORE ERRORS - EXIT
5818 016526 104016 ERROR 16 ;REPORT NEXT ERROR
5819 016530 000770 BR 64$ ;LOOP
5820 016532 65$:
5821
5822 016532 2$:
5823
5824 *****
5825 :*TEST 46 TWO SECTOR WRITE DATA (PART 2)
5826 :*
5827 :* ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312,
5828 :* TRACK 0, SECTOR 23. READ DATA BACK ONE SECTOR
5829 :* AT A TIME AND MAKE SURE A MID-TRANSFER
5830 :* SEEK DID NOT TAKE PLACE.
5831 :*
5832 :* *****
5833 TST46: SCOPE
5834 016532 000004 MOV #10, $TIMES ;;DO 10. ITERATIONS
5835 016534 012737 000012 001262 TSSINIT ;CLEAR SUBSYSTEM
5836 016542 104416 ERROR 3 ;BAD INIT ERROR
5837 016544 104003
5838 016546 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5839 016552 000123 WRDATA ;WRDATA
5840 016554 177000 -1000 ;-1000 WORDS
5841 016556 072414 OBUFF ;OBUFF IS BUFF ADDRESS
5842 016560 023 .BYTE 23 ;SECTOR 23
5843 016561 000 .BYTE 0 ;TRACK 0
```

```
5843 016562 000312 312 ;CYLINDER 312
5844
5845 016564 004437 042260 JSR R4,GENCCM ;GENERATE DATA
5846 016570 000016 16 ;PATTERN 16
5847 016572 001000 1000 ;1000 WORDS
5848
5849 016574 104417 TLOADRK ;LOAD RK REGS
5850 016576 104430 TWAT96 ;WAIT FOR INTERRUPT
5851 016600 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5852
5853 016602 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5854 016604 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5855
5856 : IF THE TRACK ADDRESS AT THE END OF OPERATION IS IN ERROR
5857 : THE CONTROLLER DID A MID-TRANSFER SEEK AS THOUGH IT
5858 : WERE IN 24(8) SECTORS PER TRACK MODE.
5859
5860 016606 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5861 016612 000121 RDDATA ;RDDATA
5862 016614 177400 -400 ;-400 WORDS
5863 016616 070414 IBUFF ;IBUFF IS BUFF ADDRESS
5864 016620 023 .BYTE 23 ;SECTOR 23
5865 016621 000 .BYTE 0 ;TRACK 0
5866 016622 000312 312 ;CYLINDER 312
5867
5868 016624 004437 042260 JSR R4,GENCOM ;CLEAR IBUFF TO ALL ONES
5869 016630 002007 2007
5870 016632 001000 1000
5871
5872 016634 104417 TLOADRK ;LOAD RK REGS
5873 016636 104424 TWAT32 ;WAIT FOR INTERRUPT
5874 016640 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5875
5876 016642 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5877 016644 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5878
5879 016646 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
5880 016652 000121 RDDATA ;RDDATA
5881 016654 177400 -400 ;-400 WORDS
5882 016656 071414 IBUFF+1000 ;IBUFF+1000 IS BUFF ADDRESS
5883 016660 024 .BYTE 24 ;SECTOR 24
5884 016661 000 .BYTE 0 ;TRACK 0
5885 016662 000312 312 ;CYLINDER 312
5886
5887 016664 104417 TLOADRK ;LOAD RK REGS
5888 016666 104424 TWAT32 ;WAIT FOR INTERRUPT
5889 016670 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5890
5891 016672 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5892 016674 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5893
5894 016676 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5895 016702 100000 100000
5896 016704 001000 1000 ;1000 WORDS
5897 016706 000413 BR 1$ ;NO ERRORS-SKIP
5898 016710 104015 ERROR 15 ;REPORT FIRST ERROR
```



```
5899
5900 016712 013700 001634
5901 016716 005300
5902 016720 001406
5903 016722 004437 042260
5904 016726 040000
5905 016730 000402
5906 016732 104016
5907 016734 000770
5908 016736
5909 016736
5910
5911
5912
5913
5914
5915
5916
5917
5918 016736 000004
5919 016740 012737 000012 001262
5920 016746 104416
5921 016750 104003
5922
5923 016752 004437 035674
5924 016756 000123
5925 016760 177377
5926 016762 072414
5927 016764 010
5928 016765 000
5929 016766 000312
5930
5931 016770 004437 042260
5932 016774 000002
5933 016776 001000
5934
5935 017000 104417
5936 017002 104430
5937 017004 104002
5938
5939 017006 104421
5940 017010 104004
5941
5942 017012 012700 073416
5943 017016 012701 000377
5944 017022 005020
5945 017024 005301
5946 017026 001375
5947 017030 004437 042260
5948 017034 002007
5949 017036 001000
5950
5951 017040 004437 035674
5952 017044 000121
5953 017046 177400
5954 017050 070414
```

```
64$: MOV ERRLMT,R0 ;GET ERROR LIMIT
      DEC R0 ;DECREMENT COUNT
      BEQ 65$ ;IF ZERO - EXIT
      JSR R4,GENCOM ;CONTINUE DATA COMPARE
      40000
      BR 65$ ;NO MORE ERRORS - EXIT
      ERROR 16 ;REPORT NEXT ERROR
      BR 64$ ;LOOP

65$:
1$:
*****
*TEST 47 TWO SECTOR WRITE DATA (PART 3)
*
* ISSUE A WRITE DATA OF 401 WORDS TO CYLINDER 312,
* TRACK 0, SECTOR 10. READ DATA BACK ONE SECTOR AT
* A TIME AND CHECK ZERO FILL OF SECOND SECTOR.
*****
TST47: SCOPE
        MOV #10, $TIMES ;DO 10. ITERATIONS
        TSSINIT ;CLEAR SUBSYSTEM
        ERROR 3 ;BAD INIT ERROR
        JSR R4,LRLOAD ;LOAD "L" REGS
        WRDATA ;WRDATA
        -401 ;-401 WORDS
        OBUFF ;OBUFF IS BUFF ADDRESS
        .BYTE 10 ;SECTOR 10
        .BYTE 0 ;TRACK 0
        312 ;CYLINDER 312
        JSR R4,GENCOM ;GENERATE DATA
        2 ;PATTERN 2
        1000 ;1000 WORDS
        TLOADRK ;LOAD RK REGS
        TWAT96 ;WAIT FOR INTERRUPT
        ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
        TCHKOP ;CHECK OPERATION FOR ANY ERRORS
        ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
        CLEAR LAST 377 WORDS OF OBUFF FOR EXPECTED ZEROS FROM ZERO FILL
        MOV #OBUFF+1002,R0 ;GET STARTING ADDRESS TO BE CLEARED
        MOV #377,R1 ;NUMBER OF WORDS
        CLR (R0)+ ;CLEAR WORD
        DEC R1 ;DEC COUNT
        BNE 1$ ;LOOP UNTIL COUNT ZERO
        JSR R4,GENCOM ;CLEAR Ibuff TO ONES
        2007
        1000
        JSR R4,LRLOAD ;LOAD "L" REGS
        RDDATA ;RDDATA
        -400 ;-400 WORDS
        Ibuff ;IBUFF IS BUFF ADDRESS
```

```
5955 017052 010 .BYTE 10 ;SECTOR 10
5956 017053 000 .BYTE 0 ;TRACK 0
5957 017054 000312 312 ;CYLINDER 312
5958
5959 017056 104417 TLOADRK ;LOAD RK REGS
5960 017060 104424 TWAT32 ;WAIT FOR INTERRUPT
5961 017062 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5962
5963 017064 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5964 017066 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5965 017070 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5966 017074 000121 RDDATA ;RDDATA
5967 017076 177400 -400 ;-400 WORDS
5968 017100 071414 Ibuff+1000 ;IBUFF+1000 IS BUFF ADDRESS
5969 017102 011 .BYTE 11 ;SECTOR 11
5970 017103 000 .BYTE 0 ;TRACK 0
5971 017104 000312 312 ;CYLINDER 312
5972
5973 017106 104417 TLOADRK ;LOAD RK REGS
5974 017110 104424 TWAT32 ;WAIT FOR INTERRUPT
5975 017112 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5976
5977 017114 004437 042260 JSR R4,GENCOM ;DATA COMPARE
5978 017120 100000 100000 ;100000
5979 017122 001000 1000 ;1000 WORDS
5980 017124 000413 BR 2$ ;NO ERROR-SKIP
5981 017126 104015 ERROR 15 ;REPORT FIRST ERROR
5982
5983 017130 013700 001634 MOV ERR_LMT,R0 ;GET ERROR LIMIT
5984 017134 005300 64$ DEC R0 ;DECREMENT COUNT
5985 017136 001406 BEQ 65$ ;IF ZERO - EXIT
5986 017140 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5987 017144 040000 40000
5988 017146 000402 BR 65$ ;NO MORE ERRORS - EXIT
5989 017150 104016 ERROR 16 ;REPORT NEXT ERROR
5990 017152 000770 BR 64$ ;LOOP
5991 017154 65$:
5992
5993 017154 2$:
5994
5995 *****
5996 :*TEST 50 MID-TRANSFER SEEK ON WRITE (PART 1)
5997 :*
5998 :* ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312,
5999 :* TRACK 0, SECTOR 25. READ DATA BACK ONE SECTOR
6000 :* AT A TIME AND MAKE SURE A MID-TRANSFER SEEK
6001 :* DID TAKE PLACE.
6002 :*
6003 :* *****
6003 017154 000004 TST50: SCOPE
6004 017156 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
6005 017164 104416 TSSINIT ;CLEAR SUBSYSTEM
6006 017166 104003 ERROR 3 ;BAD INIT ERROR
6007
6008 017170 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6009 017174 000123 WRDATA ;WRDATA
6010 017176 177000 -1000 ;-1000 WORDS
```

6011	017200	072414		OBUF		:OBUF IS BUFF ADDRESS
6012	017202	025		.BYTE	25	:SECTOR 25
6013	017203	000		.BYTE	0	:TRACK 0
6014	017204	000312		312		:CYLINDER 312
6015						
6016	017206	004437	042260	JSR	R4,GENCOM	:GENERATE DATA
6017	017212	000003		3		:PATTERN 3
6018	017214	001000		1000		:1000 WORDS
6019						
6020	017216	104417		TLOADRK		:LOAD RK REGS
6021	017220	104430		TWAT96		:WAIT FOR INTERRUPT
6022	017222	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
6023						
6024	017224	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
6025	017226	104004		ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
6026			:	A TRACK ERROR PRINTED OUT AT THE END OF THE OPERATION INDICATES A		
6027			:	MID-TRANSFER HEAD SWITCH DID NOT OCCUR.		
6028	017230	004437	042260	JSR	R4,GENCOM	
6029	017234	002007		2007		
6030	017236	001000		1000		
6031						
6032	017240	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
6033	017244	000121		RDDATA		:RDDATA
6034	017246	177400		-400		: -400 WORDS
6035	017250	070414		IBUFF		:IBUFF IS BUFF ADDRESS
6036	017252	025		.BYTE	25	:SECTOR 25
6037	017253	000		.BYTE	0	:TRACK 0
6038	017254	000312		312		:CYLINDER 312
6039						
6040	017256	104417		TLOADRK		:LOAD RK REGS
6041	017260	104425		TWAT48		:WAIT FOR INTERRUPT
6042	017262	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
6043						
6044	017264	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
6045	017266	104004		ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
6046						
6047	017270	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
6048	017274	000121		RDDATA		:RDDATA
6049	017276	177400		-400		: -400 WORDS
6050	017300	071414		IBUFF+1000		:IBUFF+1000 IS BUFF ADDRESS
6051	017302	000		.BYTE	0	:SECTOR 0
6052	017303	001		.BYTE	1	:TRACK 1
6053	017304	000312		312		:CYLINDER 312
6054						
6055	017306	104417		TLOADRK		:LOAD RK REGS
6056	017310	104425		TWAT48		:WAIT FOR INTERRUPT
6057	017312	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
6058						
6059	017314	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
6060	017316	104004		ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
6061						
6062	017320	004437	042260	JSR	R4,GENCOM	:COMPARE DATA
6063	017324	100000		100000		
6064	017326	001000		1000		:1000 WORDS
6065	017330	000413		BR	1\$:NO ERRORS-SKIP
6066	017332	104015		ERROR	15	:REPORT FIRST ERROR

```
6067
6068 017334 013700 001634
6069 017340 005300
6070 017342 001406
6071 017344 004437 042260
6072 017350 040000
6073 017352 000402
6074 017354 104016
6075 017356 000770
6076 017360
6077
6078 017360
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088 017360 000004
6089 017362 012737 000012 001262
6090 017370 104416
6091 017372 104003
6092
6093 017374 004437 035674
6094 017400 000123
6095 017402 177000
6096 017404 072414
6097 017406 025
6098 017407 002
6099 017410 000312
6100
6101 017412 004437 042260
6102 017416 000004
6103 017420 001000
6104
6105 017422 104417
6106 017424 104430
6107 017426 104002
6108
6109
6110 017430 104421
6111 017432 104004
6112
6113 017434 004437 042260
6114 017440 002007
6115 017442 001000
6116
6117 017444 004437 035674
6118 017450 000121
6119 017452 177400
6120 017454 070414
6121 017456 025
6122 017457 002
```

```
64$: MOV ERR, R0 ;GET ERROR LIMIT
      DEC R0 ;DECREMENT COUNT
      BEQ 65$ ;IF ZERO - EXIT
      JSR R4, GENCOM ;CONTINUE DATA COMPARE
      40000
      BR 65$ ;NO MORE ERRORS - EXIT
      ERROR 16 ;REPORT NEXT ERROR
      BR 64$ ;LOOP

65$:

1$:
*****
: *TEST 51 MID-TRANSFER SEEK ON WRITE (PART 2)
: *
: * ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312,
: * TRACK 2, SECTOR 25. READ DATA BACK ONE SECTOR
: * AT A TIME AND MAKE SURE A MID-TRANSFER SEEK
: * DID TAKE PLACE.
*****
TST51: SCOPE
MOV #10., $TIMES ;:DO 10. ITERATIONS
TSSINIT ;:CLEAR SUBSYSTEM
ERROR 3 ;:BAD INIT ERROR

JSR R4, LRLOAD ;:LOAD 'L' REGS
WRDATA ;:WRDATA
-1000 ;:-1000 WORDS
OBUFF ;:OBUFF IS BUFF ADDRESS
.BYTE 25 ;:SECTOR 25
.BYTE 2 ;:TRACK 2
312 ;:CYLINDER 312

JSR R4, GENCOM ;:GENERATE DATA
4 ;:PATTERN 4
1000 ;:1000 WORDS

TLOADRK ;:LOAD RK REGS
TWAT96 ;:WAIT FOR INTERRUPT
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
: A CYLINDER ERROR REPORTED AT THE END OF THE OPERATION INDICATES A
: MID-TRANSFER SEEK DID NOT OCCUR.
TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;:REPORT ALL ERRORS

JSR R4, GENCOM ;:CLEAR Ibuff TO ALL ONES
2007
1000

JSR R4, LRLOAD ;:LOAD 'L' REGS
RDDATA ;:RDDATA
-400 ;:-400 WORDS
IBUFF ;:IBUFF IS BUFF ADDRESS
.BYTE 25 ;:SECTOR 25
.BYTE 2 ;:TRACK 2
```

```
6123 017460 000312          312          ;CYLINDER 312
6124
6125 017462 104417          TLOADRK        ;LOAD RK REGS
6126 017464 104425          TWAT48         ;WAIT FOR INTERRUPT
6127 017466 104002          ERROR 2       ;TO SLOW/NOT COMPLETE ERROR
6128
6129 017470 104421          TCHKOP        ;CHECK OPERATION FOR ANY ERRORS
6130 017472 104004          ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6131
6132 017474 004437 035674    JSR R4,LRLOAD  ;LOAD "L" REGS
6133 017500 000121          RDDATA        ;RDDATA
6134 017502 177400          -400         ;-400 WORDS
6135 017504 071414          Ibuff+1000   ;IBUFF+1000 IS BUFF ADDRESS
6136 017506 000          .BYTE 0      ;SECTOR 0
6137 017507 000          .BYTE 0      ;TRACK 0
6138 017510 000313          313         ;CYLINDER 313
6139
6140 017512 104417          TLOADRK        ;LOAD RK REGS
6141 017514 104425          TWAT48         ;WAIT FOR INTERRUPT
6142 017516 104002          ERROR 2       ;TO SLOW/NOT COMPLETE ERROR
6143
6144 017520 104421          TCHKOP        ;CHECK OPERATION FOR ANY ERRORS
6145 017522 104004          ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6146
6147 017524 004437 042260    JSR R4,GENCOM  ;COMPARE DATA
6148 017530 100000          100000
6149 017532 001000          1000         ;1000 WORDS
6150 017534 000413          BR 1$        ;NO MISCOMPARES-SKIP
6151 017536 104015          ERROR 15     ;REPORT 1ST ERROR
6152
6153 017540 013700 001634    MOV ERRMT,R0  ;GET ERROR LIMIT
6154 017544 005300          64$: DEC R0   ;DECREMENT COUNT
6155 017546 001406          BEQ 65$     ;IF ZERO - EXIT
6156 017550 004437 042260    JSR R4,GENCOM  ;CONTINUE DATA COMPARE
6157 017554 040000          40000
6158 017556 000402          BR 65$     ;NO MORE ERRORS - EXIT
6159 017560 104016          ERROR 16   ;REPORT NEXT ERROR
6160 017562 000770          BR 64$     ;LOOP
6161 017564
6162
6163 017564          1$:
6164          ;*****
6165          ;*TEST 52 TWO SECTOR READ DATA (PART 1)
6166          ;*
6167          ;* ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312,
6168          ;* TRACK 0, SECTOR 0, VERIFY THAT CORRECT DATA IS
6169          ;* READ.
6170          ;*
6171          ;* NOTE: TWO SECTOR WRITE DATA (PART 1) MUST BE
6172          ;* EXECUTED BEFORE THIS TEST.
6173          ;*
6174          ;*****
6175 017564 000004          TST52: SCOPE
6176 017566 012737 000012 001262 MOV #10,,$TIMES ;:DO 10. ITERATIONS
6177 017574 104416          TSSINIT      ;CLEAR SUBSYSTEM
6178 017576 104003          ERROR 3     ;BAD INIT ERROR
```

```
6179
6180 ;GENERATE SAME DATA AS USED IN TWO SECTOR WRITE DATA (PART 1)
6181
6182 : GENERATE SAME DATA AS USED IN TWO SECTOR WRITE DATA PART 1
6183 017600 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6184 017604 000015 15 ;PATTERN 15
6185 017606 001000 1000 ;1000 WORDS
6186
6187 017610 004437 042260 JSR R4,GENCOM ;CLEAR IBUFF TO ALL ONES
6188 017614 002007 2007
6189 017616 001000 1000
6190
6191 017620 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6192 017624 000121 RDDATA ;RDDATA
6193 017626 177000 -1000 ;-1000 WORDS
6194 017630 070414 IBUFF ;IBUFF IS BUFF ADDRESS
6195 017632 000 .BYTE 0 ;SECTOR 0
6196 017633 000 .BYTE 0 ;TRACK 0
6197 017634 000312 312 ;CYLINDER 312
6198
6199 017636 104417 TLOADRK ;LOAD RK REGS
6200 017640 104430 TWAT96 ;WAIT FOR INTERRUPT
6201 017642 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6202
6203 017644 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6204 017646 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6205
6206 017650 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6207 017654 100000 100000
6208 017656 001000 1000 ;1000 WORDS
6209 017660 000413 BR 1$ ;NO MISCOMPARES-SKIP
6210 017662 104015 ERROR 15
6211
6212 017664 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
6213 017670 005300 64$: DEC R0 ;DECREMENT COUNT
6214 017672 001406 BEQ 65$ ;IF ZERO - EXIT
6215 017674 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6216 017700 040000 40000
6217 017702 000402 BR 65$ ;NO MORE ERRORS - EXIT
6218 017704 104016 ERROR 16 ;REPORT NEXT ERROR
6219 017706 000770 BR 64$ ;LOOP
6220 017710 65$:
6221
6222 017710 1$:
6223 *****
6224 *TEST 53 TWO SECTOR READ DATA (PART 2)
6225 *
6226 * ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312,
6227 * TRACK 0, SECTOR 23. VERIFY THAT CORRECT DATA IS
6228 * READ AND A MID-TRANSFER SEEK DOES NOT OCCUR.
6229 *
6230 * NOTE: TWO SECTOR WRITE DATA (PART 2) MUST BE
6231 * EXECUTED BEFORE THIS TEST.
6232 *****
6233
6234 017710 000004 TST53: SCOPE
```

```

6235 017712 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
6236 017720 104416 TSSINIT ;;CLEAR SUBSYSTEM
6237 017722 104003 ERROR 3 ;;BAD INIT ERROR
6238
6239 ;GENERATE SAME DATA AS USED IN TWO SECTOR WRITE (PART 2)
6240
6241 017724 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6242 017730 000016 16 ;PATTERN 16
6243 017732 001000 1000 ;1000 WORDS
6244
6245 017734 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6246 017740 002007 2007
6247 017742 001000 1000
6248 017744 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6249 017750 000121 RDDATA ;RDDATA
6250 017752 177000 -1000 ;-1000 WORDS
6251 017754 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6252 017756 023 .BYTE 23 ;SECTOR 23
6253 017757 000 .BYTE 0 ;TRACK 0
6254 017760 000312 312 ;CYLINDER 312
6255
6256 017762 104417 TLOADRK ;LOAD RK REGS
6257 017764 104430 TWAT96 ;WAIT FOR INTERRUPT
6258 017766 104002 ERROR 2 ;TO SLOW,NOT COMPLETE ERROR
6259
6260 017770 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6261 017772 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6262
6263 017774 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6264 020000 100000 100000
6265 020002 001000 1000 ;1000 WORDS
6266 020004 000413 BR 1$ ;NO MISCOMPARES-SKIP
6267 020006 104015 ERROR 15 ;REPORT 1ST ERROR
6268
6269 020010 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
6270 020014 005300 64$: DEC R0 ;DECREMENT COUNT
6271 020016 001406 BEQ 65$ ;IF ZERO - EXIT
6272 020020 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6273 020024 040000 40000
6274 020026 000402 BR 65$ ;NO MORE ERRORS - EXIT
6275 020030 104016 ERROR 16 ;REPORT NEXT ERROR
6276 020032 000770 BR 64$ ;LOOP
6277 020034 65$:
6278
6279 020034 1$:
6280 ;*****
6281 ;*TEST 54 TWO SECTOR READ DATA (PART 3)
6282 ;*
6283 ;* ISSUE A READ DATA OF 401 WORDS TO CYLINDER 312,
6284 ;* TRACK 0, SECTOR 10. VERIFY THAT ALL 401 WORDS
6285 ;* ARE PLACED IN MEMORY.
6286 ;*
6287 ;* NOTE: TWO SECTOR WRITE DATA (PART 3) MUST BE
6288 ;* EXECUTED BEFORE THIS TEST.
6289 ;*
6290 ;*****

```

```
6291 020034 000004 TST54: SCOPE
6292 020036 012737 000012 001262 MOV #10.,$TIMES ;:DO 10. ITERATIONS
6293 020044 104416 TSSINIT ;:CLEAR SUBSYSTEM
6294 020046 104003 ERROR 3 ;:BAD INIT ERROR
6295
6296 ; GENERATE SAME DATA AS USED IN TWO SECTOR WRITE (PART 3)
6297
6298 020050 004437 042260 JSR R4,GENCOM ;:GENERATE DATA
6299 020054 000002 2 ;:PATTERN 2
6300 020056 000401 401 ;:401 WORDS
6301
6302 020060 004437 042260 JSR R4,GENCOM ;:CLEAR Ibuff TO ALL ONES
6303 020064 002007 2007
6304 020066 001000 1000
6305
6306 020070 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
6307 020074 000121 RDDATA ;:RDDATA
6308 020076 177377 -401 ;:-401 WORDS
6309 020100 070414 Ibuff ;:IBUFF IS BUFF ADDRESS
6310 020102 010 .BYTE 10 ;:SECTOR 10
6311 020103 000 .BYTE 0 ;:TRACK 0
6312 020104 000312 312 ;:CYLINDER 312
6313
6314 020106 104417 TLOADRK ;:LOAD RK REGS
6315 020110 104430 TWAT96 ;:WAIT FOR INTERRUPT
6316 020112 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
6317
6318 020114 104421 TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
6319 020116 104004 ERROR 4 ;OR 5, 6, 7, 10 ;:REPORT ALL ERRORS
6320
6321 020120 004437 042260 JSR R4,GENCOM ;:COMPARE DATA
6322 020124 100000 100000 ;:COMPARE DATA
6323 020126 000401 401 ;:401 WORDS
6324 020130 000413 BR 1$ ;:NO MISCOMPARES-SKIP
6325 020132 104015 ERROR 15 ;:PRINT FIRST ERROR
6326
6327 020134 013700 001634 MOV ERRLMT,R0 ;:GET ERROR LIMIT
6328 020140 005300 64$: DEC R0 ;:DECREMENT COUNT
6329 020142 001406 BEQ 65$ ;:IF ZERO - EXIT
6330 020144 004437 042260 JSR R4,GENCOM ;:CONTINUE DATA COMPARE
6331 020150 040000 40000
6332 020152 000402 BR 65$ ;:NO MORE ERRORS - EXIT
6333 020154 104016 ERROR 16 ;:REPORT NEXT ERROR
6334 020156 000770 BR 64$ ;:LOOP
6335 020160
6336 020160
6337
6338 ;:*****
6339 ;:*TEST 55 MID-TRANSFER SEEK ON READ (PART 1)
6340 ;:*
6341 ;:* ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312,
6342 ;:* TRACK 0, SECTOR 25. VERIFY THAT CORRECT DATA IS
6343 ;:* READ AND A MID-TRANSFER SEEK DOES OCCUR.
6344 ;:*
6345 ;:* NOTE: MID-TRANSFER SEEK ON WRITE (PART 1) MUST BE
6346 ;:* EXECUTED BEFORE THIS TEST.
```



```
6347  
6348 020160 000004  
6349 020162 012737 000012 001262  
6350 020170 104416  
6351 020172 104003  
6352  
6353  
6354 020174 004437 042260  
6355 020200 000003  
6356 020202 001000  
6357  
6358 020204 004437 042260  
6359 020210 002007  
6360 020212 001000  
6361  
6362 020214 004437 035674  
6363 020220 000121  
6364 020222 177000  
6365 020224 070414  
6366 020226 025  
6367 020227 000  
6368 020230 000312  
6369  
6370 020232 104417  
6371 020234 104430  
6372 020236 104002  
6373  
6374 020240 104421  
6375 020242 104004  
6376  
6377 020244 004437 042260  
6378 020250 100000  
6379 020252 001000  
6380 020254 000413  
6381 020256 104015  
6382  
6383 020260 013700 001634  
6384 020264 005300  
6385 020266 001406  
6386 020270 004437 042260  
6387 020274 040000  
6388 020276 000402  
6389 020300 104016  
6390 020302 000770  
6391 020304  
6392 020304  
6393  
6394  
6395  
6396  
6397  
6398  
6399  
6400  
6401  
6402
```

```
*****  
TST55: SCOPE  
MOV #10.,$TIMES ;DO 10. ITERATIONS  
TSSINIT ;CLEAR SUBSYSTEM  
ERROR 3 ;BAD INIT ERROR  
  
; GENERATE SAME DATA AS USED IN MID TRANSFER SEEK ON WRITE (PART 1)  
JSR R4,GENCOM ;GENERATE DATA  
3 ;PATTERN 3  
1000 ;1000 WORDS  
  
JSR R4,GENCOM ;CLEAR IBUFF TO ALL ONES  
2007  
1000  
  
JSR R4,LRLOAD ;LOAD 'L' REGS  
RDDATA ;RDDATA  
-1000 ;-1000 WORDS  
IBUFF ;IBUFF IS BUFF ADDRESS  
.BYTE 25 ;SECTOR 25  
.BYTE 0 ;TRACK 0  
312 ;CYLINDER 312  
  
TLOADRK ;LOAD RK REGS  
TWAT96 ;WAIT FOR INTERRUPT  
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR  
  
TCHKOP ;CHECK OPERATION FOR ANY ERRORS  
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS  
  
JSR R4,GENCOM ;COMPARE DATA  
100000  
1000 ;1000 WORDS  
BR 1$ ;NO MISCOMPARES-SKIP  
ERROR 15 ;PRINT FIRST ERROR  
  
MOV ERRLMT,R0 ;GET ERROR LIMIT  
64$: DEC R0 ;DECREMENT COUNT  
BEQ 65$ ;IF ZERO - EXIT  
JSR R4,GENCOM ;CONTINUE DATA COMPARE  
40000  
BR 65$ ;NO MORE ERRORS - EXIT  
ERROR 16 ;REPORT NEXT ERROR  
BR 64$ ;LOOP  
  
65$:  
1$:  
*****
```

```
*****  
*TEST 56 MID-TRANSFER SEEK ON READ (PART 2)  
*****  
; ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312,  
; TRACK 2, SECTOR 25. VERIFY THAT CORRECT DATA IS  
; READ AND A MID-TRANSFER SEEK DOES OCCUR.  
; NOTE: MID-TRANSFER SEEK ON WRITE (PART 2) MUST BE  
; EXECUTED BEFORE THIS TEST.  
*****
```

```
6403 .....  
6404 020304 000004 TST56: SCOPE  
6405 020306 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS  
6406 020314 104416 TSSINIT ;CLEAR SUBSYSTEM  
6407 020316 104003 ERROR 3 ;BAD INIT ERROR  
6408  
6409 ; GENERATE SAME DATA AS USED IN MID TRANSFER SEEK ON WRITE (PART 2)  
6410 020320 004437 042260 JSR R4,GENCOM ;GENERATE DATA  
6411 020324 000004 4 ;PATTERN 4  
6412 020326 001000 1000 ;1000 WORDS  
6413  
6414 020330 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES  
6415 020334 002007 2007  
6416 020336 001000 1000  
6417  
6418 020340 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS  
6419 020344 000121 RDDATA ;RDDATA  
6420 020346 177000 -1000 ;-1000 WORDS  
6421 020350 070414 Ibuff ;IBUFF IS BUFF ADDRESS  
6422 020352 025 .BYTE 25 ;SECTOR 25  
6423 020353 002 .BYTE 2 ;TRACK 2  
6424 020354 000312 312 ;CYLINDER 312  
6425  
6426 020356 104417 TLOADRK ;LOAD RK REGS  
6427 020360 104430 TWAT96 ;WAIT FOR INTERRUPT  
6428 020362 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR  
6429  
6430 020364 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS  
6431 020366 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS  
6432  
6433 020370 004437 042260 JSR R4,GENCOM ;COMPARE DATA  
6434 020374 100000 100000  
6435 020376 001000 1000 ;1000 WORDS  
6436 020400 000413 BR 1$ ;NO MISCOMPARES-SKIP  
6437 020402 104015 ERROR 15 ;REPORT FIRST ERROR  
6438  
6439 020404 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT  
6440 020410 005300 64$: DEC R0 ;DECREMENT COUNT  
6441 020412 001406 BEQ 65$ ;IF ZERO - EXIT  
6442 020414 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE  
6443 020420 040000 40000  
6444 020422 000402 BR 65$ ;NO MORE ERRORS - EXIT  
6445 020424 104016 ERROR 16 ;REPORT NEXT ERROR  
6446 020426 000770 BR 64$ ;LOOP  
6447 020430  
6448 020430  
6449 .....  
6450 ;*TEST 57 CYLINDER ADDRESS OVERFLOW (PART 1)  
6451 ;*  
6452 ;* ISSUE A READ DATA OF 400 WORDS TO CYLINDER 632,  
6453 ;* TRACK 2, SECTOR 25. MAKE SURE CYLINDER ADDRESS  
6454 ;* OVERFLOW ERROR DOES NOT OCCUR.  
6455 ;* CYLINDER 1456 IS USED FOR THE RK07.  
6456 ;*  
6457 .....  
6458 020430 000004 TST57: SCOPE
```

```

6459 020432 012737 000012 001262      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
6460 020440 104416                      TSSINIT                    ;CLEAR SUBSYSTEM
6461 020442 104003                      ERROR    3                  ;BAD INIT ERROR
6462
6463 020444 013737 036120 020466      MOV      LSTCYL,1$
6464 020452 004437 035674                      JSR      R4,LRLOAD        ;LOAD 'L' REGS
6465 020456 000121                      RDDATA
6466 020460 177400                      -400                        ;400 WORDS
6467 020462 070414                      Ibuff                    ;IBUFF IS BUFF ADDR
6468 020464      025                      .BYTE 25                    ;SECTOR 25
6469 020465      002                      .BYTE 2                      ;TRK 2
6470 020466 000000      1$: 0                      ;CYL 632/1456
6471
6472 020470 104417                      TLOADRK                    ;LOAD RK REGS
6473 020472 104434                      TWAT159                    ;WAIT FOR INTERRUPT
6474 020474 104002                      ERROR    2                  ;TO SLOW/NOT COMPLETE ERROR
6475
6476 020476 104421                      TCHKOP                      ;CHECK OPERATION FOR ANY ERRORS
6477 020500 104004                      ERROR    4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6478
6479      ;*****
6480      ;*TEST 60      CYLINDER ADDRESS OVERFLOW (PART 2)
6481      ;*
6482      ;*      ISSUE A READ DATA OF 401 WORDS TO CYLINDER 632,
6483      ;*      TRACK 2, SECTOR 25. MAKE SURE CYLINDER ADDRESS
6484      ;*      OVERFLOW ERROR DOES OCCUR.
6485      ;*      CYLINDER 1456 IS USED FOR THE RK07.
6486      ;*****
6487 020502 000004      TST60: SCOPE
6488 020504 012737 000012 001262      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
6489 020512 104416                      TSSINIT                    ;CLEAR SUBSYSTEM
6490 020514 104003                      ERROR    3                  ;BAD INIT ERROR
6491 020516 013737 036120 020540      MOV      LSTCYL,1$
6492 020524 004437 035674                      JSR      R4,LRLOAD        ;LOAD 'L' REGS
6493 020530 000121                      RDDATA
6494 020532 177377                      -401                        ;401 WORDS
6495 020534 070414                      Ibuff                    ;IBUFF IS BUFF ADDR
6496 020536      025                      .BYTE 25                    ;SEC 25
6497 020537      002                      .BYTE 2                      ;TRK 2
6498 020540 000000      1$: 0                      ;CYL 632/1456
6499
6500 020542 104417                      TLOADRK                    ;LOAD RK REGS
6501 020544 104434                      TWAT159                    ;WAIT FOR INTERRUPT
6502 020546 104002                      ERROR    2                  ;TO SLOW/NOT COMPLETE ERROR
6503
6504 020550 104422                      TCHKWE                      ;CHECK OPERATION WITH EXPECTED ERROR
6505 020552 001000                      COERR                      ;CYLINDER OVERFLOW
6506 020554 000000                      0
6507 020556 000000                      0
6508 020560 104004                      ERROR    4; OR 5,6,7      ;REPORT ANY DISCREPANCIES
6509
6510
6511      .SBTTL **18 BIT DATA TRANSFER TESTS
6512
6513      ;*****
6514      ;*TEST 61      FORMAT IN 24 SECTOR FORMAT
  
```

6515
6516
6517
6518
6519
6520
6521
6522 020562 000004
6523 020564 012737 000012 001262
6524 020572 012737 000312 001676
6525 020600 012737 020610 001110
6526 020606 005001
6527 020610
6528 020610 104416
6529 020612 104003
6530 020614 012737 010127 001600
6531 020622 053737 001720 001600
6532 020630 013737 001626 001610
6533 020636 012737 000074 001602
6534 020644 110137 001607
6535 020650 012737 072414 001604
6536 020656 012737 000312 001614
6537
6538 020664 004437 042260
6539 020670 001200
6540
6541 020672 104417
6542 020674 104434
6543 020676 104002
6544
6545 020700 104421
6546 020702 104004
6547
6548 020704 104415
6549
6550 020706 005701
6551 020710 001002
6552 020712 005201
6553 020714 000735
6554 020716 012737 020726 001110 2\$:
6555 020724 005001
6556 020726 3\$:
6557 020726 104416
6558 020730 104003
6559 020732 012737 010125 001600
6560 020740 013737 001626 001610
6561 020746 110137 001607
6562 020752 012737 000312 001614
6563
6564 020760 004437 036362
6565 020764 104421
6566 020766 104004
6567 020770 104013
6568 020772 104002
6569
6570

```
*****  
: * FORMAT CYLINDER 312, TRACK 0, AND TRACK 1 FOR 24 SECTOR  
: * FORMAT. VERIFY FORMAT AND THAT DATA LATE DID NOT  
: * OCCUR WITH WRITE HEADER ON IN READING DATA BUFFER  
: * AFTER READ HEADER.  
: *  
: *****  
TST61: SCOPE  
MOV #10, $TIMES ;DO 10. ITERATIONS  
MOV #312, REFMT ;SET REFORMAT SWITCH  
MOV #1$, $LPERR ;SET LOCAL LOOP ON ERROR  
CLR R1 ;CLEAR R1 FOR TRACK COUNTER  
  
1$:  
TSSINIT ;CLEAR SUBSYSTEM  
ERROR 3 ;BAD INIT ERROR  
MOV #WRHEAD!CFMT, L.CS1 ;SET UP FOR WRITE HEADER 24(8) SECTOR MODE  
BIS DTYPE, L.CS1 ;SET DRV TYP  
MOV DRVNUM, L.CS2 ;SET DRIVE NUMBER  
MOV #74, L.WC ;SET WORD COUNT  
MOV R1, L.DT ;LOAD TRACK ADDRESS  
MOV #OBUFF, L.BA ; SET BUS ADDRESS  
MOV #312, L.DCYL ; CYLINDER ADDRESS  
  
JSR R4, GENCOM ;GENERATE HEADER  
1200 ;INCLUDE BAD SECTOR BITS  
  
TLOADRK ;LOAD RK REGS  
TWAT159 ;WAIT FOR INTERRUPT  
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR  
  
TCHKOP ;CHECK OPERATION FOR ANY ERRORS  
ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS  
  
SCOP1 ;LOCAL LOOP ON ERROR TO 1$  
  
TST R1 ;R1 POINTING TO TRACK 0  
BNE 2$ ;NO-SKIP  
INC R1 ;BUMP TO TRACK 1  
BR 1$ ;LOOP  
MOV #3$, $LPERR ;SET LOCAL LOOP ON ERROR  
CLR R1 ;CLEAR TRACK POINTER  
  
3$:  
TSSINIT ;CLEAR SUBSYSTEM  
ERROR 3 ;BAD INIT ERROR  
MOV #RDHEAD!CFMT, L.CS1 ;LOAD READ 24(8) SECTOR FORMAT  
MOV DRVNUM, L.CS2 ;LOAD DRIVE NUMBER  
MOVB R1, L.DT ;LOAD TRACK  
MOV #312, L.DCYL ;LOAD CYLINDER  
  
JSR R4, RDSTHD ;GO READ STANDARD HEADER  
TCHKOP ;RETURN IF CERR W/O DATA LATE SET  
ERROR 4 ;OR 5, 6, 7 ;REPORT ALL OTHER ERRORS  
ERROR 13 ;REPORT DATA LATE  
ERROR 2 ;REPORT "OPERATION TO SLOW" OR "HEADER  
:0 NOT FOUND" MESSAGE
```

6571	020774	104415			SCOPI		:LOCAL LOOP TO 3\$ ON ERROR
6572	020776	004437	042260		JSR R4,GENCOM		:GENERATE & COMPARE HEADERS
6573	021002	101200			101200		:INCLUDING BAD SECTOR LISTS
6574	021004	000413			BR 4\$:NO MISCOMPARES-SKIP
6575	021006	104015			ERROR 15		:REPORT FIRST MISCOMPARE
6576							
6577	021010	013700	001634		MOV ERRMT,RO		:GET ERROR LIMIT
6578	021014	005300		64\$:	DEC R0		:DECREMENT COUNT
6579	021016	001406			BEQ 65\$:IF ZERO - EXIT
6580	021020	004437	042260		JSR R4,GENCOM		:CONTINUE DATA COMPARE
6581	021024	040000			40000		
6582	021026	000402			BR 65\$:NO MORE ERRORS - EXIT
6583	021030	104016			ERROR 16		:REPORT NEXT ERROR
6584	021032	000770			BR 64\$:LOOP
6585	021034			65\$:			
6586							
6587	021034	104415		4\$:	SCOPI		:LOCAL LOOP TO 3\$
6588	021036	005701			TST R1		:POINTING TO TRACK 1
6589	021040	001002			BNE 5\$:NO-EXIT
6590	021042	005201			INC R1		:BUMP TO TRACK 1
6591	021044	000730			BR 3\$:LOOP
6592							
6593	021046			5\$:			
6594							
6595							
6596							
6597							
6598							
6599							
6600							
6601							
6602	021046	000004					
6603	021050	012737	000012	001262	TST62: SCOPE		
6604	021056	012737	000312	001676	MOV #10,\$TIMES		:DO 10. ITERATIONS
6605	021064	104416			MOV #312,REFMT		:SET REFORMAT SWITCH
6606	021066	104003			TSSINIT		:CLEAR SUBSYSTEM
6607					ERROR 3		:BAD INIT ERROR
6608	021070	004437	042260		JSR R4,GENCOM		:GENERATE DATA
6609	021074	000013			13		:PATTERN 13
6610	021076	000400			400		:400 WORDS
6611							
6612	021100	004437	042260		JSR R4,GENCOM		:CLEAR Ibuff TO ALL ONES
6613	021104	002007			2007		
6614	021106	000400			400		
6615							
6616	021110	004437	035674		JSR R4,LRLOAD		:LOAD 'L' REGS
6617	021114	010123			WRDATA!CFMT		:WRDATA!CFMT
6618	021116	177400			-400		: -400 WORDS
6619	021120	072414			OBUFF		:OBUFF IS BUFF ADDRESS
6620	021122	000			.BYTE 0		:SECTOR 0
6621	021123	000			.BYTE 0		:TRACK 0
6622	021124	000312			312		:CYLINDER 312
6623							
6624	021126	104417			TLOADRK		:LOAD RK REGS
6625	021130	104430			TWAT96		:WAIT FOR INTERRUPT
6626	021132	104002			ERROR 2		:TO SLOW/NOT COMPLETE ERROR

```

6627
6628 021134 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6629 021136 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6630
6631 021140 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
6632 021144 010121 RDDATA:CFMT ;RDDATA:CFMT
6633 021146 177400 -400 ;-400 WORDS
6634 021150 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6635 021152 000 .BYTE 0 ;SECTOR 0
6636 021153 000 .BYTE 0 ;TRACK 0
6637 021154 000312 312 ;CYLINDER 312
6638
6639 021156 104417 TLOADRK ;LOAD RK REGS
6640 021160 104424 TWAT32 ;WAIT FOR INTERRUPT
6641 021162 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6642
6643 021164 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6644 021166 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6645
6646 021170 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6647 021174 100000 100000 ;
6648 021176 000400 400 ;400 WORDS
6649 021200 000413 BR 1$ ;NO MISCOMPARES-SKIP
5650 021202 104015 ERROR 15 ;REPORT 1ST ERROR
6651
6652 021204 013700 001634 MOV ERR,LR ;GET ERROR LIMIT
6653 021210 005300 64$: DEC R0 ;DECREMENT COUNT
6654 021212 001406 BEQ 65$ ;IF ZERO - EXIT
6655 021214 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6656 021220 040000 40000 ;
6657 021222 000402 BR 65$ ;NO MORE ERRORS - EXIT
6658 021224 104016 ERROR 16 ;REPORT NEXT ERROR
6659 021226 000770 BR 64$ ;LOOP
6660 021230 65$:
6661
6662 021230 1$:
6663
6664
6665
6666 *****
6667 *TEST 63 24 SECTOR FORMAT DATA TRANSFER (PART 2)
6668 *
6669 * ISSUE A WRITE DATA OF 1000 WORDS IN 24 SECTOR FORMAT
6670 * TO CYLINDER 312, TRACK 0, SECTOR 23. READ SECTOR BACK
6671 * AND MAKE SURE IT IS CORRECT. MAKE SURE THAT MID-TRANSFER
6672 * SEEK HAS TAKEN PLACE.
6673 *****
6674 021230 000004 TST63: SCOPE
6675 021232 012737 000012 001262 MOV #10,$TIMES ;DO 10. ITERATIONS
6676 021240 012737 000312 001676 MOV #312,REFMT ;SET REFORMAT SWITCH
6677 021246 004737 034522 JSR PC,OPTTST ;SET UP OPTIONS
6678 021252 104416 TSSINIT ;CLEAR SUBSYSTEM
6679 021254 104003 ERROR 3 ;BAD INIT ERROR
6680
6681 021256 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6682 021262 000015 15 ;PATTERN 15

```

6683	021264	001000		1000		;1000 WORDS
6684						
6685	021266	004437	042260	JSR	R4,GENCCM	;CLEAR Ibuff TO ALL ONES
6686	021272	002007		2007		
6687	021274	001000		1000		
6688						
6689	021276	004437	035674	JSR	R4,LRLOAD	;LOAD 'L' REGS
6690	021302	010123		WRDATA!CFMT		;WRDATA!CFMT
6691	021304	177000		-1000		;-1000 WORDS
6692	021306	072414		OBUFF		;OBUFF IS BUFF ADDRESS
6693	021310	023		.BYTE	23	;SECTOR 23
6694	021311	000		.BYTE	0	;TRACK 0
6695	021312	000312		312		;CYLINDER 312
6696						
6697	021314	104417		TLOADRK		;LOAD RK REGS
6698	021316	104430		TWAT96		;WAIT FOR INTERRUPT
6699	021320	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
6700						
6701	021322	104421		TCHKOP		;CHECK OPERATION FOR ANY ERRORS
6702	021324	104004		ERROR	4 ;OR 5, 6, 7, 10	;REPORT ALL ERRORS
6703						
6704	021326	004437	035674	JSR	R4,LRLOAD	;LOAD 'L' REGS
6705	021332	010121		RDDATA!CFMT		;RDDATA!CFMT
6706	021334	177000		-1000		;-1000 WORDS
6707	021336	070414		IBUFF		;IBUFF IS BUFF ADDRESS
6708	021340	023		.BYTE	23	;SECTOR 23
6709	021341	000		.BYTE	0	;TRACK 0
6710	021342	000312		312		;CYLINDER 312
6711						
6712	021344	104417		TLOADRK		;LOAD RK REGS
6713	021346	104426		TWAT64		;WAIT FOR INTERRUPT
6714	021350	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
6715						
6716	021352	104421		TCHKOP		;CHECK OPERATION FOR ANY ERRORS
6717	021354	104004		ERROR	4 ;OR 5, 6, 7, 10	;REPORT ALL ERRORS
6718						
6719	021356	004437	042260	JSR	R4,GENCOM	;COMPARE DATA
6720	021362	100000		100000		
6721	021364	001000		1000		;1000 WORDS
6722	021366	000413		BR	1\$;NO MISCOMPARES-SKIP
6723	021370	104015		ERROR	15	;REPORT FIRST ERROR
6724						
6725	021372	013700	001634	MOV	ERRLMT,R0	;GET ERROR LIMIT
6726	021376	005300		DEC	R0	;DECREMENT COUNT
6727	021400	001406		BEQ	65\$;IF ZERO - EXIT
6728	021402	004437	042260	JSR	R4,GENCOM	;CONTINUE DATA COMPARE
6729	021406	040000		40000		
6730	021410	000402		BR	65\$;NO MORE ERRORS - EXIT
6731	021412	104016		ERROR	16	;REPORT NEXT ERROR
6732	021414	000770		BR	64\$;LOOP
6733	021416					
6734						
6735	021416					
6736						
6737						
6738						

64\$:

65\$:

1\$:

.SBTTL **SPECIAL DATA TRANSFER TESTS

6739
6740
6741
6742
6743
6744
6745
6746
6747
6748
6749
6750
6751
6752
6753 021416 000004
6754 021420 012737 000012 001262
6755 021426 012737 000312 001676
6756 021434 104416
6757 021436 104003
6758
6759 021440 004437 035674
6760 021444 000127
6761 021446 177676
6762 021450 072414
6763 021452 000
6764 021453 000
6765 021454 000312
6766
6767 021456 004437 042260
6768 021462 000600
6769
6770 021464 042737 040000 072424
6771 021472 042737 040000 072426
6772
6773 021500 104417
6774 021502 104431
6775 021504 104002
6776
6777 021506 104421
6778 021510 104004
6779
6780 021512 004437 042260
6781 021516 000016
6782 021520 001000
6783
6784 021522 004437 042260
6785 021526 002007
6786 021530 001000
6787
6788 021532 004437 035674
6789 021536 000123
6790 021540 177000
6791 021542 072414
6792 021544 000
6793 021545 000
6794 021546 000312

```
*****  
*TEST 64 MULTI-SECTOR DATA TRANSFER AND BSE  
*  
* FORMAT CYLINDER 312, TRACK 0 IN 26 SECTOR FORMAT WITH  
* SECTOR 1 MARKED BAD. ISSUE A WRITE DATA OF 1000 WORDS  
* TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE BAD SECTOR  
* ERROR SETS AND R&DA IS CORRECT. READ SECTOR 0 AND  
* MAKE SURE IT IS CORRECT.  
*  
* ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0,  
* SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS AND THE  
* PREVIOUS SECTOR IS LOADED CORRECTLY INTO MEMORY.  
*****  
TST64: SCOPE  
MOV #10,$TIMES ;:DO 10. ITERATIONS  
MOV #312,REFMT ;:SET REFORMAT SWITCH  
TSSINIT ;:CLEAR SUBSYSTEM  
ERROR 3 ;:BAD INIT ERROR  
  
JSR R4,LRLOAD ;:LOAD 'L' REGS  
WRHEAD ;:WRHEAD  
-102 ;:-102 WORDS  
OBUFF ;:OBUFF IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0  
312 ;:CYLINDER 312  
  
JSR R4,GENCOM ;:BUILD HEADERS  
600  
  
BIC #BIT14,OBUFF+10 ;:MARK SECTOR 1 BAD  
BIC #BIT14,OBUFF+12 ;:CORRECT HURC  
  
TLOADRK ;:LOAD RK REGS  
TWT112 ;:WAIT FOR INTERRUPT  
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR  
  
TCHKOP ;:CHECK OPERATION FOR ANY ERRORS  
ERROR 4 ;OR 5, 6, 7 ;:REPORT ALL ERRORS  
  
JSR R4,GENCOM ;:GENERATE DATA  
16 ;:PATTERN 16  
1000 ;:1000 WORDS  
  
JSR R4,GENCOM ;:CLEAR Ibuff TO ALL ONES  
2007  
1000  
  
JSR R4,LRLOAD ;:LOAD 'L' REGS  
WRDATA ;:WRDATA  
-1000 ;:-1000 WORDS  
OBUFF ;:OBUFF IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0  
312 ;:CYLINDER 312
```



```
6851 021774 005300          64$: DEC R0          ;DECREMENT COUNT
6852 021776 001406          BEQ 65$          ;IF ZERO - EXIT
6853 022000 004437 042260   JSR R4,GENCCM   ;CONTINUE DATA COMPARE
6854 022004 040000          40000
6855 022006 000402          BR 65$          ;NO MORE ERRORS - EXIT
6856 022010 104016          ERROR 16        ;REPORT NEXT ERROR
6857 022012 000770          BR 64$          ;LOOP
6858 022014
6859
6860 022014 004437 042260   4$: JSR R4,GENCOM ;CLEAR IBUFF
6861 022020 002007          2007
6862 022022 001000          1000
6863
6864 022024 004437 035674   JSR R4,LRLOAD  ;LOAD 'L' REGS
6865 022030 000121          RDDATA         ;RDDATA
6866 022032 177000          -1000         ;-1000 WORDS
6867 022034 070414          IBUFF         ;IBUFF IS BUFF ADDRESS
6868 022036 000          .BYTE 0       ;SECTOR 0
6869 022037 000          .BYTE 0       ;TRACK 0
6870 022040 000312          312          ;CYLINDER 312
6871
6872 022042 104417          TLOADRK        ;LOAD RK REGS
6873 022044 104424          TWAT32        ;WAIT FOR INTERRUPT
6874 022046 104002          ERROR 2      ;TO SLOW, NOT COMPLETE ERROR
6875
6876 022050 104422          TCHKWE        ;CHECK OPERATION WITH EXPECTED ERROR
6877 022052 000000          0
6878 022054 000100          BSERR         ;BAD SECTOR ERROR
6879 022056 000000          0
6880 022060 104004          ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
6881
6882 022062 004437 042260   JSR R4,GENCOM ;COMPARE DATA AGAIN
6883 022066 100000          100000
6884 022070 000400          400          ;400 WORDS
6885 022072 000413          BR 5$         ;NO MISCOMPARES
6886 022074 104015          ERROR 15      ;REPORT FIRST ERROR
6887
6888 022076 013700 001634   66$: MOV ERRLMT,R0  ;GET ERROR LIMIT
6889 022102 005300          DEC R0        ;DECREMENT COUNT
6890 022104 001406          BEQ 67$      ;IF ZERO - EXIT
6891 022106 004437 042260   JSR R4,GENCOM ;CONTINUE DATA COMPARE
6892 022112 040000          40000
6893 022114 000402          BR 67$      ;NO MORE ERRORS - EXIT
6894 022116 104016          ERROR 16     ;REPORT NEXT ERROR
6895 022120 000770          BR 66$      ;LOOP
6896 022122
6897
6898 022122
6899
6900          5$:
6901          ;*****
6902          ;*TEST 65          FORMAT TEST
6903          ;*
6904          ;*          FORMAT CYLINDER 312, TRACKS 0 AND 1 IN 26 SECTOR FORMAT.
6905          ;*          MAKE SURE NO ERRORS SET. READ SECTORS 0-25 AND MAKE
6906          ;*          SURE DATA CHECK DOES NOT OCCUR.
6907          ;*****
```

```

6907 022122 000004
6908 022124 012737 000001 001262
6909 022132 005000
6910 022134 012737 022142 001110
6911
6912 022142
6913 022142 104416
6914 022144 104003
6915
6916 022146 013737 001626 001610
6917 022154 012737 000127 001600
6918 022162 053737 001720 001600
6919 022170 110037 001607
6920 022174 012737 072414 001604
6921 022202 012737 177676 001602
6922 022210 012737 000312 001614
6923
6924 022216 004437 042260
6925 022222 001200
6926
6927 022224 104417
6928 022226 104431
6929 022230 104002
6930
6931 022232 104421
6932 022234 104004
6933
6934 022236 104415
6935
6936 022240 005700
6937 022242 001002
6938 022244 005200
6939 022246 000735
6940 022250 005000
6941 022252 012737 022260 001110
6942
6943 022260
6944 022260 104416
6945 022262 104003
6946 022264
6947 022264 004437 035674
6948 022270 000121
6949 022272 177400
6950 022274 070414
6951 022276 000
6952 022277 000
6953 022300 000312
6954
6955 022302 110037 001606
6956
6957 022306 104417
6958 022310 104424
6959 022312 104002
6960
6961 022314 104421
6962 022316 104004

TST65: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
CLR R0 ;CLEAR TRACK COUNTER
MOV #1$,$LPERR ;SET LOCAL LOOP

1$:
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR

MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
MOV #WRHEAD,L.CS1 ;LOAD WRITE HEADER
BIS DTYPE,L.CS1 ;SET DRV TYP
MOV R0,L.DT ;LOAD DESIRED TRACK FROM TRACK COUNTER
MOV #OBUFF,L.BA ;LOAD BUS ADDRESS
MOV #-102,L.WC ;WORD COUNT
MOV #312,L.DCYL ;CYLINDER

JSR R4,GENCOM ;BUILD HEADER
1200 ;WITH BSE FLAGGED

TLOADRK ;LOAD RK REGS
TWAT112 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS

SCOP1 ;LOCAL LOOP TO 1$

TST R0 ;R0 AT ZERO?
BNE 2$ ;NO-EXIT
INC R0 ;BUMP COUNTER
BR 1$ ;LOOP
CLR R0 ;CLEAR SECTOR COUNTER
MOV #113$,$LPERR ;SET LOCAL LOOP ON ERROR

113$:
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR

3$:
JSR R4,LRLOAD ;LOAD "L" REGS
RDDATA ;RDDATA
-400 ;-400 WORDS
IBUFF ;IBUFF IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312

MOVB R0,L.DS ;LOAD SECTOR COUNTER INTO DESIRED SECTOR

TLOADRK ;LOAD RK REGS
TWAT32 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
  
```

```
6963
6964 022320 104415          SCOP1          ;LOCAL LOOP TO 3$ ON ERROR
6965
6966 022322 022700 000024   CMP #24,R0    ;LAST SECTOR READ?
6967 022326 001402          BEQ 4$       ;YES-EXIT
6968 022330 005200          INC R0       ;BUMP SECTOR COUNTER
6969 022332 000754          BR 3$       ;LOOP
6970
6971 022334 005037 001676   4$: CLR REFMT ;CLEAR REFORMAT SWITCH
6972
6973 .SBTTL **WRITE CHECK TESTS
6974
6975 :*****
6976 :*TEST 66 WRITE-CHECK WITH NO ERROR
6977 :*
6978 :* WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH A KNOWN PATTERN.
6979 :* DO A WRITE-CHECK OF 400 WORDS. MAKE SURE NO
6980 :* ERROR OCCURS.
6981 :*
6982 :*****
6983 022340 000004          TST66: SCOPE
6984 022342 012737 000012 001262 MOV #10, $TIMES ;:DO 10. ITERATIONS
6985 022350 104416          TSSINIT      ;:CLEAR SUBSYSTEM
6986 022352 104003          ERROR 3     ;:BAD INIT; ERROR
6987
6988 022354 004437 035674   JSR R4,LRLOAD ;:LOAD "L" REGS
6989 022360 000123          WRDATA      ;:WRDATA
6990 022362 177400          -400        ;:-400 WORDS
6991 022364 072414          OBUFF       ;:OBUFF IS BUFF ADDRESS
6992 022366 000          .BYTE 0     ;:SECTOR 0
6993 022367 000          .BYTE 0     ;:TRACK 0
6994 022370 000312          312        ;:CYLINDER 312
6995
6996 022372 004437 042260   JSR R4,GENCOM ;:GENERATE DATA
6997 022376 000002          2          ;:PATTERN 2
6998 022400 000400          400        ;:400 WORDS
6999
7000 022402 104417          TLOADRK     ;:LOAD RK REGS
7001 022404 104430          TWAT96     ;:WAIT FOR INTERRUPT
7002 022406 104002          ERROR 2    ;:TO SLOW/NOT COMPLETE ERROR
7003
7004 022410 104421          TCHKOP     ;:CHECK OPERATION FOR ANY ERRORS
7005 022412 104004          ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7006
7007 022414 004437 035674   JSR R4,LRLOAD ;:LOAD "L" REGS
7008 022420 000131          WRTCHK     ;:WRTCHK
7009 022422 177400          -400        ;:-400 WORDS
7010 022424 072414          OBUFF       ;:OBUFF IS BUFF ADDRESS
7011 022426 000          .BYTE 0     ;:SECTOR 0
7012 022427 000          .BYTE 0     ;:TRACK 0
7013 022430 000312          312        ;:CYLINDER 312
7014
7015 022432 104417          TLOADRK     ;:LOAD RK REGS
7016 022434 104424          TWAT32     ;:WAIT FOR INTERRUPT
7017 022436 104002          ERROR 2    ;:TO SLOW/NOT COMPLETE ERROR
7018
```

7019 022440 104421
7020 022442 104004
7021
7022
7023
7024
7025
7026
7027
7028
7029
7030
7031
7032
7033
7034
7035
7036
7037
7038
7039 022444 000004
7040 022446 012737 000012 001262
7041 022454 012700 000001
7042
7043 022460 104416
7044 022462 104003
7045 022464 004437 042260
7046 022470 000001
7047 022472 000400
7048
7049 022474 004437 035674
7050 022500 000123
7051 022502 177400
7052 022504 072414
7053 022506 000
7054 022507 000
7055 022510 000312
7056
7057 022512 104417
7058 022514 104430
7059 022516 104002
7060
7061 022520 104421
7062 022522 104004
7063
7064 022524 004437 035674
7065 022530 000131
7066 022532 177400
7067 022534 072414
7068 022536 000
7069 022537 000
7070 022540 000312
7071
7072 022542 104417
7073 022544 104424
7074 022546 104002

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS

*TEST 67 WRITE CHECK ERROR (PART 1)
*
* WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH ALL ZEROES.
* WRITE CHECK CYLINDER 312, TRACK 0, SECTOR 0 WITH SAME
* DATA EXCEPT WORD 110 HAS ONE OF THE FOLLOWING
* CONFIGURATIONS:
*
* 000001 000020 000400 010000
* 000002 000040 001000 020000
* 000004 000100 002000 040000
* 000010 000200 004000 100000
*
* MAKE SURE WRITE CHECK ERROR SET FOR EACH
* OF THE CONFIGURATIONS AND THAT THE BUS ADDRESS
* AND WORD COUNT IS CORRECT.
*

TST67: SCOPE
MOV #10,\$TIMES ;DO 10. ITERATIONS
MOV #BIT0,R0 ;SET LO ORDER BIT IN R0 FOR
;CAUSING WRITE CHECK ERROR
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
JSR R4,GENCOM ;GENERATE DATA, ALL 0'S
1
400
JSR R4,LRLOAD ;LOAD "L" REGS
WRDATA ;WRDATA
-400 ;-400 WORDS
OBUFF ;OBUFF IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312
TLOADRK ;LOAD RK REGS
TWAT96 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
JSR R4,LRLOAD ;LOAD "L" REGS
WRTCHK ;WRTCHK
-400 ;-400 WORDS
OBUFF ;OBUFF IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312
TLOADRK ;LOAD RK REGS
TWAT32 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

```
7075
7076 022550 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7077 022552 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7078
7079 022554 104415 SCOP1 ;LOCAL LOOP ON WRITE CHECK
7080 022556 012737 022564 001110 MOV #1$, $LPERR ;SET LOCAL LOOP
7081 022564 010037 072634 1$: MOV R0, OBUFF+220 ;CAUSE ERROR BIT IN BUFFER
7082 022570 104416 TSSINIT ;CLEAR SUBSYSTEM
7083 022572 104003 ERROR 3 ;BAD INIT ERROR
7084 022574 004437 035674 JSR R4, LRLOAD ;LOAD "L" REGS
7085 022600 000131 WRTCHK ;WRTCHK
7086 022602 177400 -400 ;-400 WORDS
7087 022604 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7088 022606 000 .BYTE 0 ;SECTOR 0
7089 022607 000 .BYTE 0 ;TRACK 0
7090 022610 000312 312 ;CYLINDER 312
7091
7092 022612 104417 TLOADRK ;LOAD RK REGS
7093 022614 104424 TWAT32 ;WAIT FOR INTERRUPT
7094 022616 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7095
7096 022620 104422 TCHKWE ;CHECK OPERATION WITH EXPECTED ERROR
7097 022622 000000 0
7098 022624 000004 WCKERR ;WRITE CHECK ERROR
7099 022626 000000 0
7100 022630 104004 ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
7101
7102 022632 104415 SCOP1 ;LOCAL LOOP ON ERROR TO 1$
7103
7104 :
7105 : NOTE: THE WORD COUNT AND BUS ADDRESS CAN BE EITHER OF THREE
7106 : VALUES AND BE CORRECT. THE DIFFERENCE IS CAUSED BY WHEN THE
7107 : WCE OCCURRED. IF IT OCCURRED ON THE FIRST WORD OF A DOUBLE
7108 : NPR CYCLE, WC AND BA WILL BE TWO PAST WHERE THE ERROR ACTUALLY
7109 : OCCURRED. IF WCE OCCURRED ON A SINGLE NPR CYCLE OR THE LAST
7110 : NPR CYCLE OF A DOUBLE CYCLE, WC AND BA CONTENTS ARE ONE PAST
7111 : THE ACTUAL WORD WHERE THE ERROR WAS. IN SOME CASES, WC AND BA
7112 : CAN BE THREE PAST WHERE THE ERROR WAS AND STILL BE ACCEPTABLE.
7113 :
7114 022634 023727 001544 072636 CMP T.BA, #OBUFF+222 ;CHECK BA HALT AT PROPER PLACE
7115 022642 001416 BEQ 2$ ;YES-SKIP
7116 022644 101040 BHI 6$ ;IF TO HI - SKIP
7117 022646 012737 054123 001450 MOV #EM11, EM10N ;"INCORRECT BA"
7118 022654 012737 072636 001202 MOV #OBUFF+222, $REG10 ;GOOD VALUE
7119 022662 013737 001544 001204 MOV T.BA, $REG11 ;BAD VALUE
7120 022670 012737 050613 061072 MOV #OPER41, DF010A ;"WRITE CHECK ABORTED WITH WCE"
7121 022676 104010 ERROR 10
7122
7123 022700 023727 001542 177511 2$: CMP T.WC, #-267 ;CHECK WORD COUNT AT CORRECT VALUE
7124 022706 001461 BEQ 3$ ;YES-SKIP
7125 022710 101037 BHI 7$ ;IF HIGHER SKIP
7126 022712 012737 054076 001450 MOV #EM10, EM10N ;"INCORRECT WC"
7127 022720 012737 050613 061072 MOV #OPER41, DF010A ;"WRITE CHECK ABORTED WITH WCE"
7128 022726 012737 177511 001202 MOV #-267, $REG10 ;GOOD VALUE
7129 022734 013737 001542 001204 MOV T.WC, $REG11 ;ERROR VALUE
7130 022742 104010 ERROR 10
```

```

7131 022744 000442 BR 3$ ;EXIT
7132
7133 022746 023727 001544 072642 6$: CMP T.BA,#OBUFF+226 ;TEST IF BA AT HI SIDE
7134 022754 101415 BLOS 7$ ;YES - SKIP
7135 022756 012737 054123 001450 MOV #EM11,EM10N ;SET MESSAGE
7136 022764 012737 072642 001202 MOV #OBUFF+226,$REG10 ;GOOD VALUE
7137 022772 013737 001544 001204 MOV T.BA,$REG11 ;ERROR VALUE
7138 023000 012737 050613 061072 MOV #OPER41,DF010A ;'WRITE CHECK ABORTED WITH WCE''
7139 023006 104010 ERROR 10
7140
7141 023010 023727 001542 177513 7$: CMP T.WC,#-265 ;TEST IF WORD COUNT AT HI SIDE
7142 023016 101415 BLOS 3$ ;YES - SKIP
7143 023020 012737 054076 001450 MOV #EM10,EM10N ;SET MESSAGE
7144 023026 012737 050613 061072 MOV #OPER41,DF010A ;'WC ABORTED WITH WCE''
7145 023034 012737 177513 001202 MOV #-265,$REG10 ;GOOD VALUE
7146 023042 013737 001542 001204 MOV T.WC,$REG11 ;ERROR VALUE
7147 023050 104010 ERROR 10
7148
7149 023052 104415 3$: SCOP1 ;LOCAL LOOP ON ERROR TO 1$
7150
7151 023054 032700 100000 BIT #BIT15,R0 ;BIT 15 SET?
7152 023060 001002 BNE 4$ ;YES-EXIT
7153 023062 006300 ASL R0 ;SHIFT ERROR BIT
7154 023064 000637 BR 1$ ;LOOP
7155
7156 023066 4$:
7157 ;*****
7158 ;*TEST 70 WRITE CHECK ERROR (PART 2)
7159 ;*
7160 ;* WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH 17777
7161 ;* IN ALL WORDS. WRITE CHECK CYLINDER 312, TRACK 0,
7162 ;* SECTOR 0 WITH THE SAME DATA EXCEPT WORD 120 HAS
7163 ;* ONE OF THE FOLLWOING CONFIGURATIONS:
7164 ;*
7165 ;* 177776 177757 177377 167777
7166 ;* 177775 177737 176777 157777
7167 ;* 177773 177677 175777 137777
7168 ;* 177767 177577 173777 077777
7169 ;*
7170 ;* MAKE SURE WRITE CHECK ERROR SET FOR EACH
7171 ;* OF THE CONFIGURATIONS AND THAT THE BUS ADDRESS
7172 ;* AND WORD COUNT IS CORRECT.
7173 ;*
7174 ;*****
7175 023066 000004 TST70: SCOPE
7176 023070 012737 000012 001262 MOV #10,$TIMES ;:DO 10. ITERATIONS
7177 023076 012700 177776 MOV #177776,R0 ;:LOAD R0 FOR CAUSING WRITE CHECK ERROR
7178
7179 023102 104416 TSSINIT ;:CLEAR SUBSYSTEM
7180 023104 104003 ERROR 3 ;:BAD INIT ERROR
7181 023106 004437 042260 JSR R4,GENCOM ;:GENERATE DATA
7182 023112 000007 7 ;:ALL 1'S
7183 023114 000400 400 ;:400 WORDS
7184
7185 023116 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
7186 023122 000123 WRDATA ;:WRDATA
  
```

```

7187 023124 177400      -400      ; -400 WORDS
7188 023126 072414      OBUFF      ; OBUFF IS BUFF ADDRESS
7189 023130      000      .BYTE 0    ; SECTOR 0
7190 023131      000      .BYTE 0    ; TRACK 0
7191 023132 000312      312       ; CYLINDER 312
7192
7193 023134 104417      TLOADRK    ; LOAD RK REGS
7194 023136 104430      TWAT96     ; WAIT FOR INTERRUPT
7195 023140 104002      ERROR 2    ; TO SLOW/NOT COMPLETE ERROR
7196
7197 023142 104421      TCHKOP     ; CHECK OPERATION FOR ANY ERRORS
7198 023144 104004      ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7199
7200 023146 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7201 023152 000131      WRTCHK     ;WRTCHK
7202 023154 177400      -400      ; -400 WORDS
7203 023156 072414      OBUFF      ; OBUFF IS BUFF ADDRESS
7204 023160      000      .BYTE 0    ; SECTOR 0
7205 023161      000      .BYTE 0    ; TRACK 0
7206 023162 000312      312       ; CYLINDER 312
7207
7208 023164 104417      TLOADRK    ; LOAD RK REGS
7209 023166 104424      TWAT32     ; WAIT FOR INTERRUPT
7210 023170 104002      ERROR 2    ; TO SLOW,NOT COMPLETE ERROR
7211
7212 023172 104421      TCHKOP     ; CHECK OPERATION FOR ANY ERRORS
7213 023174 104004      ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7214
7215 023176 104415      SCOPI      ;LOCAL LOOP TO START OF TEST
7216 023200 012737 023206 001110 MOV #1$, $LPERR ;SET LOCAL LOOP
7217
7218 023206 010037 072634 1$: MOV R0,OBUFF+220 ;PUT WORD IN OBUFF TO CAUSE WCE
7219 023212 104416      TSSINIT    ;CLEAR SUBSYSTEM
7220 023214 104003      ERROR 3    ;BAD INIT ERROR
7221
7222 023216 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7223 023222 000131      WRTCHK     ;WRTCHK
7224 023224 177400      -400      ; -400 WORDS
7225 023226 072414      OBUFF      ; OBUFF IS BUFF ADDRESS
7226 023230      000      .BYTE 0    ; SECTOR 0
7227 023231      000      .BYTE 0    ; TRACK 0
7228 023232 000312      312       ; CYLINDER 312
7229
7230 023234 104417      TLOADRK    ; LOAD RK REGS
7231 023236 104424      TWAT32     ; WAIT FOR INTERRUPT
7232 023240 104002      ERROR 2    ; TO SLOW/NOT COMPLETE ERROR
7233
7234 023242 104422      TCHKWE     ; CHECK OPERATION WITH EXPECTED ERROR
7235 023244 000000      0          ;
7236 023246 000004      WCKERR     ; WRITE CHECK ERROR
7237 023250 000000      0          ;
7238 023252 104004      ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
7239
7240 023254 104415      SCOPI      ;LOCAL LOOP TO 1$
7241
7242      ; NOTE: THE WORD COUNT AND BUS ADDRESS CAN BE EITHER OF THREE

```


7243 :
7244 :
7245 :
7246 :
7247 :
7248 :
7249 :
7250 :
7251 :
7252 023256 023727 001544 072636
7253 023264 001416
7254 023266 101040
7255 023270 012737 054123 001450
7256 023276 012737 072636 001202
7257 023304 013737 001544 001204

VALUES AND BE CORRECT. THE DIFFERENCE IS CAUSED BY WHEN THE
WCE OCCURRED. IF IT OCCURRED ON THE FIRST WORD OF A DOUBLE
NPR CYCLE, WC AND BA WILL BE TWO PAST WHERE THE ERROR ACTUALLY
OCCURRED. IF WCE OCCURRED ON A SINGLE NPR CYCLE OR THE LAST
NPR CYCLE OF A DOUBLE CYCLE, WC AND BA CONTENTS ARE ONE PAST
THE ACTUAL WORD WHERE THE ERROR WAS. IN SOME CASES, WC AND BA
CAN BE THREE PAST WHERE THE ERROR WAS AND STILL BE ACCEPTABLE.

CMP T.BA,#OBUFF+222 ;CHECK BA HALT AT PROPER PLACE
BEQ 2\$;YES-SKIP
BHI 6\$;IF TO HI - SKIP
MOV #EM11,EM10N ;"INCORRECT BA"
MOV #OBUFF+222,\$REG10 ;GOOD VALUE
MOV T.BA,\$REG11 ;BAD VALUE

```

7258 023312 012737 050613 061072      MOV    #OPER41,DF010A  ;'WRITE CHECK ABORTED WITH WCE''
7259 023320 104010      ERROR  10
7260
7261 023322 023727 001542 177511 2$:    CMP    T.WC,#-267      ;CHECK WORD COUNT AT CORRECT VALUE
7262 023330 001461      BEQ    3$              ;YES-SKIP
7263 023332 101037      BHI    7$              ;IF HIGHER - SKIP
7264 023334 012737 054076 001450      MOV    #EM10,EM10N    ;'INCORRECT WC''
7265 023342 012737 050613 061072      MOV    #OPER41,DF010A  ;'WRITE CHECK ABORTED WITH WCE''
7266 023350 012737 177511 001202      MOV    #-267,$REG10   ;GOOD VALUE
7267 023356 013737 001542 001204      MOV    T.WC,$REG11    ;ERROR VALUE
7268 023364 104010      ERROR  10
7269 023366 000442      BR     3$              ;EXIT
7270
7271 023370 023727 001544 072642 6$:    CMP    T.BA,#OBUFF+226 ;TEST IF BA AT HI SIDE
7272 023376 101415      BLOS  7$              ;YES - SKIP
7273 023400 012737 054123 001450      MOV    #EM11,EM10N    ;SET MESSAGE
7274 023406 012737 072642 001202      MOV    #OBUFF+226,$REG10 ;GOOD VALUE
7275 023414 013737 001544 001204      MOV    T.BA,$REG11    ;ERROR VALUE
7276 023422 012737 050613 061072      MOV    #OPER41,DF010A  ;'WRITE CHECK ABORTED WITH WCE''
7277 023430 104010      ERROR  10
7278
7279 023432 023727 001542 177513 7$:    CMP    T.WC,#-265      ;TEST IF WORD COUNT AT HI SIDE
7280 023440 101415      BLOS  3$              ;YES - SKIP
7281 023442 012737 054076 001450      MOV    #EM10,EM10N    ;SET MESSAGE
7282 023450 012737 050613 061072      MOV    #OPER41,DF010A  ;'WC ABORTED WITH WCE''
7283 023456 012737 177513 001202      MOV    #-265,$REG10   ;GOOD VALUE
7284 023464 013737 001542 001204      MOV    T.WC,$REG11    ;ERROR VALUE
7285 023472 104010      ERROR  10
7286
7287 023474 104415      3$:    SCOP1
7288
7289 023476 032700 100000      BIT    #BIT15,R0      ;BIT 15 SET? (ALL PATTERNS TESTED)
7290 023502 001002      BNE   4$              ;YES-EXIT
7291 023504 006300      ASL   R0              ;SHIFT FOR NEXT TEST
7292 023506 000637      BR    1$              ;LOOP
7293
7294 023510      4$:
7295      ;*****
7296      ;*TEST 71      WRITE CHECK OF PARTIAL SECTOR
7297      ;*
7298      ;*      WRITE CYLINDER 312, TRACK 0, SECTOR WITH A KNOWN
7299      ;*      CONFIGURATIONS.  ISSUE A WRITE CHECK COMMAND OF
7300      ;*      110 WORDS MAKING SURE THE 111TH WORD IS
7301      ;*      DIFFERENT THAN DATA ON DISK.  MAKE SURE
7302      ;*      WRITE CHECK ERROR DOES NOT SET.
7303      ;*
7304      ;*****
7305 023510 000004      TST71:  SCOPE
7306 023512 012737 000012 001262      MOV    #10,,$TIMES   ;;DO 10. ITERATIONS
7307 023520 104416      TSSINIT
7308 023522 104003      ERROR  3              ;CLEAR SUBSYSTEM
7309      ;BAD INIT ERROR
7310 023524 004437 042260      JSR   R4,GENCOM      ;GENERATE DATA
7311 023530 000007      7
7312 023532 000400      400                  ;ALL 1'S
7313      ;400 WORDS
  
```

7314 023534 004437 035674
7315 023540 000123
7316 023542 177400
7317 023544 072414
7318 023546 000
7319 023547 000
7320 023550 000312
7321
7322 023552 104417
7323 023554 104430
7324 023556 104002
7325
7326 023560 104421
7327 023562 104004
7328
7329 023564 005037 072636
7330
7331 023570 004437 035674
7332 023574 000131
7333 023576 177670
7334 023600 072414
7335 023602 000
7336 023603 000
7337 023604 000312
7338
7339 023606 104417
7340 023610 104424
7341 023612 104002
7342
7343 023614 104421
7344 023616 104004
7345
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367 023620 000004
7368 023622 012737 000002 001262
7369 023630 032737 000400 001664

```

JSR R4,LRLOAD ;LOAD "L" REGS
WRDATA ;WRDATA
-400 ;-400 WORDS
OBUFF ;OBUFF IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312

TLOADRK ;LOAD RK REGS
TWAT96 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS

CLR OBUFF+222

JSR R4,LRLOAD ;LOAD "L" REGS
WRTCHK ;WRTCHK
-110 ;-110 WORDS
OBUFF ;OBUFF IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312

TLOADRK ;LOAD RK REGS
TWAT32 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS

```

.SBTTL **MAXIMUM DATA TRANSFER AND CONTROLLER TIME OUT

```

*****
*TEST 72 MAXIMUM DATA TRANSFER (PART 1)
*****

```

```

*
* IN THE FIRST PASS OF THE PROGRAM, THE HEADERS OF
* THE FIRST 4 CYLINDERS ARE WRITTEN. THIS IS DONE TO
* INSURE THE FORMAT IS CORRECT.

```

```

*
* ZERO OUT THE FIRST 256 SECTORS OF THE DISK WITH
* ONE SECTOR WRITES. ISSUE A SEEK TO CYLINDER 0, TRACK 0.
* ISSUE A WRITE DATA OF MAXIMUM DATA TRANSFER 200000 WORDS
* TO CYLINDER 0, TRACK 0, SECTOR 0. MAKE SURE CONTROLLER
* TIME OUT IS NOT SET. CHECK CYLINDER ADDRESS,
* DISK ADDRESS, BUS ADDRESS AND WORD COUNT. READ
* EACH SECTOR TO MAKE SURE IT WAS WRITTEN CORRECTLY.

```

```

*
* NOTE: THIS TEST IS EXECUTED ONLY IF NO BAD SECTORS ARE PRESENT
* IN THE FIRST 256 SECTORS ON THE PACK.

```

```

*****
TST72: SCOPE
MOV #2,$TIMES ;DO 2 ITERATIONS
BIT #FPFMT,OPTFLG ;TEST IF FIRST PASS SWITCH SET

```

7370	023636	001043			BNE	24\$;YES - SKIP FORMAT
7371								
7372	023640	105037	023663		CLRB	21\$;CLEAR ADDRESS POINTERS
7373	023644	005037	023664		CLR	22\$		
7374								
7375	023650	004437	035674	20\$:	JSR	R4,LRLOAD		;LOAD "L" REGISTERS
7376	023654	000127			WRHEAD			;WRITE HEADER
7377	023656	177676			-102			;102 WORDS
7378	023660	072414			OBUFF			;OBUFF IS BUFF ADDRESS
7379	023662	000			.BYTE	0		;SECTOR 0
7380	023663	000		21\$:	.BYTE	0		;TRACK ADDRESS (VARIABLE)
7381	023664	000000		22\$:	0			;CYLINDER 0 (VARIABLE)
7382								
7383	023666	004437	042260		JSR	R4,GENCOM		;GO GENERATE HEADERS
7384	023672	001200			1200			;WITH BAD SECTOR ERRORS
7385								
7386	023674	104417			TLOADRK			;LOAD RK REGS
7387	023676	104431			TWAT112			;WAIT FOR INTERRUPT
7388	023700	104002			ERROR	2		;TO SLOW/NOT COMPLETE ERROR
7389								
7390	023702	104421			TCHKOP			;CHECK OPERATION FOR ANY ERRORS
7391	023704	104004			ERROR	4 ;OR 5, 6, 7		;REPORT ALL ERRORS
7392								
7393	023706	123727	023663	000002	CMPB	21\$,#2		;TEST IF LAST TRACK
7394	023714	001403			BEQ	23\$;YES - SKIP
7395	023716	105237	023663		INCB	21\$;ELSE BUMP TRACK
7396	023722	000752			BR	20\$;LOOP
7397								
7398	023724	105037	023663	23\$:	CLRB	21\$;CLEAR TRACK POINTER
7399	023730	023727	023664	000003	CMP	22\$,#3		;TEST IF LAST CYLINDER WRITTEN
7400	023736	001403			BEQ	24\$;YES - SKIP
7401	023740	005237	023664		INC	22\$;ELSE BUMP CYLINDER
7402	023744	000741			BR	20\$;LOOP
7403								
7404	023746	013737	023760	001110	MOV	1\$, \$LPERR		;SET LOCAL LOOP ON ERROR
7405	023754	012703	000400		MOV	#400,R3		;SET COUNT FOR SECTOR CLEARING
7406	023760			1\$:				
7407	023760	104416			TSSINIT			;CLEAR SUBSYSTEM
7408	023762	104003			ERROR	3		;BAD INIT ERROR
7409	023764	004437	035674		JSR	R4,LRLOAD		;LOAD "L" REGS
7410	023770	000123			WRDATA			;WRDATA
7411	023772	177400			-400			; -400 WORDS
7412	023774	072414			OBUFF			;OBUFF IS BUFF ADDRESS
7413	023776	000			.BYTE	0		;SECTOR 0
7414	023777	000			.BYTE	0		;TRACK 0
7415	024000	000000			0			;CYLINDER 0
7416								
7417	024002	104417			TLOADRK			;LOAD RK REGS
7418	024004	104434			TWAT159			;WAIT FOR INTERRUPT
7419	024006	104002			ERROR	2		;TO SLOW/NOT COMPLETE ERROR
7420								
7421	024010	104421			TCHKOP			;CHECK OPERATION FOR ANY ERRORS
7422	024012	104004			ERROR	4 ;OR 5, 6, 7, 10		;REPORT ALL ERRORS
7423								
7424	024014	104415			SCOP1			;LOCAL LOOP ON ERROR TO 1\$
7425	024016	005303			DEC	R3		;DECREMENT COUNT

```
7426 024020 012762 072414 000004 2$: MOV #OBUFF,RKBA(R2) ;SET BA
7427 024026 012762 177400 000002 MOV #-400,RKWC(R2) ;AND WC AGAIN
7428 024034 005037 001662 CLR INTSET ;CLEAR INTERRUPT FLAG
7429 024040 013762 001626 000010 MOV DRVNUM,RKCS2(R2) ;SET DRIVE NUMBER
7430 024046 012737 000123 036124 MOV #WRDATA,HOLD
7431 024054 053737 001720 036124 BIS DTYPE,HOLD
7432 024062 013762 036124 000000 MOV HOLD,RKCS1(R2) ;DO WRITE DATA
7433
7434 024070 104425 TWAT48 ;WAIT FOR INTERRUPT
7435 024072 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7436
7437 024074 032762 000200 000014 BIT #BSE,RKER(R2) ;BAD SECTOR ERROR?
7438 024102 001415 BEQ 3$ ;NO-SKIP
7439 024104 032737 000200 001664 BIT #BSERPT,OPTFLG ;TEST IF BSE TO MANY HAS BEEN REPORTED
7440 024112 001007 BNE 5$ ;YES - SKIP
7441 024114 052737 000200 001664 BIS #BSERPT,OPTFLG ;SET FLAG
7442 024122 012737 052547 001360 MOV #OPR017,EM1N ;SET MESSAGE
7443 024130 104001 ERROR 1 ;"FIRST 256 SECTOR NOT BSE FREE"
7444 024132 000137 024466 5$: JMP 14$ ;GO TO EXIT
7445
7446 024136 3$: CHECK FOR ANY OTHER ERRORS
7447 024136 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7448 024140 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7449 024142 104415 SCOP1 ;LOCAL LOOP TO 1$ (RESTART SECTOR CLEAR)
7450
7451 024144 005303 DEC R3 ;DECREMENT COUNT
7452 024146 001324 BNE 2$ ;LOOP IF NOT ZERO
7453
7454 024150 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7455 024154 000117 SEEK ;SEEK
7456 024156 000000 0 ;0 WORDS
7457 024160 000000 0 ;0 IS BUFF ADDRESS
7458 024162 000 .BYTE 0 ;SECTOR 0
7459 024163 000 .BYTE 0 ;TRACK 0
7460 024164 000000 0 ;CYLINDER 0
7461
7462 024166 104417 TLOADRK ;LOAD RK REGS
7463 024170 104423 TWAT16 ;WAIT FOR INTERRUPT
7464 024172 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7465 024174 005037 001662 CLR INTSET ;CLEAR FIRST INTERRUPT
7466 024200 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7467 024202 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7468 024204 104427 TWAT80 ;WAIT FOR SECOND INTERRUPT
7469 024206 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7470 024210 004437 042260 JSR R4,GENCOM ;GENERATE DATA
7471 024214 004006 4006 ;PATTERN 6, 1ST WORD REPEATED
7472 024216 000400 400 ;400 WORDS
7473
7474 024220 4$:
7475 024220 104416 TSSINIT ;CLEAR SUBSYSTEM
7476 024222 104003 ERROR 3 ;BAD INIT ERROR
7477 024224 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7478 024230 000123 WRDATA ;WRDATA
7479 024232 000000 0 ;0 WORDS
7480 024234 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7481 024236 000 .BYTE 0 ;SECTOR 0
```

7482	024237	000		.BYTE	0		:TRACK 0
7483	024240	000000			0		:CYLINDER 0
7484							
7485	024242	052737	000020	BIS	#BAI,L.CS2		
7486							
7487	024250	104417		TLOADRK			:LOAD RK REGS
7488	024252	104437		TWATSS			:WAIT FOR SECOND INTERRUPT
7489	024254	104002		ERROR	2		:ELSE REPORT TO SLOW/NOT COMPLETE ERROR
7490							
7491	024256					7\$:	
7492	024256	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
7493	024260	104004		ERROR	4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS
7494							
7495	024262	104415		SCOP1			:INTERNAL LOOP ON ERROR TO 4\$
7496	024264	012703	000400	MOV	#400,R3		:SET COUNTER FOR READ-COMPARE LOOP
7497	024270	005037	024332	CLR	10\$:CLEAR SECTOR AND TRACK VALUES
7498	024274	005037	024334	CLR	12\$:CLEAR CYL VALUE
7499	024300	013737	024314	MOV	8\$,\$LPERR		:SET LOCAL LOOP ON ERROR
7500	024306	042737	000020	BIC	#BAI,L.CS2		:CLEAR BAI
7501							
7502	024314					8\$:	
7503	024314	104416		TSSINIT			:CLEAR SUBSYSTEM
7504	024316	104003		ERROR	3		:BAD INIT ERROR
7505	024320	004437	035674	JSR	R4,LRLOAD		:LOAD RK REGS
7506	024324	000121		RDDATA			:READ DATA
7507	024326	177400		-400			:400 WORDS
7508	024330	070414		IBUFF			:INTO IBUFF
7509	024332	000		.BYTE	0	10\$:	:SECTOR (VARIABLE)
7510	024333	000		.BYTE	0	11\$:	:TRACK (VARIABLE)
7511	024334	000000			0	12\$:	:CYL (VARIABLE)
7512							
7513	024336	104417		TLOADRK			:LOAD RK REGS
7514	024340	104425		TWAT48			:WAIT FOR INTERRUPT
7515	024342	104002		ERROR	2		:TO SLOW/NOT COMPLETE ERROR
7516							
7517	024344	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
7518	024346	104004		ERROR	4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS
7519							
7520	024350	104415		SCOP1			:LOCAL LOOP ON ERROR TO 8\$
7521							
7522	024352	004437	042260	JSR	R4,GENCOM		:COMPARE DATA
7523	024356	100000		100000			
7524	024360	000400		400			:400 WORDS
7525	024362	000413		BR	13\$:NO MISCOMPARE-EXIT LOOP
7526	024364	104015		ERROR	15		:REPORT FIRST ERROR
7527							
7528	024366	013700	001634	MOV	ERRLMT,R0		:GET ERROR LIMIT
7529	024372	005300		DEC	R0	64\$:	:DECREMENT COUNT
7530	024374	001406		BEQ	65\$:IF ZERO - EXIT
7531	024376	004437	042260	JSR	R4,GENCOM		:CONTINUE DATA COMPARE
7532	024402	040000		40000			
7533	024404	000402		BR	65\$:NO MORE ERRORS - EXIT
7534	024406	104016		ERROR	16		:REPORT NEXT ERROR
7535	024410	000770		BR	64\$:LOOP
7536	024412					65\$:	
7537							

7538	024412	104415			13\$:	SCOPI			:LOCAL LOOP TO 8\$
7539									
7540	024414	005303				DEC	R3		:DEC READ LOOP COUNT
7541	024416	001423				BEQ	14\$:IF ZERO-EXIT
7542									
7543	024420	105237	024332			INCB	10\$:BUMP SECTOR
7544	024424	123727	024332	000026		CMPB	10\$,#26		:FINISHED WITH TRACK?
7545	024432	001332				BNE	9\$:NO-LOOP
7546	024434	105037	024332			CLRB	10\$:CLEAR SECTOR
7547	024440	105237	024333			INCB	11\$:BUMP TRACK
7548	024444	123727	024333	000003		CMPB	11\$,#3		:FINISHED WITH CYLINDER?
7549	024452	001322				BNE	9\$:NO-LOOP
7550	024454	105037	024333			CLRB	11\$:CLEAR TRACK
7551	024460	005237	024334			INC	12\$:BUMP CYL.
7552	024464	000715				BR	9\$:LOOP
7553									

7554 024466

14\$:

:TEST 73 MAXIMUM DATA TRANSFER (PART 2)

:*
:ZERO OUT FIRST 256 SECTORS OF THE DISK WITH
:200000 WORD WRITE. SEEK TO CYLINDER 632.
:ISSUE A WRITE OF MAXIMUM DATA TRANSFER
:200000 WORD WRITE. MAKE SURE CONTROLLER TIME
:OUT IS NOT SET. CHECK CYLINDER ADDRESS
:DISK ADDRESS, BUS ADDRESS AND WORD COUNT.
:SEEK TO CYLINDER 632. ISSUE A WRITE CHECK
:OF 200000 WORDS. MAKE SURE NO ERROR SETS.

:*
:NOTE: THIS TEST IS EXECUTED ONLY IF NO BAD SECTORS ARE PRESENT
:IN THE FIRST 256 SECTORS ON THE PACK.
:CYLINDER 1456 IS USED FOR RK07.

7571									
7572	024466	000004			TST73:	SCOPE			
7573	024470	012737	000002	001262		MOV	#2,\$TIMES		::DO 2 ITERATIONS
7574	024476	104416				TSSINIT			:CLEAR SUBSYSTEM
7575	024500	104003				ERROR	3		:BAD INIT ERROR
7576	024502	012700	000620			MOV	#400.,R0		:SET COUNT FOR INTERRUPT WAIT
7577	024506	005037	072414			CLR	OBUFF		
7578									
7579	024512	004437	035674			JSR	R4,LRLOAD		:LOAD 'L' REGS
7580	024516	000123				WRDATA			:WRDATA
7581	024520	000000				0			:0 WORDS
7582	024522	072414				OBUFF			:OBUFF IS BUFF ADDRESS
7583	024524	000				.BYTE	0		:SECTOR 0
7584	024525	000				.BYTE	0		:TRACK 0
7585	024526	000000				0			:CYLINDER 0
7586	024530	052737	000020	001610		BIS	#BAI,L.CS2		:SET BAI
7587	024536	104417				TLOADRK			:LOAD RK REGS
7588	024540	104434			1\$:	TWAT159			:WAIT FOR INTERRUPT
7589	024542	000401				BR	2\$:NO INTERRUPT-SKIP
7590	024544	000403				BR	3\$:INTERRUPT-SKIP
7591									
7592	024546	005300			2\$:	DEC	R0		:DEC WAIT COUNTER
7593	024550	001373				BNE	1\$:NO ZERO-LOOP

7594	024552	104002				ERROR	2		:TO SLOW/NOT COMPLETE ERROR
7595									
7596	024554	032762	000200	000014	3\$:	BIT	#BSE,RKER(R2)		:DID BSE SET
7597	024562	001415				BEQ	4\$:NO-SKIP
7598									
7599	024564	032737	000200	001664		BIT	#BSERPT,OPTFLG		:TEST IF TO MANY BAD SECTORS REPORTED
7600	024572	001007				BNE	12\$:YES - SKIP
7601	024574	052737	000200	001664		BIS	#BSERPT,OPTFLG		:SET FLAG
7602	024602	012737	052547	001360		MOV	#OPR017,EM1N		:SET MESSAGE
7603	024610	104001				ERROR	1		:FIRST 256 SECTORS NOT BSE FREE"
7604	024612	000137	025160		12\$:	JMP	11\$:EXIT
7605									
7606	024616				4\$:				
7607	024616	104421				TCHKOP			:CHECK OPERATION FOR ANY ERRORS //
7608	024620	104004				ERROR	4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS
7609	024622	013737	036120	024652		MOV	LSTCYL,13\$		
7610	024630	013737	036120	025026		MOV	LSTCYL,14\$		
7611									
7612	024636	004437	035674			JSR	R4,LRLOAD		:LOAD 'L' REGS
7613	024642	000117				SEEK			
7614	024644	000000				0			:0 WORDS
7615	024646	000000				0			:0 IS BUFF ADDR
7616	024650	000				.BYTE	0		:SEC 0
7617	024651	000				.BYTE	0		:TRK 0
7618	024652	000000			13\$:	0			:CYL 632/1456
7619									
7620	024654	104417				TLOADRK			:LOAD RK REGS
7621	024656	104423				TWAT16			:WAIT FOR INTERRUPT
7622	024660	104002				ERROR	2		:TO SLOW/NOT COMPLETE ERROR
7623	024662	005037	001662			CLR	INTSET		:CLEAR INTERRUPT FLAG
7624									
7625	024666	104421				TCHKOP			:CHECK OPERATION FOR ANY ERRORS
7626	024670	104004				ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS
7627									
7628	024672	104427				TWAT80			:WAIT FOR 2ND INTERRUPT
7629	024674	104002				ERROR	2		
7630									
7631	024676	104421				TCHKOP			:CHECK OPERATION FOR ANY ERRORS
7632	024700	104004				ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS
7633									
7634	024702	004437	035674			JSR	R4,LRLOAD		:LOAD 'L' REGS
7635	024706	000105				CLEAR			:CLEAR
7636	024710	000000				0			:0 WORDS
7637	024712	000000				0			:0 IS BUFF ADDRESS
7638	024714	000				.BYTE	0		:SECTOR 0
7639	024715	000				.BYTE	0		:TRACK 0
7640	024716	000000				0			:CYLINDER 0
7641									
7642	024720	104417				TLOADRK			:LOAD RK REGS
7643	024722	104423				TWAT16			:WAIT FOR INTERRUPT
7644	024724	104002				ERROR	2		:TO SLOW/NOT COMPLETE ERROR
7645									
7646	024726	104421				TCHKOP			:CHECK OPERATION FOR ANY ERRORS
7647	024730	104004				ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS
7648									
7649	024732	004437	035674			JSR	R4,LRLOAD		:LOAD 'L' REGS

7650	024736	000123		WRDATA		:WRDATA
7651	024740	000000		0		:0 WORDS
7652	024742	072414		OBUFF		:OBUFF IS BUFF ADDRESS
7653	024744	000		.BYTE 0		:SECTOR 0
7654	024745	000		.BYTE 0		:TRACK 0
7655	024746	000000		0		:CYLINDER 0
7656						
7657	024750	012737	135143 072414	MOV #135143,OBUFF		:SET WORD FOR OUTPUT
7658	024756	012700	000621	MOV #401.,R0		:SET COUNT FOR INTERRUPT WAIT
7659	024762	052737	000020 001610	BIS #BAI,L.CS2		:SET BUS ADDRESS INC INHIBIT
7660						
7661	024770	104417		TLOADRK		:LOAD RK REGS
7662	024772	104434		5\$: TWAT159		:WAIT FOR INTERRUPT
7663	024774	000401		BR 6\$:NO INTERRUPT-BRANCH
7664	024776	000403		BR 7\$:INTERRUPT-BRANCH
7665						
7666	025000	005300		6\$: DEC R0		:DEC WAIT COUNT
7667	025002	001373		BNE 5\$:LOOP IF NOT ZERO
7668	025004	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
7669						
7670	025006			7\$:		
7671	025006	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7672	025010	104004		ERROR 4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS
7673						
7674	025012	004437	035674	JSR R4,LRLOAD		:LOAD "L" REGS
7675	025016	000117		SEEK		
7676	025020	000000		0		:0 WORDS
7677	025022	000000		0		:0 IS BUFF ADDR
7678	025024	000		.BYTE 0		:SEC 0
7679	025025	000		.BYTE 0		:TRK 0
7680	025026	000000		0		:CYL 632/1456
7681	025030	104417		TLOADRK		:LOAD RK REGS
7682	025032	104423		TWAT16		:WAIT FOR INTERRUPT
7683	025034	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
7684	025036	005037	001662	CLR INTSET		:CLEAR INTERRUPT FLAG
7685						
7686	025042	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7687	025044	104004		ERROR 4 ;OR 5, 6, 7		:REPORT ALL ERRORS
7688						
7689	025046	104427		TWAT80		:WAIT FOR SECOND INIT
7690	025050	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
7691	025052	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7692	025054	104004		ERROR 4 ;OR 5, 6, 7		:REPORT ALL ERRORS
7693						
7694	025056	004437	035674	JSR R4,LRLOAD		:LOAD "L" REGS
7695	025062	000105		CLEAR		:CLEAR
7696	025064	000000		0		:0 WORDS
7697	025066	000000		0		:0 IS BUFF ADDRESS
7698	025070	000		.BYTE 0		:SECTOR 0
7699	025071	000		.BYTE 0		:TRACK 0
7700	025072	000000		0		:CYLINDER 0
7701						
7702	025074	104417		TLOADRK		:LOAD RK REGS
7703	025076	104423		TWAT16		:WAIT FOR INTERRUPT
7704	025100	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
7705						

```
7706 025102 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7707 025104 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7708
7709 025106 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7710 025112 000131 WRTCHK ;WRTCHK
7711 025114 000000 0 ;0 WORDS
7712 025116 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7713 025120 000 .BYTE 0 ;SECTOR 0
7714 025121 000 .BYTE 0 ;TRACK 0
7715 025122 000000 0 ;CYLINDER 0
7716 025124 052737 000020 001610 BIS #BAI,L,CS2 ;SET BAI FLAG
7717 025132 012700 000621 MOV #401.,R0 ;SET WAIT COUNT
7718
7719 025136 104417 TLOADRK ;LOAD RK REGS
7720 025140 104434 8$: TWAT159 ;WAIT FOR INTERRUPT
7721 025142 000401 BR 9$ ;NO INTERRUPT-SKIP
7722 025144 000403 BR 10$ ;INTERRUPT-SKIP
7723
7724 025146 005300 9$: DEC R0 ;DEC WAIT COUNT
7725 025150 001373 BNE 8$ ;NOT ZERO-LOOP
7726 025152 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7727
7728 025154 10$:
7729 025154 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7730 025156 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7731
7732 025160 11$:
7733 *****
7734 :*TEST 74 CONTROLLER TIME OUT
7735 :*
7736 :* SEEK TO CYLINDER 632. ISSUE A RECALIBRATE AND DO NOT
7737 :* WAIT FOR SECOND INTERRUPT. NOW ISSUE A READ HEADER
7738 :* OF CYLINDER 0, TRACK 0. MAKE SURE CONTROLLER TIME
7739 :* OUT SETS.
7740 :* CYLINDER 1456 IS USED FOR THE RK07.
7741 :*
7742 :* *****
7743 TST74: SCOPE
7744 025160 000004 MOV #5.,$TIMES ;;DO 5. ITERATIONS
7745 025162 012737 000005 001262 TSSINIT ;CLEAR SUBSYSTEM
7746 025170 104416 ERROR 3 ;BAD INIT ERROR
7747 025174 013737 036120 025216 MOV LSTCYL,1$
7748
7749 025202 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7750 025206 000117 SEEK
7751 025210 000000 0 ;0 WORDS
7752 025212 000000 0 ;0 IS BUFF ADDR
7753 025214 000 .BYTE 0 ;SEC 0
7754 025215 000 .BYTE 0 ;TRK 0
7755 025216 000000 1$: 0 ;CYL 632/1456
7756
7757 025220 104417 TLOADRK ;LOAD RK REGS
7758 025222 104423 TWAT16 ;WAIT FOR INTERRUPT
7759 025224 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7760
7761 025226 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
```

7762	025230	104004		ERROR	4 ;OR 5, 6, 7	;REPORT ALL ERRORS
7763						
7764	025232	005037	001662	CLR	INTSET	;CLEAR INTERRUPT FLAG
7765	025236	104427		TWAT80		;WAIT FOR SECOND INTERRUPT
7766	025240	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
7767	025242	104421		TCHKOP		;CHECK OPERATION FOR ANY ERRORS
7768	025244	104004		ERROR	4 ;OR 5, 6, 7	;REPORT ALL ERRORS
7769						
7770	025246	004437	035674	JSR	R4,LRLOAD	;LOAD 'L' REGS
7771	025252	000105		CLEAR		;CLEAR
7772	025254	000000		0		;0 WORDS
7773	025256	000000		0		;0 IS BUFF ADDRESS
7774	025260	000		.BYTE	0	;SECTOR 0
7775	025261	000		.BYTE	0	;TRACK 0
7776	025262	000000		0		;CYLINDER 0
7777						
7778	025264	104417		TLOADRK		;LOAD RK REGS
7779	025266	104423		TWAT16		;WAIT FOR INTERRUPT
7780	025270	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
7781						
7782	025272	104421		TCHKOP		;CHECK OPERATION FOR ANY ERRORS
7783	025274	104004		ERROR	4 ;OR 5, 6, 7	;REPORT ALL ERRORS
7784						
7785	025276	004437	035674	JSR	R4,LRLOAD	;LOAD 'L' REGS
7786	025302	000113		RECAL		;RECAL
7787	025304	000000		0		;0 WORDS
7788	025306	000000		0		;0 IS BUFF ADDRESS
7789	025310	000		.BYTE	0	;SECTOR 0
7790	025311	000		.BYTE	0	;TRACK 0
7791	025312	000000		0		;CYLINDER 0
7792						
7793	025314	104417		TLOADRK		;LOAD RK REGS
7794	025316	104423		TWAT16		;WAIT FOR INTERRUPT
7795	025320	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
7796						
7797	025322	004437	035674	JSR	R4,LRLOAD	;LOAD 'L' REGS
7798	025326	000125		RDHEAD		;RDHEAD
7799	025330	000000		0		;0 WORDS
7800	025332	000000		0		;0 IS BUFF ADDRESS
7801	025334	000		.BYTE	0	;SECTOR 0
7802	025335	000		.BYTE	0	;TRACK 0
7803	025336	000000		0		;CYLINDER 0
7804						
7805	025340	104417		TLOADRK		;LOAD RK REGS
7806	025342	104436		TWAT2S		;WAIT FOR INTERRUPT
7807	025344	104002		ERROR	2	;TO SLOW/NOT COMPLETE ERROR
7808						
7809	025346	104422		TCHKWE		;CHECK OPERATION WITH EXPECTED ERROR
7810	025350	000000		0		
7811	025352	000000		0		
7812	025354	000002		CTOERR		;CONTROLLER TIME OUT
7813	025356	104004		ERROR	4; OR 5,6,7	;REPORT ANY DISCREPANCIES
7814	025360	104416		TSSINIT		;CLEAR SUBSYSTEM
7815	025362	104003		ERROR	3	;BAD INIT ERROR
7816	025364	005037	001662	CLR	INTSET	;CLEAR INT FLAG
7817	025370	012762	000100 000000	MOV	#IE,RKCS1(R2)	;SET INT ENABLE

7818 025376 104437
7819 025400 104002
7820
7821
7822
7823
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835 025402 000004
7836 025404 012737 000003 001262
7837
7838 025412 104416
7839 025414 104003
7840
7841 025416 004437 035674
7842 025422 000113
7843 025424 000000
7844 025426 000000
7845 025430 000
7846 025431 000
7847 025432 000000
7848
7849 025434 104417
7850 025436 104423
7851 025440 104002
7852
7853 025442 005037 001662
7854 025446 104437
7855 025450 104002
7856
7857 025452 104421
7858 025454 104004
7859
7860 025456 004437 035674
7861 025462 000117
7862 025464 000000
7863 025466 000000
7864 025470 000
7865 025471 000
7866 025472 000002
7867 025474 012737 000020 001616
7868 025502 104417
7869 025504 104423
7870 025506 104002
7871
7872 025510 104416
7873 025512 104003

TWAT8S ;WAIT FOR SECOND INT
ERROR 2

.SBTTL **ERRORS DURING DATA TRANSFER

*TEST 75 LIMIT DETECT ON DATA TRANSFER
*
* ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. ISSUE
* A SEEK TO CYLINDER 2 WITH BAD PARITY. ISSUE A DRIVE
* CLEAR. ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 1,
* TRACK 0, HEAD 0. SEEK INCOMPLETE BECAUSE OF OUTER
* LIMIT SHOULD BE THE ONLY ERROR SET.

TST75: SCOPE
MOV #3,\$TIMES ;:DO 3. ITERATIONS
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
JSR R4,LRLOAD ;LOAD 'L' REGS
RECAL ;RECAL
0 ;0 WORDS
0 ;0 IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
0 ;CYLINDER 0
TLOADRK ;LOAD RK REGS
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
CLR INTSET ;CLEAR INTERRUPT FLAG
TWAT8S ;WAIT FOR SECOND INTERRUPT
ERROR 2
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
JSR R4,LRLOAD ;LOAD 'L' REGS
SEEK ;SEEK
0 ;0 WORDS
0 ;0 IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
2 ;CYLINDER 2
MOV #PAT,L.MR1 ;SET EVEN PARITY BIT
TLOADRK ;LOAD RK REGS
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TSSINIT ;CLEAR SUBSYSTEM
ERPOR 3 ;BAD INIT ERROR

7874									
7875	025514	004437	035674		JSR	R4,LRLOAD		:LOAD "L" REGS	
7876	025520	000123			WRDATA			:WRDATA	
7877	025522	177400			-400			: -400 WORDS	
7878	025524	072414			OBUFF			:OBUFF IS BUFF ADDRESS	
7879	025526	000			.BYTE	0		:SECTOR 0	
7880	025527	000			.BYTE	0		:TRACK 0	
7881	025530	000001			i			:CYLINDER 1	
7882									
7883	025532	104417			TLOADRK			:LOAD RK REGS	
7884	025534	104423			TWAT16			:WAIT FOR INTERRUPT	
7885	025536	104002			ERROR	2		:TO SLOW/NOT COMPLETE ERROR	
7886									
7887	025540	104422			TCHKWE			:CHECK OPERATION WITH ERROR	
7888	025542	000002			SKIERR			:SEEK INCOMPLETE	
7889	025544	000000			0				
7890	025546	000000			0				
7891	025550	104004			ERROR	4 ;OR 5,6,OR7		:REPORT ALL DISCREPANCIES	
7892									
7893	025552	104416			TSSINIT			:CLEAR SUBSYSTEM	
7894	025554	104003			ERROR	3		:BAD INIT ERROR	
7895									
7896									
7897	025556			3\$:					
7898	025556	004437	035674		JSR	R4,LRLOAD		:LOAD "L" REGS	
7899	025562	000101			SELDRV			:SELDRV	
7900	025564	000000			0			:0 WORDS	
7901	025566	000000			0			:0 IS BUFF ADDRESS	
7902	025570	000			.BYTE	0		:SECTOR 0	
7903	025571	000			.BYTE	0		:TRACK 0	
7904	025572	000000			0			:CYLINDER 0	
7905									
7906	025574	012737	000001	001616	MOV	#1,L.MR1		:SET TO GET STATUS PAIR 1	
7907	025602	104417			TLOADRK			:LOAD RK REGS	
7908	025604	104423			TWAT16			:WAIT FOR INTERRUPT	
7909	025606	104002			ERROR	2		:TO SLOW/NOT COMPLETE ERROR	
7910									
7911	025610	104421			TCHKOP			:CHECK OPERATION FOR ANY ERRORS	
7912	025612	104004			ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS	
7913									
7914	025614	032737	000040	001574	BIT	#S.HDHM,T.MR2		:TEST IF HEADS HOME	
7915	025622	001755			BEQ	3\$:NO - GET STATUS AGAIN	
7916									
7917	025624	104416			TSSINIT			:CLEAR SUBSYSTEM	
7918	025626	104003			ERROR	3		:BAD INIT ERROR	
7919									
7920	025630	012762	000100	000000	MOV	#IE,RKCS1(R2)		:SET IE	
7921									
7922	025636	005037	001662		CLR	INTSET		:CLEAR INT FLAG	
7923	025642	104437			TWAT8S			:WAIT FOR SECOND INTERRUPT	
7924	025644	000401			BR	1\$			
7925	025646	000404			BR	2\$			
7926									
7927	025650	012737	060404	001372	1\$:	MOV	#DH016,DH2N	: "SUBSYSTEM CLEAR TO RESET LIMIT ERROR	
7928								: ALLOWING HEADS TO RELOAD"	
7929	025656	104002			ERROR	2			

7930
7931 025660 2\$:
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941 025660 000004
7942 025662 012737 000012 001262
7943 025670 104416
7944 025672 104003
7945
7946 025674 004437 035674
7947 025700 000121
7948 025702 177400
7949 025704 070414
7950 025706 000
7951 025707 000
7952 025710 000000
7953
7954 025712 104417
7955
7956 025714 012762 000001 000022
7957
7958 025722 104423
7959 025724 104002
7960
7961 025726 104422
7962 025730 000000
7963 025732 000000
7964 025734 000020
7965 025736 104004
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979 025740 000004
7980 025742 012737 000012 001262
7981 025750 104416
7982 025752 104003
7983
7984 025754 004437 042260
7985 025760 000001

```
*****  
*TEST 76 PROGRAMMING ERROR  
* ISSUE A SUBSYSTEM CLEAR. ISSUE  
* A READ DATA OF 400 WORDS ON CYLINDER 0,  
* TRACK 0, SECTOR 0. DURING READ ISSUE A  
* WRITE TO THE SPARE REGISTER. MAKE SURE  
* PROGRAMMING ERROR SETS.  
*****  
TST76: SCOPE  
MOV #10.,$TIMES ;:DO 10. ITERATIONS  
TSSINIT ;:CLEAR SUBSYSTEM  
ERROR 3 ;:BAD INIT ERROR  
  
JSR R4,LRLOAD ;:LOAD "L" REGS  
RDDATA ;:RDDATA  
-400 ;:-400 WORDS  
IBUFF ;:IBUFF IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0  
0 ;:CYLINDER 0  
  
TLOADRK ;:LOAD RK REGS  
  
MOV #1,RKSPAR(R2) ;:WRITE SPARE REGISTER  
  
TWT16 ;:WAIT FOR INTERRUPT  
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR  
  
TCHKWE ;:CHECK OPERATION WITH EXPECTED ERROR  
0  
0  
PGERR ;:PROG ERROR  
ERROR 4 ;:OR 5,6,7 ;:REPORT ALL DISCREPANCIES
```

```
*****  
*TEST 77 ECC HARD  
* ISSUE A SUBSYSTEM CLEAR. ISSUE  
* A WRITE DATA WORDS CONSISTING OF 177777 TO  
* CYLINDER 0, TRACK 0, SECTOR 0. NOW WRITE  
* ALL ZEROS TO CYLINDER 0, TRACK 0, SECTOR 0.  
* DURING WRITE ISSUE CONTROLLER CLEAR. MAKE  
* SURE PROGRAMMING ERROR IS RESET. NOW  
* ISSUE A READ DATA TO CYLINDER 0, TRACK 0,  
* HEAD 0 AND AN ECC HARD ERROR SHOULD SET.  
*****  
TST77: SCOPE  
MOV #10.,$TIMES ;:DO 10. ITERATIONS  
TSSINIT ;:CLEAR SUBSYSTEM  
ERROR 3 ;:BAD INIT ERROR  
  
JSR R4,GENCOM ;:GENERATE DATA OF ALL ONES  
1
```

7986	025762	000400		400		
7987						
7988	025764	004437	035674	JSR R4,LRLOAD		:LOAD "L" REGS
7989	025770	000123		WRDATA		:WRDATA
7990	025772	177400		-400		: -400 WORDS
7991	025774	072414		OBUFF		:OBUFF IS BUFF ADDRESS
7992	025776	000		.BYTE 0		:SECTOR 0
7993	025777	000		.BYTE 0		:TRACK 0
7994	026000	000000		0		:CYLINDER 0
7995						
7996	026002	104417		TLOADRK		:LOAD RK REGS
7997	026004	104430		TWAT96		:WAIT FOR INTERRUPT
7998	026006	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
7999						
8000	026010	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
8001	026012	104004		ERROR 4	;OR 5, 6, 7, 10	:REPORT ALL ERRORS
8002						
8003	026014	004437	042260	JSR R4,GENCOM		:GENERATE DATA OF ZEROS
8004	026020	000002		2		
8005	026022	000400		400		
8006						
8007	026024	004437	035674	JSR R4,LRLOAD		:LOAD "L" REGS
8008	026030	000123		WRDATA		:WRDATA
8009	026032	177630		-150		: -150 WORDS
8010	026034	072414		OBUFF		:OBUFF IS BUFF ADDRESS
8011	026036	000		.BYTE 0		:SECTOR 0
8012	026037	000		.BYTE 0		:TRACK 0
8013	026040	000000		0		:CYLINDER 0
8014						
8015	026042	104417		TLOADRK		:START OPERATION
8016						
8017	026044	005737	001662	TST INTSET	1\$:	:CHECK IF INTERRUPT HAS OCCURRED
8018	026050	001026		BNE 2\$:YES - MUCH TO SOON. REPORT ERROR
8019	026052	005762	000002	TST RKWC(R2)		:TEST IF NPR'S DONE
8020	026056	001372		BNE 1\$:NO - LOOP
8021						
8022	026060	052762	100000 000000	BIS #CCLR,RKCS1(R2)		:CLEAR CONTROLLER (CROWBAR WRITE)
8023						
8024	026066	004437	035674	JSR R4,LRLOAD		:LOAD "L" REGS
8025	026072	000121		RDDATA		:RDDATA
8026	026074	177400		-400		: -400 WORDS
8027	026076	070414		IBUFF		:IBUFF IS BUFF ADDRESS
8028	026100	000		.BYTE 0		:SECTOR 0
8029	026101	000		.BYTE 0		:TRACK 0
8030	026102	000000		0		:CYLINDER 0
8031						
8032	026104	104417		TLOADRK		:LOAD RK REGS
8033	026106	104425		TWAT48		:WAIT FOR INTERRUPT
8034	026110	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
8035						
8036	026112	104422		TCHKWE		:CHECK OPERATION WITH ERROR
8037	026114	000000		0		
8038	026116	000003		DCKERR!ECHERR		:DATA CHECK AND ECC HARD
8039	026120	000000		0		
8040	026122	104004		ERROR 4	;OR 5,6,7	:REPORT ALL DISCREPANCIES
8041						

8042 026124 000402
8043 026126
8044 026126 104421
8045 026130 104004
8046 026132
8047
8048
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058
8059 026132 000004
8060 026134 012737 000005 001262
8061
8062 026142 104416
8063 026144 104003
8064
8065 026146 004437 035674
8066 026152 000117
8067 026154 000000
8068 026156 000000
8069 026160 000
8070 026161 000
8071 026162 000632
8072
8073 026164 104417
8074 026166 104423
8075 026170 104002
8076 026172 005037 001662
8077 026176 104430
8078 026200 104002
8079
8080 026202 004437 035674
8081 026206 000113
8082 026210 000000
8083 026212 000000
8084 026214 000
8085 026215 000
8086 026216 000000
8087
8088 026220 104417
8089 026222 104423
8090 026224 104002
8091
8092 026226 004437 035674
8093 026232 000125
8094 026234 000000
8095 026236 000000
8096 026240 000
8097 026241 000

BR 3\$;SKIP TO EXIT
2\$: TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
3\$:

```
*****  
:TEST 100 DRIVE TIMING ERROR  
:ISSUE A SUBSYSTEM CLEAR. SEEK TO CYLINDER 632.  
:ISSUE A RECALIBRATE BUT DO NOT WAIT FOR SECOND INTERRUPT.  
:PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER  
:OF CYLINDER 0, TRACK 0. CLOCK THROUGH SEEK  
:AND DRIVE CLEAR MESSAGES. TURN OFF DIAGNOSTIC MODE.  
:DRIVE TIMING ERROR SHOULD SET BECAUSE OF NO DATA  
:TRANSISTIONS ON DATA LINE.  
*****
```

```
TST100: SCOPE  
MOV #5.,$TIMES ;:DO 5. ITERATIONS  
TSSINIT ;:CLEAR SUBSYSTEM  
ERROR 3 ;:BAD INIT ERROR  
JSR R4,LRLOAD ;:LOAD "L" REGS  
SEEK ;:SEEK  
0 ;:0 WORDS  
0 ;:0 IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0  
632 ;:CYLINDER 632  
TLOADRK ;:LOAD RK REGS  
TWAT16 ;:WAIT FOR INTERRUPT  
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR  
CLR INTSET ;:CLEAR INT FLAG  
TWAT96 ;:WAIT FOR SECOND INTERRUPT  
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR  
JSR R4,LRLOAD ;:LOAD "L" REGS  
RECAL ;:RECAL  
0 ;:0 WORDS  
0 ;:0 IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0  
0 ;:CYLINDER 0  
TLOADRK ;:LOAD RK REGS  
TWAT16 ;:WAIT FOR INTERRUPT  
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR  
JSR R4,LRLOAD ;:LOAD "L" REGS  
RDHEAD ;:RDHEAD  
0 ;:0 WORDS  
0 ;:0 IS BUFF ADDRESS  
.BYTE 0 ;:SECTOR 0  
.BYTE 0 ;:TRACK 0
```



```

8098 026242 000000 0 ;CYLINDER 0
8099 026244 012737 000040 001616 MOV #DMD,L.MR1 ;SET DIAG MODE
8100 026252 104417 TLOADRK ;LOAD RK REGS
8101
8102 026254 004437 036316 JSR R4,MCLOCK ;CLOCK CONTROLLER THROUGH SEEK
8103 026260 001062 1062 ;AND CLEAR TO READ
8104
8105 026262 005062 000026 CLR RKMR1(R2) ;RESET DIAG MODE, LET RD HDRS COMPLETE
8106
8107 026266 104424 TWAT32 ;WAIT FOR INTERRUPT
8108 026270 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8109 026272 104422 TCHKWE ;CHECK OPERATION WITH EXP ERROR
8110 026274 010000 DTERR ;DRIVE TIMING ERROR
8111 026276 000000 0
8112 026300 000000 0
8113 026302 104004 ERROR 4 ;OR 5,6,7 ;REPORT ALL DISCREPANCIES
8114
8115 026304 1$:
8116 026304 104416 TSSINIT ;CLEAR SUBSYSTEM
8117 026306 104003 ERROR 3 ;BAD INIT ERROR
8118 026310 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
8119 026316 005037 001662 CLR INTSET ;CLEAR INT FLAG
8120
8121 026322 104437 TWAT8S ;WAIT FOR INTERRUPT FOR END OF RECAL
8122 026324 104002 ERROR 2
8123
8124 026326 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8125 026332 000105 CLEAR ;CLEAR
8126 026334 000000 0 ;0 WORDS
8127 026336 000000 0 ;0 IS BUFF ADDRESS
8128 026340 000 .BYTE 0 ;SECTOR 0
8129 026341 000 .BYTE 0 ;TRACK 0
8130 026342 000000 0 ;CYLINDER 0
8131
8132 026344 104417 TLOADRK ;LOAD RK REGS
8133 026346 104423 TWAT16 ;WAIT FOR INTERRUPT
8134 026350 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8135 026352 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8136 026354 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150
8151 026356 000004 TST101: SCOPE
8152 026360 012737 000012 001262 MOV #10, $TIMES ;DO 10. ITERATIONS
8153 026366 104416 TSSINIT ;CLEAR SUBSYSTEM
  
```

.SBTTL **ERROR FORCING IN DRIVE

```

*****
*TEST 101 INITIALIZE CLEARING SACK
*
* ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE
* DRIVE. ISSUE A SUBSYSTEM CLEAR. PUT CONTROLLER IN
* DIAGNOSTIC MODE. ISSUE A SELECT COMMAND WITH
* MESSAGE ID = 3 AND DRIVE SELECTED = 0. CLOCK THROUGH
* PHASE ADDRESS 6. TURN OFF DIAGNOSTIC MODE. MAKE
* SURE UNIT FIELD ERROR DOES NOT SET.
*****
TST101: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
TSSINIT ;CLEAR SUBSYSTEM
  
```

```
8154 026370 104003 ERROR 3 ;BAD INIT ERROR
8155
8156 026372 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8157 026376 000101 SELDRV ;SELDRV
8158 026400 000000 0 ;0 WORDS
8159 026402 000000 0 ;0 IS BUFF ADDRESS
8160 026404 000 .BYTE 0 ;SECTOR 0
8161 026405 000 .BYTE 0 ;TRACK 0
8162 026406 000000 0 ;CYLINDER 0
8163
8164 026410 104417 TLOADRK ;LOAD RK REGS
8165 026412 104423 TWAT16 ;WAIT FOR INTERRUPT
8166 026414 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8167
8168 026416 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8169 026420 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8170
8171 026422 104416 TSSINIT ;CLEAR SUBSYSTEM
8172 026424 104003 ERROR 3 ;BAD INIT ERROR
8173
8174 026426 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8175 026432 000101 SELDRV ;SELDRV
8176 026434 000000 0 ;0 WORDS
8177 026436 000000 0 ;0 IS BUFF ADDRESS
8178 026440 000 .BYTE 0 ;SECTOR 0
8179 026441 000 .BYTE 0 ;TRACK 0
8180 026442 000000 0 ;CYLINDER 0
8181 026444 012737 000043 001616 MOV #3!DMD,L.MR1 ;SET DIAG MODE AND MESSAGE PAIR 3
8182 026452 005037 001610 CLR L.CS2 ;SELECT DRIVE 0
8183
8184 026456 104417 TLOADRK ;LOAD RK REGS
8185
8186 026460 004437 036316 JSR R4,MCLOCK ;CLOCK THROUGH PHASE ADDRESS 6
8187 026464 001027 1027
8188
8189 026466 042762 000040 000026 BIC #DMD,RKMR1(R2) ;CLEAR MAINTENANCE MODE
8190
8191 026474 104424 TWAT32 ;WAIT FOR INTERRUPT
8192 026476 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
8193
8194 026500 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8195 026502 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8196
8197 *****
8198 :*TEST 102 DRIVE OFF TRACK
8199 :*
8200 :* ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. ISSUE
8201 :* OFFSET OF +1200 MICRO-INCHES. PUT CONTROLLER IN DIAGNOSTIC
8202 :* MODE. ISSUE A WRITE DATA OF 1 WORD TO CYLINDER 0,
8203 :* TRACK 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
8204 :* MESSAGES. TURN OFF DIAGNOSTIC MODE. DRIVE OFF TRACK
8205 :* SHOULD SET IN DRIVE. REPEAT FOR ALL AVAILIABLE DRIVES.
8206 :*
8207 *****
8208 :*ST102: SCOPE
8209 026504 000004 MOV #5, $TIMES ;DO 5. ITERATIONS
026506 012737 000005 001262 TSSINIT ;CLEAR SUBSYSTEM
026514 104416
```

8210	026516	104003		ERROR	3		:BAD INIT ERROR
8211							
8212	026520	004437	035674	JSR	R4,LRLOAD		:LOAD 'L' REGS
8213	026524	000113		RECAL			:RECAL
8214	026526	000000		0			:0 WORDS
8215	026530	000000		0			:0 IS BUFF ADDRESS
8216	026532	000		.BYTE	0		:SECTOR 0
8217	026533	000		.BYTE	0		:TRACK 0
8218	026534	000000		0			:CYLINDER 0
8219							
8220	026536	104417		TLOADRK			:LOAD RK REGS
8221	026540	104423		TWAT16			:WAIT FOR INTERRUPT
8222	026542	104002		ERROR	2		:TO SLOW/NOT COMPLETE ERROR
8223	026544	005037	001662	CLR	INTSET		:CLEAR INTERRUPT FLAG
8224							
8225	026550	104437		TWAT8S			:WAIT FOR INTERRUPT #2
8226	026552	104002		ERROR	2		
8227							
8228	026554	004437	035674	JSR	R4,LRLOAD		:LOAD 'L' REGS
8229	026560	000105		CLEAR			:CLEAR
8230	026562	000000		0			:0 WORDS
8231	026564	000000		0			:0 IS BUFF ADDRESS
8232	026566	000		.BYTE	0		:SECTOR 0
8233	026567	000		.BYTE	0		:TRACK 0
8234	026570	000000		0			:CYLINDER 0
8235							
8236	026572	104417		TLOADRK			:LOAD RK REGS
8237	026574	104423		TWAT16			:WAIT FOR INTERRUPT
8238	026576	104002		ERROR	2		:TO SLOW/NOT COMPLETE ERROR
8239							
8240	026600	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
8241	026602	104004		ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS
8242							
8243	026604	004437	035674	JSR	R4,LRLOAD		:LOAD 'L' REGS
8244	026610	000115		OFFSET			:OFFSET
8245	026612	000000		0			:0 WORDS
8246	026614	000000		0			:0 IS BUFF ADDRESS
8247	026616	000		.BYTE	0		:SECTOR 0
8248	026617	000		.BYTE	0		:TRACK 0
8249	026620	000000		0			:CYLINDER 0
8250	026622	112737	000060 001612	MOVB	#60,L.ASOF		:SET OFFSET AT +1200
8251							
8252	026630	104417		TLOADRK			:LOAD RK REGS
8253	026632	104423		TWAT16			:WAIT FOR INTERRUPT
8254	026634	104002		ERROR	2		:TO SLOW/NOT COMPLETE ERROR
8255							
8256	026636	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
8257	026640	104004		ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS
8258							
8259	026642	005037	001662	CLR	INTSET		:CLEAR INT FLAG
8260							
8261	026646	104424		TWAT32			:WAIT FOR INT #2
8262	026650	104002		ERROR	2		
8263							
8264	026652	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
8265	026654	104004		ERROR	4 ;OR 5, 6, 7		:REPORT ALL ERRORS

```

8266
8267 026656 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8268 026662 000105 CLEAR ;CLEAR
8269 026664 000000 0 ;0 WORDS
8270 026666 000000 0 ;0 IS BUFF ADDRESS
8271 026670 000 .BYTE 0 ;SECTOR 0
8272 026671 000 .BYTE 0 ;TRACK 0
8273 026672 000000 0 ;CYLINDER 0
8274
8275 026674 104417 TLOADRK ;LOAD RK REGS
8276 026676 104423 TWAT16 ;WAIT FOR INTERRUPT
8277 026700 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8278
8279 026702 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8280 026704 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8281
8282 026706 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8283 026712 000023 23 ;WRITE DATA W/O INTERRUPT ENABLE
8284 026714 177777 -1 ;-1 WORDS
8285 026716 072414 OBUFF ;OBUFF IS BUFF ADDRESS
8286 026720 000 .BYTE 0 ;SECTOR 0
8287 026721 000 .BYTE 0 ;TRACK 0
8288 026722 000000 0 ;CYLINDER 0
8289 026724 012737 000040 001616 MOV #DMD,L.MR1 ;SET DIAGNOSTIC MODE
8290
8291 026732 104417 TLOADRK
8292
8293 026734 004437 036316 JSR R4,MCLOCK ;CLOCK THROUGH SEEK & DRIVE CLEAR
8294 026740 001064 1064
8295
8296 026742 005062 000026 CLR RKM1(R2) ;CLEAR DIAGNOSTIC MODE
8297 026746 104426 TWAT64 ;WAIT FOR INTERRUPT
8298 026750 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
8299 026756 104423 TWAT16 ;STALL FOR 16 MS
8300 026760 000240 NOP
8301 026762 000240 NOP
8302
8303 026764 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8304 026770 000101 SELDRV ;SELDRV
8305 026772 000000 0 ;0 WORDS
8306 026774 000000 0 ;0 IS BUFF ADDRESS
8307 026776 000 .BYTE 0 ;SECTOR 0
8308 026777 000 .BYTE 0 ;TRACK 0
8309 027000 000000 0 ;CYLINDER 0
8310 027002 005037 001616 CLR L.MR1 ;RESET DIAG MODE
8311
8312 027006 104417 TLOADRK ;LOAD RK REGS
8313 027010 104423 TWAT16 ;WAIT FOR INTERRUPT
8314 027012 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8315
8316 027014 104422 TCHKWE ;CHECK OPERATION WITH ERROR EXPECTED
8317 027016 000400 DROTERR ;DRIVE OFF TRACK
8318 027020 000000 0
8319 027022 000000 0
8320 027024 104004 ERROR 4; OR 5,6,7 ;REPORT ANY DISCREPANCIES
8321

```

::*****

```
8322      ;*TEST 103      FILE UNSAFE
8323      ;*
8324      ;*      ISSUE A SUBSYSTEM CLEAR.  ISSUE A RECLAIBRATE.  ISSUE
8325      ;*      A READ HEAD OF CYLINDER 0, TRACK 0 IN 24 SECTOR
8326      ;*      FORMAT.  DO A SELECT COMMAND IN 26 SECTOR FORMAT.
8327      ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
8328      ;*      HEADER TO CYLINDER 0, TRACK 0, ONE WORD IN 26 SECTOR
8329      ;*      FORMAT.  CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
8330      ;*      SIMULATE INDEX PULSE.  TURN OFF DIAGNOSTIC MODE.  FILE
8331      ;*      UNSAFE SHOULD SET BECAUSE OF ATTEMPTING TO WRITE
8332      ;*      THROUGH SECTOR PULSE.  REPEAT FOR ALL AVAILIABLE DRIVES.
8333      ;*
8334      ;*****
8335      027026 000004      TST103: SCOPE
8336      027030 012737      MOV      #5, $TIMES      ;;DO 5. ITERATIONS
8337      027036 012737 000005 001262      MOV      #-1, REFMT      ;;SET REFORMAT SWITCH
8338      027044 104416      TSSINIT      ;;CLEAR SUBSYSTEM
8339      027046 104003      ERROR      3      ;;BAD INIT ERROR
8340
8341      027050 004437 035674      JSR      R4, LRLOAD      ;;LOAD 'L' REGS
8342      027054 000113      RECAL      ;;RECAL
8343      027056 000000      0      ;;0 WORDS
8344      027060 000000      0      ;;0 IS BUFF ADDRESS
8345      027062 000      .BYTE      0      ;;SECTOR 0
8346      027063 000      .BYTE      0      ;;TRACK 0
8347      027064 000012      12      ;;CYLINDER 12
8348
8349      027066 104417      TLOADRK      ;;LOAD RK REGS
8350      027070 104423      TWAT16      ;;WAIT FOR INTERRUPT
8351      027072 104002      ERROR      2      ;;TO SLOW/NOT COMPLETE ERROR
8352
8353      027074 104421      TCHKOP      ;;CHECK OPERATION FOR ANY ERRORS
8354      027076 104004      ERROR      4 ;OR 5, 6, 7      ;;REPORT ALL ERRORS
8355
8356      027100 005037 001662      CLR      INTSET      ;;CLEAR INT FLAG
8357      027104 104437      TWAT8S      ;;WAIT FOR SECOND INT
8358      027106 104002      ERROR      2
8359
8360      027110 104421      TCHKOP      ;;CHECK OPERATION FOR ANY ERRORS
8361      027112 104004      ERROR      4 ;OR 5, 6, 7      ;;REPORT ALL ERRORS
8362
8363      027114 004437 035674      JSR      R4, LRLOAD      ;;LOAD 'L' REGS
8364      027120 000105      CLEAR      ;;CLEAR
8365      027122 000000      0      ;;0 WORDS
8366      027124 000000      0      ;;0 IS BUFF ADDRESS
8367      027126 000      .BYTE      0      ;;SECTOR 0
8368      027127 000      .BYTE      0      ;;TRACK 0
8369      027130 000012      12      ;;CYLINDER 12
8370
8371      027132 104417      TLOADRK      ;;LOAD RK REGS
8372      027134 104423      TWAT16      ;;WAIT FOR INTERRUPT
8373      027136 104002      ERROR      2      ;;TO SLOW/NOT COMPLETE ERROR
8374
8375      027140 104421      TCHKOP      ;;CHECK OPERATION FOR ANY ERRORS
8376      027142 104004      ERROR      4 ;OR 5, 6, 7      ;;REPORT ALL ERRORS
8377
```

8378	027144	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
8379	027150	010125		RDHEAD!	CFMT	:RDHEAD!CFMT
8380	027152	000000		0		:0 WORDS
8381	027154	000000		0		:0 IS BUFF ADDRESS
8382	027156	000		.BYTE	0	:SECTOR 0
8383	027157	000		.BYTE	0	:TRACK 0
8384	027160	000012		12		:CYLINDER 12
8385						
8386	027162	104417		TLOADRK		:LOAD RK REGS
8387	027164	104424		TWAT32		:WAIT FOR INTERRUPT
8388	027166	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
8389						
8390	027170	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
8391	027172	104004		ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
8392						
8393	027174	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
8394	027200	000101		SELDRV		:SELDRV
8395	027202	000000		0		:0 WORDS
8396	027204	000000		0		:0 IS BUFF ADDRESS
8397	027206	000		.BYTE	0	:SECTOR 0
8398	027207	000		.BYTE	0	:TRACK 0
8399	027210	000012		12		:CYLINDER 12
8400						
8401	027212	104417		TLOADRK		:LOAD RK REGS
8402	027214	104423		TWAT16		:WAIT FOR INTERRUPT
8403	027216	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
8404						
8405	027220	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
8406	027224	000127		WRHEAD		:WRHEAD
8407	027226	177777		-1		:-1 WORDS
8408	027230	072414		OBUFF		:OBUFF IS BUFF ADDRESS
8409	027232	000		.BYTE	0	:SECTOR 0
8410	027233	000		.BYTE	0	:TRACK 0
8411	027234	000012		12		:CYLINDER 12
8412	027236	012737	000040 001616	MOV	#DMD,L.MR1	:SET DIAGNOSTIC-MODE
8413						
8414	027244	104417		TLOADRK		:LOAD RK REGS
8415	027246	004437	036316	JSR	R4,MCLOCK	:CLOCK THROUGH SEEK AND DRIVE CLEAR
8416	027252	001064		1064		
8417						
8418	027254	052762	000200 000026	BIS	#MIND,RKMR1(R2)	:SET INDEX
8419						
8420	027262	004437	036316	JSR	R4,MCLOCK	:CLOCK INDEX
8421	027266	001001		1001		
8422						
8423	027270	042762	000200 000026	BIC	#MIND,RKMR1(R2)	:CLEAR INDEX
8424						
8425	027276	004437	036316	JSR	R4,MCLOCK	:CLOCK CLEAR
8426	027302	001001		1001		
8427						
8428	027304	005062	000026	CLR	RKMR1(R2)	:CLEAR DIAGNOSTIC MODE
8429						
8430	027310	104426		TWAT64		:WAIT FOR INTERRUPT
8431	027312	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
8432						
8433	027314	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS

8434	027316	104004		ERROR	4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
8435					
8436	027320	004437	035674	JSR	R4,LRLOAD ;LOAD 'L' REGS
8437	027324	000101		SELDRV	;SELDRV
8438	027326	000000		0	;0 WORDS
8439	027330	000000		0	;0 IS BUFF ADDRESS
8440	027332	000		.BYTE	0 ;SECTOR 0
8441	027333	000		.BYTE	0 ;TRACK 0
8442	027334	000012		12	;CYLINDER 12
8443					
8444	027336	005037	001616	CLR	L.MR1 ;CLEAR DIAG MODE
8445					
8446	027342	104417		TLOADRK	;LOAD RK REGS
8447	027344	104423		TWAT16	;WAIT FOR INTERRUPT
8448	027346	104002		ERROR	2 ;TO SLOW/NOT COMPLETE ERROR
8449					
8450	027350	104422		TCHKWE	;CHECK OPERATION WITH EXPECTED ERROR
8451	027352	040400		UNSERR!DROTERR	;UNSAFE AND DRIVE OFF TRACK
8452	027354	000000		0	
8453	027356	000000		0	
8454	027360	104004		ERROR	4; OR 5,6,7 ;REPORT ANY DISCREPANCIES
8455					
8456	027362	104416		TSSINIT	;CLEAR SUBSYSTEM
8457	027364	104003		ERROR	3 ;BAD INIT ERROR
8458					
8459	027366	004437	035674	JSR	R4,LRLOAD ;LOAD 'L' REGS
8460	027372	000101		SELDRV	;SELDRV
8461	027374	000000		0	;0 WORDS
8462	027376	000000		0	;0 IS BUFF ADDRESS
8463	027400	000		.BYTE	0 ;SECTOR 0
8464	027401	000		.BYTE	0 ;TRACK 0
8465	027402	000012		12	;CYLINDER 12
8466					
8467	027404	012737	000001 001616	MOV	#1,L.MR1 ;SET MESSAGE SELECT ONE
8468					
8469	027412				
8470	027412	104417	1\$:	TLOADRK	;LOAD RK REGS
8471	027414	104423		TWAT16	;WAIT FOR INTERRUPT
8472	027416	104002		ERROR	2 ;TO SLOW/NOT COMPLETE ERROR
8473					
8474	027420	104421		TCHKOP	;CHECK OPERATION FOR ANY ERRORS
8475	027422	104004		ERROR	4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8476					
8477	027424	032737	000040 001574	BIT	#S.HDHM,T.MR2 ;TEST IF HEADS HOME
8478	027432	001767		BEQ	1\$
8479					
8480	027434	104416		TSSINIT	;CLEAR SUBSYSTEM
8481	027436	104003		ERROR	3 ;BAD INIT ERROR
8482					
8483	027440	005037	001662	CLR	INTSET ;CLEAR INT FLAG
8484	027444	104434		TWAT159	;WAIT FOR APPROX 160 MS
8485	027446	000240		NOP	;DON'T CARE ERROR RETURN
8486					
8487	027450	104416		TSSINIT	;CLEAR SUBSYSTEM
8488	027452	104003		ERROR	3 ;BAD INIT ERROR
8489					

```

8490 027454 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
8491
8492 027462 104437 TWAT8S ;WAIT FOR SECOND INTERRUPT FROM HDS LOADED
8493 027464 104002 ERROR 2
8494
8495 027466 005037 001616 CLR L.MR1 ;CLEAR MR1
8496
8497 027472 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8498 027476 000105 CLEAR ;CLEAR
8499 027500 000000 0 ;0 WORDS
8500 027502 000000 0 ;0 IS BUFF ADDRESS
8501 027504 000 .BYTE 0 ;SECTOR 0
8502 027505 000 .BYTE 0 ;TRACK 0
8503 027506 000012 12 ;CYLINDER 12
8504
8505 027510 104417 TLOADRK ;LOAD RK REGS
8506 027512 104423 TWAT16 ;WAIT FOR INTERRUPT
8507 027514 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8508
8509 027516 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8510 027520 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8511
8512 027522 004437 042260 JSR R4,GENCOM ;BUILD HEADERS
8513 027526 001200 1200
8514
8515 027530 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
8516 027534 000127 WRHEAD ;WRHEAD
8517 027536 177676 -102 ;-102 WORDS
8518 027540 072414 OBUFF ;OBUFF IS BUFF ADDRESS
8519 027542 000 .BYTE 0 ;SECTOR 0
8520 027543 000 .BYTE 0 ;TRACK 0
8521 027544 000012 12 ;CYLINDER 12
8522
8523 027546 104417 TLOADRK ;LOAD RK REGS
8524 027550 104426 TWAT64 ;WAIT FOR INTERRUPT
8525 027552 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8526
8527 027554 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8528 027556 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8529
8530
8531
8532
8533
8534
8535
8536
8537
8538
8539
8540
8541
8542 027560 000004 TST104: SCOPE
8543 027562 012737 000001 001262 MOV #1,$TIMES ;;DO 1 ITERATION
8544
8545 027570 104416 TSSINIT ;CLEAR SUBSYSTEM
  
```

```

*****
*TEST 104 DUMMY TEST FOR PREVIOUS TEST EXIT
* THIS TEST IS PRESENT TO MAKE $SW08TB TABLE HAVE AN ENTRY
* WHICH RELATES TO 'NEWDRV'. THIS IS NECESSARY IF AN ERROR OCCURS
* IN THE PRECEEDING TEST AND THAT ERROR ABORTS THE TEST.
* IF THIS TEST WERE NOT PRESENT, THE PROGRAM WOULD SKIP THE
* 'NEWDRV' ROUTINE AND GO TO THE TEST FOLLOWING 'NEWDRV'.
*
* IN ADDITION, THE DRIVE IS CLEARED AND THE HEADS ARE ALLOWED
* TO RELOAD. THIS MUST BE DONE TO PREVENT UNEXPECTED INTERRUPTS
* FROM THE DRIVE COMING READY AT A LATER TIME.
*****
  
```



```
8546 027572 104003          ERROR 3          ;BAD INIT ERROR
8547
8548 027574 013762 001626 000010      MOV   DRVNUM,RKCS2(R2) ;LOAD DRIVE NUMBER
8549 027602 005037 036126          CLR   HOLD1
8550 027606 012737 000001 036124      MOV   #1,HOLD
8551 027614 053737 001720 036124      BIS   DTYPE,HOLD
8552 027622 013762 036124 000000      MOV   HOLD,RKCS1(R2)  ;SELECT THE DRIVE
8553 027630 032762 000200 000012 1$:  BIT   #DRDY,RKDS(R2)  ;TEST IF DRIVE READY
8554 027636 001004          RNE   2$          ;BR IF YES
8555 027640 005237 036126          INC   HOLD1        ;ELSE TRY AGAIN
8556 027644 001371          BNE   1$
8557 027646 104002          ERROR 2          ;NO RDY
8558
8559 027650          2$:
8560 027650 104416          TSSINIT          ;CLEAR SUBSYSTEM
8561 027652 104003          ERROR 3          ;BAD INIT ERROR
8562
8563 027654 004437 042260      JSR   R4,GENCOM   ;GENERATE HEADERS FOR CYL 0
8564 027660 001200          1200
8565
8566 027662 004437 035674      JSR   R4,LRLOAD   ;LOAD "L" REGS
8567 027666 000127          WRHEAD          ;WRHEAD
8568 027670 177676          -102           ;-102 WORDS
8569 027672 072414          OBUFF          ;OBUFF IS BUFF ADDRESS
8570 027674 000          .BYTE 0         ;SECTOR 0
8571 027675 000          .BYTE 0         ;TRACK 0
8572 027676 000000          0             ;CYLINDER 0
8573
8574 027700 104417          TLOADRK          ;LOAD RK REGS
8575 027702 104426          TWAT64          ;WAIT FOR INTERRUPT
8576 027704 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
8577
8578 027706 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8579 027710 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8580 027712 005037 001676      CLR   REFMT      ;CLEAR REFORMAT SWITCH
8581          .SBTTL **MULTI-DRIVE OPERATIONS
8582 027716 000004          NEWDRV: SCOPE
8583 027720 012737 000001 001262      MOV   #1,$TIMES   ;DO ONLY ONCE
8584 027726 032737 000200 001630      BIT   #BIT7,DRVBIT ;WERE WE TESTING DRIVE 7?
8585 027734 001026          BNE   3$          ;YES-SKIP
8586
8587 027736 005237 001626          1$:  INC   DRVNUM      ;BUMP TO NEXT SEQUENTIAL ADDRESS
8588 027742 006337 001630          ASL   DRVBIT      ;BUMP DRIVEBIT TO THAT POSITION
8589 027746 033737 001630 001354      BIT   DRVBIT,$DEVM ;IS THIS DRIVE TO BE TESTED?
8590 027754 001005          BNE   2$          ;YES-EXIT
8591 027756 032737 000400 001630      BIT   #BIT8,DRVBIT ;ALL DRIVES TESTED?
8592 027764 001012          BNE   3$          ;YES-EXIT
8593 027766 000763          BR    1$          ;ELSE CHECK NEXT DRIVE AVAILABLE
8594
8595 027770 112737 000004 001102 2$:  MOVB  #4,$STSTM   ;SET TEST NUMBER FOR REPORTS
8596 027776 013701 001626          MOV   DRVNUM,R1
8597 030002 004737 036150          JSR   PC,GTYP1   ;GET DRV TYP FOR NEXT DRV TO TST
8598 030006 000137 004720          JMP   TSTLUP     ;GO TO TEST LOOP TO CHECK THIS DRIVE
8599 030012 005037 001630          3$:  CLR   DRVBIT      ;CLEAR DRIVE BIT
8600 030016 005037 001626          CLR   DRVNUM     ;CLEAR DRIVE NUMBER
8601
```

```
8602
8603
8604
8605
8606
8607
8608
8609 030022 000004
8610 030024 012737 000012 001262
8611 030032 005000
8612 030034 012701 000001
8613 030040 013703 001354
8614 030044 104416
8615 030046 104003
8616 030050 030103 1$: BIT R1,R3 ;TEST IF THIS DRIVE AVAILABLE
8617 030052 001006 BNE 2$ ;YES-SKIP TO SEEK
8618 030054 006301 3$: ASL R1 ;SHIFT DRIVE SELECT BIT
8619 030056 005200 INC R0 ;BUMP DRIVE POSITION COUNTER
8620 030060 032701 000400 BIT #BIT8,R1 ;ALL DRIVE POSITIONS CHECKED
8621 030064 001771 BEQ 1$ ;NO-LOOP
8622 030066 000443 BR 4$ ;SKIP TO RESET
8623
8624 030070 010037 001610 2$: MOV R0,L.CS2 ;LOAD DRIVE NUMBER
8625 030074 004737 036134 JSR PC,GTYP0
8626 030100 012737 000113 001600 MOV #RECAL,L.CS1 ;LOAD RECALIBRATE
8627
8628 030106 104417 TLOADRK ;LOAD RK REGS
8629 030110 104423 TWAT16 ;WAIT FOR INTERRUPT
8630 030112 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8631
8632 030114 005037 001662 CLR INTSET ;CLEAR INTERRUPT FLAG
8633 030120 012705 000764 MOV #500.,R5 ;SET COUNT FOR 8 SECONDS
8634 030124 012762 000012 000C00 MOV #12,RKCS1(R2) ;RESET INTERRUPT ENABLE
8635 030132 016237 000016 001556 12$: MOV RKASOF(R2),T.ASOF ;GET ATTENTION REGISTER
8636 030140 113704 001557 MOVVB T.ASOF+1,R4 ;ADJUST FOR CHECK OF ATTENTION
8637 030144 042704 177400 BIC #177400,R4 ;CLEAR UNUSED BITS
8638 030150 030104 BIT R1,R4 ;CHECK IF ATT SET FROM DRIVE RECAL'ED
8639 030152 001006 BNE 10$ ;YES - SKIP
8640
8641 030154 104423 TWAT16 ;WAIT FOR 16 MS
8642 030156 000240 NOP ;DON'T CARE RETURNS
8643 030160 000240 NOP
8644 030162 005305 DEC R5 ;TATOL WAIT TIME IS 8 SECONDS
8645 030164 001362 BNE 12$ ;CHECK ATTENTION EACH 16 MS
8646 030166 104002 ERROR 2 ;REPORT IF NO ATTENTION IN 8 SEC
8647
8648 030170 10$: TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8649 030170 104421 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8650 030172 104004
8651
8652 030174 000727 4$: BR 3$ ;LOOP FOR NEXT DRIVE
8653 030176 000005 RESET ;UNIBUS RESET
8654 030200 004737 045220 JSR PC,$TKINT ;RESET KEYBOARD INTERRUPT ENABLE
8655
8656 030204 012701 000031 5$: MOV #25.,R1 ;DO A SHORT DELAY
8657 030210 005301 DEC R1
```

```
8658 030212 001376 BNE 5$
8659 030214 004737 034522 JSR PC,OPTTST ;SET UP OPTIONS
8660
8661 030220 104420 TGETRK ;GET RK611 REGS
8662
8663 030222 105737 001557 TSTB T,ASOF+1 ;ALL ATTENTION RESET?
8664 030226 001407 BEQ 6$ ;YES-SKIP
8665
8666 030230 012737 056630 001470 MOV #EMDA,EM12N ;'DRIVE ATT NOT RESET RESULT OF
8667 030236 012737 056754 061112 MOV #EMUR,DF011A ;UNIBUS RESET''
8668 030244 104012 ERROR 12
8669
8670 030246
8671
8672
8673
8674
8675
8676
8677
8678
8679 030246 000004
8680 030250 012737 000012 001262 TST106: SCOPE
8681 030256 005000 MOV #10.,$TIMES ;:DO 10. ITERATIONS
8682 030260 012701 000001 CLR R0 ;:CLEAR DRIVE POSITION COUNTER
8683 030264 013703 001354 MOV #1,R1 ;:PRESET TO TEST POSITION 0
8684 030270 104416 MOV $DEVN,R3 ;:CUT DEVICE MAP
8685 030272 104003 TSSINIT ;:CLEAR SUBSYSTEM
8686 030274 030103 ERROR 3 ;:BAD INIT ERROR
8687 030276 001006 1$: BIT R1,R3 ;:THIS DRIVE AVAILABLE?
8688 030300 006301 BNE 2$ ;:YES-SKIP TO SEEK
8689 030302 005200 3$: ASL R1 ;:SHIFT TO NEXT DRIVE POSITION
8690 030304 032701 000400 INC R0 ;:DUMP POSITION COUNTER
8691 030310 001771 BIT #BIT8,R1 ;:ALL POSITIONS CHECKED
8692 030312 000443 BEQ 1$ ;:NO-LOOP
8693 BR 4$ ;:YES-SKIP TO CLEAR
8694 030314 010037 001610 2$: MOV R0,L,CS2 ;:LOAD DRIVE NUMBER
8695 030320 004737 036134 JSR PC,GTYP0
8696 030324 012737 000113 001600 MOV #RECAL,L,CS1 ;:LOAD RECALIBRATE
8697 030332 104417 TLOADRK ;:LOAD RK REGS
8698 030334 104423 TWAT16 ;:WAIT FOR INTERRUPT
8699 030336 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
8700
8701 030340 005037 001662 CLR INTSET ;:CLEAR INT FLAG
8702 030344 012705 000764 MOV #500.,R5 ;:SET COUNT FOR 8 SECONDS
8703 030350 012762 000012 000000 MOV #12,RKCS1(R2) ;:RESET INTERRUPT ENABLE
8704 030356 016237 000016 001556 12$: MOV RKASOF(R2),T,ASOF ;:GET ATTENTION REGISTER
8705 030364 113704 001557 MOV T,ASOF+1,R4 ;:ADJUST FOR CHECK OF ATTENTION
```

8706	030370	042704	177400			BIC	#177400,R4	:CLEAR UNUSED BITS
8707	030374	030104				BIT	R1,R4	:CHECK IF ATT SET FROM DRIVE RECAL'ED
8708	030376	001006				BNE	10\$:YES - SKIP
8709								
8710	030400	104423				TWAT16		:WAIT FOR 16 MS
8711	030402	000240				NOP		:DON'T CARE RETURNS
8712	030404	000240				NOP		
8713	030406	005305				DEC	R5	:TATOL WAIT TIME IS 8 SECONDS
8714	030410	001362				RNE	12\$:CHECK ATTENTION EACH 16 MS
8715	030412	104002				ERROR	2	:REPORT IF NO ATTENTION IN 8 SEC
8716								
8717	030414				10\$:			
8718	030414	104421				TCHKOP		:CHECK OPERATION FOR ANY ERRORS
8719	030416	104004				ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
8720								
8721	030420	000727				BR	3\$:LOOP FOR NEXT DRIVE
8722								
8723	030422	052762	000040	000010	4\$:	BIS	#SCLR,RKCS2(R2)	:DO SUBSYSTEM CLEAR
8724	030430	012701	000031			MOV	#25.,R1	:DO A SHORT DELAY
8725	030434	005301			5\$:	DEC	R1	
8726	030436	001376				BNE	5\$	
8727								
8728	030440	104420				TGETRK		:GET RK611 REGS
8729								
8730	030442	105737	001557			TSTB	T.ASOF+1	:TEST ALL ATTENTION RESET
8731	030446	001407				BEQ	6\$:YES-SKIP
8732								
8733	030450	012737	056630	001470		MOV	#EMDA,EM12N	: "DRIVE ATT NOT RESET AS RESULT OF
8734	030456	012737	057044	061112		MOV	#EMSCLR,DF011A	:SUBSYSTEM CLEAR"
8735	030464	104012				ERROR	12	
8736								
8737	030466				6\$:			
8738								
8739								
8740								
8741								
8742								
8743								
8744								
8745								
8746								
8747								
8748								
8749								
8750								
8751								
8752								
8753	030466	000004						
8754	030470	012737	000012	001262	TST107: SCOPE	MOV	#10.,\$TIMES	::DO 10. ITERATIONS
8755	030476	013746	001354			MOV	\$DEVM,-(SP)	:PUT DEVICE MAP ON STACK
8756	030502	004437	036242			JSR	R4,BITCNT	:COUNT NUMBER OF BITS(# OF DRIVES)
8757	030506	022627	000001			CMP	(SP)+,#1	:COMPARE TO 1
8758	030512	101007				BHI	2\$:SKIP IF MORE THAN 1
8759	030514	005737	001304			TST	\$PASS	:CHECK IF PASS 0
8760	030520	001002				BNE	1\$:NO-SKIP
8761								

```

8762 030522 104401 052664          TYPE      ,OPR018      ;"OVERLAPPED OPERATION BYPASSED"
8763 030526 000137 031772          1$:      JMP      $EOP      ;GET OUT.
8764
8765 030532 012737 177777 001224 2$:      MOV      #-1,$TMP1    ;SET LOOP CONTROL FLAG
8766 030540 013705 001354          MOV      $DEVM,R5    ;GET DEVICE MAP
8767 030544 005000          CLR      R0          ;CLEAR FOR DRIVE #A
8768 030546 005001          CLR      R1          ;CLEAR FOR DRIVE #B
8769 030550 012703 000001          MOV      #1,R3       ;SET DRIVE POSITION A
8770 030554 012704 000001          MOV      #1,R4       ;SET DRIVE POSITION B
8771 030560 012737 030570 001110  MOV      #3,$LPERR   ;SET LOCAL LOOP ON ERROR
8772 030566 000503          BR       11$        ;GO SET UP POINTERS
8773
8774 030570          3$:
8775 030570 104416          TSSINIT          ;CLEAR SUBSYSTEM
8776 030572 104003          ERROR 3          ;BAD INIT ERROR
8777
8778 030574 010037 001610          MOV      R0,L,CS2   ;LOAD DRIVE A
8779 030600 010037 001626          MOV      R0,DRVNUM  ;LOAD FOR REPORT
8780 030604 004737 036134          JSR      PC,GTYP0   ;
8781 030610 012737 000113 001600  MOV      #RECAL,L,CS1 ;LOAD RECAL
8782
8783 030616 104417          TLOADRK          ;LOAD RK REGS
8784 030620 104423          TWAT16          ;WAIT FOR INTERRUPT
8785 030622 104002          ERROR 2          ;TO SLOW,NOT COMPLETE ERROR
8786 030624 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8787 030626 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8788
8789 030630 005037 001662          CLR      INTSET    ;CLEAR INT FLAG
8790
8791 030634 010137 001610          MOV      R1,L,CS2   ;LOAD DRIVE B
8792 030640 010137 001626          MOV      R1,DRVNUM  ;LOAD FOR REPORT
8793 030644 004737 036150          JSR      PC,GTYP1   ;
8794 030650 012737 000101 001600  MOV      #SELDRV,L,CS1 ;LOAD DRIVE SELECT
8795
8796 030656 104417          TLOADRK          ;LOAD RK REGS
8797 030660 104423          TWAT16          ;WAIT FOR INTERRUPT
8798 030662 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
8799
8800 030664 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8801 030666 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8802
8803 030670 032737 100000 001552  BIT      #SVAL,T,DS  ;CHECK IF STATUS VALID SET
8804 030676 001007          BNE      4$        ;YES - SKIP
8805 030700 012737 056444 001460  MOV      #EMSVAL,EM11N ;"STATUS VALID NOT SET RESULT OF
8806 030706 012737 050250 061112  MOV      #OPER00,DF011A ;DRIVE SELECT"
8807 030714 104011          ERROR 11
8808
8809 030716 005037 001662          4$:      CLR      INTSET    ;CLEAR INT FLAG
8810 030722 104436          TWAT2S          ;WAIT FOR SEEK COMPLETE INTERRUPT
8811 030724 000401          BR       44$      ;NONE RECEIVED - SKIP
8812 030726 000406          BR       55$      ;RECEIVED - SKIP
8813
8814 030730 010037 001626          44$:     MOV      R0,DRVNUM  ;SET DRIVE FOR REPORT
8815 030734 012737 060501 001372  MOV      #DH017,DH2N ;"COMMAND - SELECT AFTER RECAL"
8816 030742 104002          ERROR 2
8817
  
```

```

8818 030744 104420          55$: TGETRK          ;GET RK REGS
8819 030746 032737 100000 001552 BIT #SVAL,T.DS ;TEST IF SVAL STILL SET
8820 030754 001007          BNE 5$          ;YES - SKIP
8821
8822 030756 012737 056444 001510 MOV #EMSVL,EM14N ;"STATUS VALID RESET RESULT OF
8823 030764 012737 060524 061112 MOV #DH018,DF011A ;RECAL COMPLETE ATTENTION AFTER SEL"
8824 030772 104014          ERROR 14
8825
8826 030774 104415          5$: SCOP1          ;LOCAL LOOP TO 3$
8827
8828 : THE FOLLOWING CODE CAUSES THE TEST TO BE RUN ON EVERY COMBINATION
8829 : OF DRIVES AVAILABLE. THE FIRST PASS OF THE PROGRAM WILL USE THE
8830 : LOWEST NUMBER DRIVE AS A AND THE NEXT HIGHER NUMBER DRIVE AS
8831 : B. THE SECOND PASS SWAPS DRIVE A & B. THE THIRD PASS USES
8832 : THE LOWEST NUMBER DRIVE AS B AND THE 3RD HIGHEST NUMBER DRIVE
8833 : AS A. THE FORTH PASS SWAPS A & B AGAIN. THIS CONTINUES
8834 : UNTIL ALL DRIVES HAVE BEEN TESTED WITH THE LOWEST NUMBER
8835 : DRIVE.
8836
8837 : THE SECOND HIGHEST NUMBER DRIVE IS THEN USED AS A AND THE
8838 : THIRD HIGHEST AS B. THEY ARE SWAPPED ON THE NEXT PASS.
8839
8840 : THIS TECHNIQUE IS CONTINUED UNTIL ALL COMBINATIONS ARE
8841 : CHECKED.
8842
8843 030776 005237 001224          11$: INC $TMP1        ;INCREMENT PASS CONTROL
8844 031002 001024          BNE 16$        ;SKIP IF NOT ZERO
8845 : (IT WILL BE ZERO ON THE 1ST PASS)
8846
8847 031004 030305          12$: BIT R3,R5          ;TEST IF BIT POSITION FOR A AT AVAIL DRIVE
8848 031006 001006          BNE 13$        ;YES-SKIP
8849
8850 031010 005200          22$: INC R0          ;BUMP R0 (DRIVE A)
8851 031012 006303          ASL R3          ;SHIFT DRIVE SELECT BIT ONE POSITION
8852 031014 032703 000400 BIT #BIT8,R3    ;IF BIT 8 IS SET, ALL DRIVES HAVE
8853 031020 001771          BEQ 12$        ;BEEN CHECKED; IF NOT CHECK NEXT POSITION
8854 031022 000464          BR 50$         ;DONE-EXIT
8855
8856 031024 010001          13$: MOV R0,R1        ;SET DRIVE B TO THE SAME AS A
8857 031026 010304          MOV R3,R4
8858 031030 005201          14$: INC R1          ;BUMP R1 (DRIVE B)
8859 031032 006304          ASL R4          ;SHIFT SELECTOR BIT ONE POSITION
8860 031034 030405          BIT R4,R5        ;IS THIS DRIVE AVAIL?
8861 031036 001004          BNE 15$        ;YES-SKIP
8862 031040 032704 000400 BIT #BIT8,R4    ;WERE ALL POSITIONS CHECKED?
8863 031044 001771          BEQ 14$        ;NO-LOOP
8864 031046 000452          BR 50$         ;DONE-EXIT
8865
8866 031050 000137 030570          15$: JMP 3$          ;GO DO THE TEST ON THE DRIVE A & B
8867 : CONTAINED IN R0 & R1
8868 031054 032737 000001 001224 16$: BIT #BIT0,$TMP1 ;IS PASS FLAGS ODD?
8869 031062 001410          BEQ 17$        ;NO-SKIP
8870
8871 031064 010046          MOV R0,-(SP)    ;
8872 031066 010346          MOV R3,-(SP)    ;SWAP R0 & R1, R3 & R4
8873 031070 010403          MOV R4,R3      ;TO EXCHANGE DRIVE A & B
  
```

```
8874 031072 010100      MOV      R1,R0
8875 031074 012604      MOV      (SP)+,R4
8876 031076 012601      MOV      (SP)+,R1
8877 031100 000137 030570      JMP      3$          ;REPEAT TEST ON THIS COMBO.
8878
8879 031104 032737 000002 001224 17$:      BIT      #BIT1,$TMP1 ;TEST IF PASS FLAGS AT HALF MODULE 4?
8880 031112 001410          BEQ      19$          ;NO-SKIP TO BUMP DRIVE B
8881 031114 005200          18$:      INC      R0          ;BUMP DRIVE A
8882 031116 006303          ASL      R3          ;SHIFT DRIVE SELECT BIT
8883 031120 030305          BIT      R3,R5      ;AVAILABLE?
8884 031122 001014          BNE      20$          ;YES-SKIP
8885 031124 032703 000400      BIT      #BIT8,R3   ;ALL CHECKED?
8886 031130 001771          BEQ      18$          ;NO-SKIP
8887 031132 000412          BR       21$          ;GO TO NEXT PASS
8888
8889 031134 005201          19$:      INC      R1          ;BUMP DRIVE B
8890 031136 006304          ASL      R4          ;SHIFT DRIVE SELECT BIT
8891 031140 030405          BIT      R4,R5      ;AVAILABLE?
8892 031142 001004          BNE      20$          ;YES-SKIP
8893 031144 032704 000400      BIT      #BIT8,R4   ;ALL CHECKED?
8894 031150 001771          BEQ      19$          ;NO-LOOP
8895 031152 000404          BR       23$          ;YES-SKIP TO NEXT PASS
8896
8897 031154 000137 030570      20$:      JMP      3$          ;GO TEST THIS COMBO
8898
8899 031160 010100          21$:      MOV      R1,R0      ;SET DRIVE 0 TO LOW POSITION THIS PASS
8900 031162 010403          MOV      R4,R3      ;SET SELECT BITS TO AGREE
8901 031164 005037 001224      23$:      CLR      $TMP1      ;CLEAR PASS FLAGS
8902 031170 000137 031010      JMP      22$          ;GO SET UP A & B
8903 031174
8904
8905 *****
8906 :*TEST 110 OVERLAPPED OPERATIONS
8907 :*
8908 :* DO A RECALIBRATE ON BOTH DRIVE A AND DRIVE B. ISSUE A
8909 :* SEEK ON DRIVE A TO CYLINDER 1. IMMEDIATELY ISSUE A WRITE
8910 :* DATA TO CYLINDER 100, TRACK 0, HEAD 0 ON DRIVE B.
8911 :* AT THE END OF THE DATA TRANSFER NO ERRORS SHOULD
8912 :* BE SET AND DRIVE A HAS ATTENTION SET.
8913 :*
8914 :* REPEAT FOR ALL COMBINATIONS OF TWO DRIVES.
8915 :*
8916 :* NOTE: IF ONLY ONE DRIVE IS AVAILABLE THE
8917 :* TEST WILL NOT BE DONE.
8918 *****
8919 031174 000004      TST110: SCOPE
8920 031176 012737 000005 001262      MOV      #5,$TIMES ;;DO 5. ITERATIONS
8921
8922 031204 012737 177777 001224 2$:      MOV      #-1,$TMP1 ;SET LOOP CONTROL FLAG
8923 031212 013705 001354          MOV      $DEVM,R5 ;GET DEVICE MAP
8924 031216 005000          CLR      R0          ;CLEAR FOR DRIVE #A
8925 031220 005001          CLR      R1          ;CLEAR FOR DRIVE #B
8926 031222 012703 000001          MOV      #1,R3      ;SET DRIVE POSITION A
8927 031226 012704 000001          MOV      #1,R4      ;SET DRIVE POSITION B
8928 031232 012737 031242 001110          MOV      #3$,$LPERR ;SET LOCAL LOOP ON ERROR
8929 031240 000555          BR       11$          ;GO SET UP POINTERS
```

```

8930 031242          3$:
8931 031242 104416  TSSINIT          :CLEAR SUBSYSTEM
8932 031244 104003  ERROR 3          :BAD INIT ERROR
8933
8934 031246 010037 001626  MOV R0,DRVNUM    :STORE DRIVE FOR REPORT
8935 031252 010037 001610  MOV R0,L.CS2    :SETUP DRIVE A TO RECAL
8936 031256 004737 036134  JSR PC,GTYP0
8937 031262 012737 000113 001600  MOV #RECAL,L.CS1
8938
8939 031270 104417  TLOADRK          :LOAD RK REGS
8940 031272 104423  TWAT16          :WAIT FOR INTERRUPT
8941 031274 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8942 031276 005037 001662  CLR INTSET      :CLEAR INTERRUPT FLAG
8943
8944 031302 104437  TWAT8S          :WAIT FOR SECOND INTERRUPT
8945 031304 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8946
8947 031306 012737 000105 001600  MOV #CLEAR,L.CS1 :SET UP TO CLEAR DRIVE
8948 031314 104417  TLOADRK          :LOAD RK REGS
8949 031316 104423  TWAT16          :WAIT FOR INTERRUPT
8950 031320 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8951 031322          4$:
8952 031322 104421  TCHKOP          :CHECK OPERATION FOR ANY ERRORS
8953 031324 104004  ERROR 4 ;OR 5, 6, 7 :REPORT ALL ERRORS
8954 031326 032737 040000 001540  BIT #DI,T.CS1   :TEST IF DI STILL SET
8955 031334 001372  BNE 4$          :YES - LOOP
8956 031336 010137 001626  MOV R1,DRVNUM    :STORE DRIVE FOR REPORT
8957 031342 010137 001610  MOV R1,L.CS2    :SETUP DRIVE B TO RECAL
8958 031346 004737 036150  JSR PC,GTYP1
8959 031352 012737 000113 001600  MOV #RECAL,L.CS1
8960
8961 031360 104417  TLOADRK          :LOAD RK REGS
8962 031362 104423  TWAT16          :WAIT FOR INTERRUPT
8963 031364 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8964 031366 005037 001662  CLR INTSET      :CLEAR INTERRUPT FLAG
8965 031372 104437  TWAT8S          :WAIT FOR SECOND INTERRUPT
8966 031374 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8967 031376 012737 000105 001600  MOV #CLEAR,L.CS1 :SET UP DRIVE CLEAR
8968 031404 104417  TLOADRK          :LOAD RK REGS
8969 031406 104423  TWAT16          :WAIT FOR INTERRUPT
8970 031410 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR
8971 031412          5$:
8972 031412 104421  TCHKOP          :CHECK OPERATION FOR ANY ERRORS
8973 031414 104004  ERROR 4 ;OR 5, 6, 7 :REPORT ALL ERRORS
8974 031416 032737 040000 001540  BIT #DI,T.CS1   :TEST IF DI STILL SET
8975 031424 001372  BNE 5$          :YES - LOOP
8976
8977 031426 010037 001626  MOV R0,DRVNUM    :STORE DRIVE FOR REPORT
8978 031432 010037 001610  MOV R0,L.CS2    :SETUP DRIVE A TO SEEK
8979 031436 004737 036134  JSR PC,GTYP0
8980 031442 012737 000001 001614  MOV #1,L.DCYL   :TO CYL 1
8981 031450 012737 000117 001600  MOV #SEEK,L.CS1
8982
8983 031456 104417  TLOADRK          :LOAD RK REGS
8984 031460 104423  TWAT16          :WAIT FOR INTERRUPT
8985 031462 104002  ERROR 2          :TO SLOW/NOT COMPLETE ERROR

```



```

8986
8987 031464 010137 001626      MOV      R1,DRVNUM      ;STORE DRIVE FOR REPORT
8988 031470 010137 001610      MOV      R1,L.CS2      ;SETUP DRIVE B TO WRITE DATA
8989 031474 004737 036150      JSR      PC,GTYP1
8990 031500 012737 000100 001614  MOV      #100,L.DCYL    ;AT CYL 100
8991 031506 012737 177400 001602  MOV      #-400,L.WC     ;400 WORDS
8992 031514 012737 072414 001604  MOV      #OBUFF,L.BA
8993 031522 012737 000123 001600  MOV      #WRDATA,L.CS1
8994
8995 031530 104417      TLOADRK      ;LOAD RK REGS-DO WRITE
8996
8997 031532 104427      TWAT80      ;WAIT FOR INTERRUPT
8998 031534 104002      ERROR 2     ;TO SLOW/NOT COMPLETE ERROR
8999
9000 031536 104421      TCHKOP      ;CHECK OPERATION FOR ANY ERRORS
9001 031540 104004      ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
9002
9003 031542 010037 001626      MOV      R0,DRVNUM      ;STORE DRIVE FOR REPORT
9004 031546 130337 001557      BITB      R3,T.ASOF+1   ;CHECK IF DRIVE ATTENTION IS DRIVE A
9005 031552 001007      BNE      10$           ;YES-SKIP
9006 031554 012737 056630 001460  MOV      #EMDA,EM11N    ;DRIVE ATTENTION NOT SET RESULT OF
9007 031562 012737 050362 061112  MOV      #EMSK,DF011A   ;SEEK''
9008 031570 104011      ERROR 11
9009
9010 031572 104415      10$: SCOPI      ;LOCAL LOOP TO 3$
9011
9012 :
9013 : THE FOLLOWING CODE CAUSES THE TEST TO BE RUN ON EVERY COMBINATION
9014 : OF DRIVES AVAILABLE. THE FIRST PASS OF THE PROGRAM WILL USE THE
9015 : LOWEST NUMBER DRIVE AS A AND THE NEXT HIGHER NUMBER DRIVE AS
9016 : B. THE SECOND PASS SWAPS DRIVE A & B. THE THIRD PASS USES
9017 : THE LOWEST NUMBER DRIVE AS B AND THE 3RD HIGHEST NUMBER DRIVE
9018 : AS A. THE FORTH PASS SWAPS A & B AGAIN. THIS CONTINUES
9019 : UNTIL ALL DRIVES HAVE BEEN TESTED WITH THE LOWEST NUMBER
9020 : DRIVE.
9021 :
9022 : THE SECOND HIGHEST NUMBER DRIVE IS THEN USED AS A AND THE
9023 : THIRD HIGHEST AS B. THEY ARE SWAPPED ON THE NEXT PASS.
9024 :
9025 : THIS TECHNIQUE IS CONTINUED UNTIL ALL COMBINATIONS ARE
9026 : CHECKED.
9027 031574 005237 001224      11$: INC      $TMP1      ;INCREMENT PASS CONTROL
9028 031600 001024      BNE      16$           ;SKIP IF NOT ZERO
9029 :
9030 : (IT WILL BE ZERO ON THE 1ST PASS)
9031 031602 030305      12$: BIT      R3,R5     ;TEST IF BIT POSITION FOR A AT AVAIL DRIVE
9032 031604 001006      BNE      13$           ;YES-SKIP
9033
9034 031606 005200      22$: INC      R0         ;BUMP R0 (DRIVE A)
9035 031610 006303      ASL      R3           ;SHIFT DRIVE SELECT BIT ONE POSITION
9036 031612 032703 000400  BIT      #BIT8,R3     ;IF BIT 8 IS SET, ALL DRIVES HAVE
9037 031616 001771      BEQ      12$          ;BEEN CHECKED; IF NOT CHECK NEXT POSITION
9038 031620 000464      BR       50$          ;DONE-EXIT
9039
9040 031622 010001      13$: MOV      R0,R1      ;SET DRIVE B TO THE SAME AS A
9041 031624 010304      MOV      R3,R4

```



```
9098 031772 $EOP:
9099 031772 000004 SCOPE
9100 031774 005037 001102 CLR $TSTNM ;;ZERO THE TEST NUMBER
9101 032000 005037 001262 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
9102 032004 005237 001304 INC $PASS ;;INCREMENT THE PASS NUMBER
9103 032010 042737 100000 001304 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
9104 032016 005327 DEC (PC)+ ;;LOOP?
9105 032020 000001 $EOPCT: .WORD 1
9106 032022 003063 BGT $DOAGN ;;YES
9107 032024 012737 MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
9108 032026 000001 $ENDCT: .WORD 1
9109 032030 032020 $EOPCT
9110 032032 104401 032040 TYPE ,65$ ;;TYPE ASCIZ STRING
9111 032036 000407 BR 64$ ;;GET OVER THE ASCIZ
9112 ;;65$: .ASCIZ <12><15>/END PASS #/
9113 032056 64$:
9114 032056 013746 001304 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
9115 ;;TYPE PASS NUMBER
9116 032062 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
9117 032064 104401 032072 TYPE ,67$ ;;TYPE ASCIZ STRING
9118 032070 000421 BR 66$ ;;GET OVER THE ASCIZ
9119 ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
9120 66$:
9121 032134 013746 001112 MOV $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
9122 ;;TOTAL NUMBER OF ERRORS
9123 032140 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
9124 032142 104401 001273 TYPE , $CRLF ;;TYPE CARRIAGE RETURN, LINE FEED
9125 032146 005037 001112 CLR $ERTTL ;;CLEAR ERROR TOTAL
9126 032152 013700 000042 $GET4: MOV @#42,R0 ;;GET MONITOR ADDRESS
9127 032156 001405 BEQ $DOAGN ;;BRANCH IF NO MONITOR
9128 032160 000005 RESET ;;CLEAR THE WORLD
9129 032162 004710 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
9130 032164 000240 NOP ;;SAVE ROOM
9131 032166 000240 NOP ;;FOR
9132 032170 000240 NOP ;;ACT11
9133 032172 $DOAGN:
9134 032172 000137 JMP @(PC)+ ;;RETURN
9135 032174 032202 $RTNAD: .WORD TSTAPT
9136 032176 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
9137 032202 .EVEN
9138 032202 122737 000001 001316 TSTAPT: CMPB #APTENV,$ENV ;;RUNNING UNDER APT MODE ?
9139 032210 001007 BNE 2$ ;;BRANCH IF NOT
9140 032212 023727 001304 000002 CMP $PASS,#2 ;;TWO PASS DONE ?
9141 032220 103403 BLO 2$ ;;BRANCH IF NOT
9142 032222 005237 001102 1$: INC $TSTNM ;;INCREMENT TEST NUMBER
9143 032226 000775 BR 1$ ;;WAITING LOOP FOR APT DUMP THE NEXT PROG
9144 032230 000137 003150 2$: JMP NEWPAS ;;REPEAT THE SAME TEST IF NOT UNDER APT
9145 .SBTTL ROUTINE TO SIZE MEMORY
9146
9147 *****
9148 *CALL:
9149 * JSR PC,$SIZE
9150 * RETURN
9151 *$LSTAD WILL CONTAIN:
9152 * WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
9153 * WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
```

```
9154      :*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
9155      :*$KT11 IS THE MEMORY MANAGEMENT KEY
9156      :*$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
9157      :*      MUST BE SETUP BEFORE THE CALL
9158      :*$BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
9159      :*      DETERMINED BY ROUTINE
9160
9161 032234 010046  $SIZE:  MOV  R0,-(SP)      ;;SAVE R0 ON THE STACK
9162 032236 010146  MOV  R1,-(SP)      ;;SAVE R1 ON THE STACK
9163 032240 010246  MOV  R2,-(SP)      ;;SAVE R2 ON THE STACK
9164 032242 010346  MOV  R3,-(SP)      ;;SAVE R3 ON THE STACK
9165 032244 013746 000114  MOV  @#114,-(SP)   ;;SAVE MEMORY ERROR VECTOR PS & PC
9166 032250 013746 000116  MOV  @#116,-(SP)
9167 032254 012737 000116 000114  MOV  #116,@#114   ;;IGNORE PARITY ERRORS WHILE SIZING
9168 032262 012737 000002 000116  MOV  #RTI,@#116
9169 032270 013746 000004  MOV  @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
9170 032274 013746 000006  MOV  @#ERRVEC+2,-(SP)
9171 032300 010600  MOV  SP,R0        ;;SAVE THE STACK POINTER
9172      ;;SET THE ERRVEC PS TO THE PRESENT PS
9173 032302 104400  TRAP                ;;PUSH OLD PSW AND PC ON STACK
9174 032304 012637 000006  MOV  (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
9175 032310 012701 003776  MOV  #3776,R1     ;;SETUP ADDRESS
9176 032314 105727  TSTB (PC)+        ;;USE MEMORY MANAGEMENT?
9177 032316 000200  $KT11: .WORD 200   ;;SET TO USE MEMORY MANAGEMENT
9178 032320 100062  BPL $SCORE        ;;BR IF NO
9179 032322 012737 032460 000004  MOV  #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
9180 032330 005737 177572  TST  @#SR0        ;;KT11 ARE YOU THERE?
9181 032334 052737 100000 032316  BIS  #100000,$KT11 ;;YES--SET KT11 KEY
9182 032342 005046  CLR  -(SP)        ;;INITIALIZE FOR "PAR" LOADING
9183 032344 012702 172340  MOV  #KIPAR0,R2   ;;ADDRESS OF FIRST "PAR"
9184 032350 012703 000010  MOV  #^D8,R3     ;;LOAD EIGHT "PAR."S AND EIGHT "PDR."S
9185 032354 012762 077406 177740 1$:  MOV  #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
9186 032362 011622  MOV  (SP),(R2)+   ;;LOAD "PAR"
9187 032364 062716 000200  ADD  #200,(SP)   ;;UPDATE FOR NEXT "PAR"
9188 032370 077307  SOB  R3,1$       ;;LOOP UNTIL ALL EIGHT ARE LOADED
9189 032372 012742 177600  MOV  #177600,-(R2) ;;SETUP KIPAR7 FOR I/O
9190 032376 005042  CLR  -(R2)       ;;SETUP KIPAR6 FOR TESTING
9191 032400 012737 032416 000004  MOV  #2$,@#ERRVEC  ;;CATCH TIMEOUT IF NO SR3
9192 032406 012737 000020 172516  MOV  #20,@#SR3    ;;ENABLE 22 BIT MODE
9193 032414 000401  BR   3$         ;;THIS PDP-11 HAS A SR3 REGISTER
9194 032416 022626 2$:  CMP  (SP)+,(SP)+ ;;CLEAN OFF THE STACK--NO SR3
9195 032420 005237 177572 3$:  INC  @#SR0       ;;TURN ON MEMORY MANAGEMENT
9196 032424 012737 032450 000004  MOV  #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
9197 032432 005737 143776 4$:  TST  @#143776   ;;TRAP ON NON-EX-MEM
9198 032436 062712 000040  ADD  #40,(R2)    ;;MAKE A 1K STEP
9199 032442 023712 172356  CMP  @#KIPAR7,(R2) ;;LAST ONE?
9200 032446 101371  BHI  4$         ;;NO--TRY IT
9201 032450 011202  $KTOUT: MOV (R2),R2 ;;GET LAST BANK+1
9202 032452 005037 177572  CLR  @#SR0       ;;TURN OFF MEMORY MANAGEMENT
9203 032456 000421  BR   $SIZEX
9204 032460 042737 100000 032316  $KTNEX: BIC #100000,$KT11 ;;KT11 NON-EXISTENT
9205 032466 012737 032516 000004  $SCORE: MOV # $SCROUT,@#ERRVEC ;;SET FOR TIMEOUT
9206 032474 005002  CLR  R2         ;;SET UP BANK
9207 032476 062701 004000 1$:  ADD  #4000,R1   ;;INCREMENT BY 1K
9208 032502 062702 000040  ADD  #40,R2     ;;1K STEP
9209 032506 005711  TST  (R1)       ;;TRAP ON TIME OUT
```

```
9210 032510 022701 177776      CMP      #177776,R1      ;;LAST ONE
9211 032514 001370              BNE      1$            ;;NO--TRY AGAIN
9212 032516 162701 004000      $CROUT: SUB      #4000,R1
9213 032522 162702 000040      $SIZEX: SUB      #40,R2      ;;DROP BACK
9214 032526 010006              MOV      R0,SP        ;;RESTORE THE STACK
9215 032530 012637 000006      MOV      (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
9216 032534 012637 000004      MOV      (SP)+,@#ERRVEC
9217 032540 012637 000116      MOV      (SP)+,@#116   ;;RESTORE MEMORY ERROR VECTOR
9218 032544 012637 000114      MOV      (SP)+,@#114
9219 032550 010137 032572      MOV      R1,$LSTAD    ;;LAST ADDRESS
9220 032554 010237 032574      MOV      R2,$LSTBK    ;;LAST BANK
9221 032560 012603              MOV      (SP)+,R3     ;;RESTORE R3
9222 032562 012602              MOV      (SP)+,R2     ;;RESTORE R2
9223 032564 012601              MOV      (SP)+,R1     ;;RESTORE R1
9224 032566 012600              MOV      (SP)+,R0     ;;RESTORE R0
9225 032570 000207              RTS      PC
9226 032572 000000      $LSTAD: .WORD    0      ;;CONTAINS THE LAST ADDRESS
9227 032574 000000      $LSTBK: .WORD    0      ;;CONTAINS THE LAST BANK
9228
9229
9230
9231
9232
9233
9234
9235
9236
9237
9238
9239
9240
9241
9242 032576
9243 032576 104407
9244
9245
9246 032600 021627 001002
9247 032604 101002
9248 032606 000137 033614
9249 032612 032777 040000 146320 1$: BIT      #BIT14,@SWR
9250 032620 001131              BNE      $OVER       ;;YES IF SW14=1
9251
9252 032622 000416      ;#####START OF CODE FOR THE XOR TESTER#####
9253      $XTSTR: BR      6$
9254 032624 013746 000004      MOV      @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
9255 032630 012737 032650 000004      MOV      #5$,@#ERRVEC  ;;SET FOR TIMEOUT
9256 032636 005737 177060      TST      @#177060     ;;TIME OUT ON XOR?
9257 032642 012637 000004      MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9258 032646 000500              BR      $SVLAD       ;;GO TO THE NEXT TEST
9259 032650 022626      .5$: CMP      (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
9260 032652 012637 000004      MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9261 032656 000440              BR      7$          ;;LOOP ON THE PRESENT TEST
9262 032660
9263 032660 032777 000400 146252 6$:;#####END OF CODE FOR THE XOR TESTER#####
9264 032666 001421              BIT      #BIT08,@SWR  ;;LOOP ON SPEC. TEST?
9265 032670 005046              BEQ      2$          ;;BR IF NO
                          CLR      -(SP)                      ;;CLEAR A TEMP. LOCATION
```

9266	032672	117716	146242			MOVW	@SWR,(SP)	::	PICKUP THE DESIRED TEST NUMBER
9267	032676	001414				BEQ	8\$::	BRANCH IF BAD TEST NUMBER IN SWR
9268	032700	022716	000110			CMP	#110,(SP)	::	CHECK THE NUMBER IN THE SWR
9269	032704	002411				BLT	8\$::	BRANCH IF TEST NUMBER IS OUT OF RANGE
9270	032706	011637	001102			MOV	(SP), \$TSTNM	::	UPDATE THE TEST NUMBER
9271	032712	005316				DEC	(SP)	::	BACKUP BY ONE
9272	032714	006316				ASL	(SP)	::	SCALE THE TEST NUMBER AS AN INDEX
9273	032716	062716	033122			ADD	#\$SW08TBL,(SP)	::	FORM THE ADDRESS OF TEST POINTER
9274	032722	013637	001106			MOV	@(SP)+,\$LPADR	::	SET LOOP ADDRESS TO DESIRED TEST
9275	032726	000466				BR	\$OVER	::	GO LOOP ON THE TEST
9276	032730	005726			8\$:	TST	(SP)+	::	CLEAN THE BAD TEST NUMBER OFF OF THE STACK
9277	032732	105737	001103		2\$:	TSTB	\$ERFLG	::	HAS AN ERROR OCCURRED?
9278	032736	001421				BEQ	3\$::	BR IF NO
9279	032740	123737	001115	001103		CMPB	\$ERMAX,\$ERFLG	::	MAX. ERRORS FOR THIS TEST OCCURRED?
9280	032746	101015				BHI	3\$::	BR IF NO
9281	032750	032777	001000	146162		BIT	#BIT09,@SWR	::	LOOP ON ERROR?
9282	032756	001404				BEQ	4\$::	BR IF NO
9283	032760	013737	001110	001106	7\$:	MOV	\$LPERR,\$LPADR	::	SET LOOP ADDRESS TO LAST SCOPE
9284	032766	000446				BR	\$OVER	::	
9285	032770	105037	001103		4\$:	CLRB	\$ERFLG	::	ZERO THE ERROR FLAG
9286	032774	005037	001262			CLR	\$TIMES	::	CLEAR THE NUMBER OF ITERATIONS TO MAKE
9287	033000	000415				BR	1\$::	ESCAPE TO THE NEXT TEST
9288	033002	032777	004000	146130	3\$:	BIT	#BIT11,@SWR	::	INHIBIT ITERATIONS?
9289	033010	001011				BNE	1\$::	BR IF YES
9290	033012	005737	001304			TST	\$PASS	::	IF FIRST PASS OF PROGRAM
9291	033016	001406				BEQ	1\$::	INHIBIT ITERATIONS
9292	033020	005237	001104			INC	\$ICNT	::	INCREMENT ITERATION COUNT
9293	033024	023737	001262	001104		CMP	\$TIMES,\$ICNT	::	CHECK THE NUMBER OF ITERATIONS MADE
9294	033032	002024				BGE	\$OVER	::	BR IF MORE ITERATION REQUIRED
9295	033034	012737	000001	001104	1\$:	MOV	#1,\$ICNT	::	REINITIALIZE THE ITERATION COUNTER
9296	033042	013737	033120	001262		MOV	\$MXCNT,\$TIMES	::	SET NUMBER OF ITERATIONS TO DO
9297	033050	105237	001102		\$SVLAD:	INCB	\$TSTNM	::	COUNT TEST NUMBERS
9298	033054	113737	001102	001302		MOVW	\$TSTNM,\$TESTN	::	SET TEST NUMBER IN APT MAILBOX
9299	033062	011637	001106			MOV	(SP),\$LPADR	::	SAVE SCOPE LOOP ADDRESS
9300	033066	011637	001110			MOV	(SP),\$LPERR	::	SAVE ERROR LOOP ADDRESS
9301	033072	005037	001264			CLR	\$ESCAPE	::	CLEAR THE ESCAPE FROM ERROR ADDRESS
9302	033076	112737	000001	001115		MOVW	#1,\$ERMAX	::	ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9303	033104	013777	001102	146030	\$OVER:	MOV	\$TSTNM,@DISPLAY	::	DISPLAY TEST NUMBER
9304	033112	013716	001106			MOV	\$LPADR,(SP)	::	FUDGE RETURN ADDRESS
9305	033116	000002				RTI		::	FIXES PS
9306	033120	003720			\$MXCNT:	2000.		::	MAX. NUMBER OF ITERATIONS
9307	033122				\$SW08TBL:			::	
9308	033122	003164				.WORD	TST1+2	::	STARTING ADDRESS OF TEST 1
9309	033124	003302				.WORD	TST2+2	::	STARTING ADDRESS OF TEST 2
9310	033126	003370				.WORD	TST3+2	::	STARTING ADDRESS OF TEST 3
9311	033130	004620				.WORD	TST4+2	::	STARTING ADDRESS OF TEST 4
9312	033132	004726				.WORD	TST5+2	::	STARTING ADDRESS OF TEST 5
9313	033134	005104				.WORD	TST6+2	::	STARTING ADDRESS OF TEST 6
9314	033136	005232				.WORD	TST7+2	::	STARTING ADDRESS OF TEST 7
9315	033140	005336				.WORD	TST10+2	::	STARTING ADDRESS OF TEST 10
9316	033142	006262				.WORD	TST11+2	::	STARTING ADDRESS OF TEST 11
9317	033144	006400				.WORD	TST12+2	::	STARTING ADDRESS OF TEST 12
9318	033146	006516				.WORD	TST13+2	::	STARTING ADDRESS OF TEST 13
9319	033150	006760				.WORD	TST14+2	::	STARTING ADDRESS OF TEST 14
9320	033152	007112				.WORD	TST15+2	::	STARTING ADDRESS OF TEST 15
9321	033154	007322				.WORD	TST16+2	::	STARTING ADDRESS OF TEST 16

9322	033156	007512	.WORD	TST17+2	:: STARTING ADDRESS OF TEST	17
9323	033160	007646	.WORD	TST20+2	:: STARTING ADDRESS OF TEST	20
9324	033162	010514	.WORD	TST21+2	:: STARTING ADDRESS OF TEST	21
9325	033164	011112	.WORD	TST22+2	:: STARTING ADDRESS OF TEST	22
9326	033166	011324	.WORD	TST23+2	:: STARTING ADDRESS OF TEST	23
9327	033170	011542	.WORD	TST24+2	:: STARTING ADDRESS OF TEST	24
9328	033172	011666	.WORD	TST25+2	:: STARTING ADDRESS OF TEST	25
9329	033174	012020	.WORD	TST26+2	:: STARTING ADDRESS OF TEST	26
9330	033176	012176	.WORD	TST27+2	:: STARTING ADDRESS OF TEST	27
9331	033200	012330	.WORD	TST30+2	:: STARTING ADDRESS OF TEST	30
9332	033202	012624	.WORD	TST31+2	:: STARTING ADDRESS OF TEST	31
9333	033204	013020	.WORD	TST32+2	:: STARTING ADDRESS OF TEST	32
9334	033206	013254	.WORD	TST33+2	:: STARTING ADDRESS OF TEST	33
9335	033210	013442	.WORD	TST34+2	:: STARTING ADDRESS OF TEST	34
9336	033212	013674	.WORD	TST35+2	:: STARTING ADDRESS OF TEST	35
9337	033214	014062	.WORD	TST36+2	:: STARTING ADDRESS OF TEST	36
9338	033216	014304	.WORD	TST37+2	:: STARTING ADDRESS OF TEST	37
9339	033220	014472	.WORD	TST40+2	:: STARTING ADDRESS OF TEST	40
9340	033222	014664	.WORD	TST41+2	:: STARTING ADDRESS OF TEST	41
9341	033224	015060	.WORD	TST42+2	:: STARTING ADDRESS OF TEST	42
9342	033226	015264	.WORD	TST43+2	:: STARTING ADDRESS OF TEST	43
9343	033230	015470	.WORD	TST44+2	:: STARTING ADDRESS OF TEST	44
9344	033232	016330	.WORD	TST45+2	:: STARTING ADDRESS OF TEST	45
9345	033234	016534	.WORD	TST46+2	:: STARTING ADDRESS OF TEST	46
9346	033236	016740	.WORD	TST47+2	:: STARTING ADDRESS OF TEST	47
9347	033240	017156	.WORD	TST50+2	:: STARTING ADDRESS OF TEST	50
9348	033242	017362	.WORD	TST51+2	:: STARTING ADDRESS OF TEST	51
9349	033244	017566	.WORD	TST52+2	:: STARTING ADDRESS OF TEST	52
9350	033246	017712	.WORD	TST53+2	:: STARTING ADDRESS OF TEST	53
9351	033250	020036	.WORD	TST54+2	:: STARTING ADDRESS OF TEST	54
9352	033252	020162	.WORD	TST55+2	:: STARTING ADDRESS OF TEST	55
9353	033254	020306	.WORD	TST56+2	:: STARTING ADDRESS OF TEST	56
9354	033256	020432	.WORD	TST57+2	:: STARTING ADDRESS OF TEST	57
9355	033260	020504	.WORD	TST60+2	:: STARTING ADDRESS OF TEST	60
9356	033262	020564	.WORD	TST61+2	:: STARTING ADDRESS OF TEST	61
9357	033264	021050	.WORD	TST62+2	:: STARTING ADDRESS OF TEST	62
9358	033266	021232	.WORD	TST63+2	:: STARTING ADDRESS OF TEST	63
9359	033270	021420	.WORD	TST64+2	:: STARTING ADDRESS OF TEST	64
9360	033272	022124	.WORD	TST65+2	:: STARTING ADDRESS OF TEST	65
9361	033274	022342	.WORD	TST66+2	:: STARTING ADDRESS OF TEST	66
9362	033276	022446	.WORD	TST67+2	:: STARTING ADDRESS OF TEST	67
9363	033300	023070	.WORD	TST70+2	:: STARTING ADDRESS OF TEST	70
9364	033302	023512	.WORD	TST71+2	:: STARTING ADDRESS OF TEST	71
9365	033304	023622	.WORD	TST72+2	:: STARTING ADDRESS OF TEST	72
9366	033306	024470	.WORD	TST73+2	:: STARTING ADDRESS OF TEST	73
9367	033310	025162	.WORD	TST74+2	:: STARTING ADDRESS OF TEST	74
9368	033312	025404	.WORD	TST75+2	:: STARTING ADDRESS OF TEST	75
9369	033314	025662	.WORD	TST76+2	:: STARTING ADDRESS OF TEST	76
9370	033316	025742	.WORD	TST77+2	:: STARTING ADDRESS OF TEST	77
9371	033320	026134	.WORD	TST100+2	:: STARTING ADDRESS OF TEST	100
9372	033322	026360	.WORD	TST101+2	:: STARTING ADDRESS OF TEST	101
9373	033324	026506	.WORD	TST102+2	:: STARTING ADDRESS OF TEST	102
9374	033326	027030	.WORD	TST103+2	:: STARTING ADDRESS OF TEST	103
9375	033330	027562	.WORD	TST104+2	:: STARTING ADDRESS OF TEST	104
9376	033332	030024	.WORD	TST105+2	:: STARTING ADDRESS OF TEST	105
9377	033334	030250	.WORD	TST106+2	:: STARTING ADDRESS OF TEST	106

9378 033336 030470 .WORD TST107+2 :: STARTING ADDRESS OF TEST 107
9379 033340 031176 .WORD TST110+2 :: STARTING ADDRESS OF TEST 110
9380 033342 031774 .WORD \$EOP+2 : ADDRESS OF END OF PASS
9381 033344 044172 .WORD ABTFAIL+2 : ADDRESS OF ABORT FAILURE HANDLER

9382
9383 .SBTTL APT COMMUNICATIONS ROUTINE
9384

9385 *****
9386 033346 112737 000001 033612 \$ATY1: MOV #1,\$FFLG :: TO REPORT FATAL ERROR
9387 033354 112737 000001 033610 \$ATY3: MOV #1,\$MFLG :: TO TYPE A MESSAGE
9388 033362 000403 BR \$ATYC
9389 033364 112737 000001 033612 \$ATY4: MOV #1,\$FFLG :: TO ONLY REPORT FATAL ERROR
9390 033372 \$ATYC:
9391 033372 010046 MOV R0,-(SP) :: PUSH R0 ON STACK
9392 033374 010146 MOV R1,-(SP) :: PUSH R1 ON STACK
9393 033376 105737 033610 TSTB \$MFLG :: SHOULD TYPE A MESSAGE?
9394 033402 001450 BEQ 5\$:: IF NOT: BR
9395 033404 122737 000001 001316 CMPB #APTENV,\$ENV :: OPERATING UNDER APT?
9396 033412 001031 BNE 3\$:: IF NOT: BR
9397 033414 132737 000100 001317 BITB #APTSPOOL,\$ENVM :: SHOULD SPOOL MESSAGES?
9398 033422 001425 BEQ 3\$:: IF NOT: BR
9399 033424 017600 000004 MOV @4(SP),R0 :: GET MESSAGE ADDR.
9400 033430 062766 000002 000004 ADD #2,4(SP) :: BUMP RETURN ADDR.
9401 033436 005737 001276 1\$: TST \$MSGTYPE :: SEE IF DONE W/ LAST XMISSION?
9402 033442 001375 BNE 1\$:: IF NOT: WAIT
9403 033444 010037 001312 MOV R0,\$MSGAD :: PUT ADDR IN MAILBOX
9404 033450 105720 2\$: TSTB (R0)+ :: FIND END OF MESSAGE
9405 033452 001376 BNE 2\$
9406 033454 163700 001312 SUB \$MSGAD,R0 :: SUB START OF MESSAGE
9407 033460 006200 ASR R0 :: GET MESSAGE LNTH IN WORDS
9408 033462 010037 001314 MOV R0,\$MSGGLT :: PUT LENGTH IN MAILBOX
9409 033466 012737 000004 001276 MOV #4,\$MSGTYPE :: TELL APT TO TAKE MSG.
9410 033474 000413 BR 5\$
9411 033476 017637 000004 033922 3\$: MOV @4(SP),4\$:: PUT MSG ADDR IN JSR LINKAGE
9412 033504 062766 000002 000004 ADD #2,4(SP) :: BUMP RETURN ADDRESS
9413 033512 013746 177776 MOV 177776,-(SP) :: PUSH 177776 ON STACK
9414 033516 004737 044202 JSR PC,\$TYPE :: CALL TYPE MACRO
9415 033522 000000 4\$: .WORD 0
9416 033524 5\$:
9417 033524 105737 033612 10\$: TSTB \$FFLG :: SHOULD REPORT FATAL ERROR?
9418 033530 001416 BEQ 12\$:: IF NOT: BR
9419 033532 005737 001316 TST \$ENV :: RUNNING UNDER APT?
9420 033536 001413 BEQ 12\$:: IF NOT: BR
9421 033540 005737 001276 11\$: TST \$MSGTYPE :: FINISHED LAST MESSAGE?
9422 033544 001375 BNE 11\$:: IF NOT: WAIT
9423 033546 017637 000004 001300 MOV @4(SP),\$FATAL :: GET ERROR #
9424 033554 062766 000002 000004 ADD #2,4(SP) :: BUMP RETURN ADDR.
9425 033562 005237 001276 INC \$MSGTYPE :: TELL APT TO TAKE ERROR
9426 033566 105037 033612 12\$: CLRB \$FFLG :: CLEAR FATAL FLAG
9427 033572 105037 033611 CLRB \$LFLG :: CLEAR LOG FLAG
9428 033576 105037 033610 CLRB \$MFLG :: CLEAR MESSAGE FLAG
9429 033602 012601 MOV (SP)+,R1 :: POP STACK INTO R1
9430 033604 012600 MOV (SP)+,R0 :: POP STACK INTO R0
9431 033606 000207 RTS PC :: RETURN
9432 033610 000 \$MFLG: .BYTE 0 :: MESSG. FLAG
9433 033611 000 \$LFLG: .BYTE 0 :: LOG FLAG


```

9434 033612 000
9435 033614 033614
9436 000200
9437 000001
9438 000100
9439 000040
9440
9441
9442
9443
9444
9445
9446
9447
9448
9449
9450
9451
9452
9453
9454 033614
9455 033614 104407
9456 033616 105237 001103
9457 033622 001775
9458 033624 013777 001102 145310
9459 033632 032777 002000 145300
9460 033640 001402
9461 033642 104401 001266
9462 033646 005237 001112
9463 033652 011637 001116
9464 033656 162737 000002 001116
9465 033664 117737 145226 001114
9466 033672 032777 020000 145240
9467 033700 001055
9468 033702 021627 001002
9469 033706 101046
9470
9471 033710 016637 000004 001116
9472 033716 162737 000002 001116
9473 033724 104401 033770
9474 033730 013746 001116
9475 033734 104402
9476 033736 104401 033776
9477 033742 162716 000004
9478 033746 011637 001116
9479 033752 013746 001116
9480 033756 104402
9481 033760 104401 001273
9482 033764 022626
9483 033766 000422
9484 033770 050200 036503 000040
9485 033776 020040 047125 054105
9486 034004 042520 052103 042105
9487 034012 052040 040522 020120
9488 034020 047524 000040
9489

```

```

$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 $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
        BIT #BIT10,@SWR    ;;BELL ON ERROR?
        BEQ 1$              ;;NO - SKIP
        TYPE $BELL         ;;RING BELL
1$:     INC $ERTTL          ;;COUNT THE NUMBER OF ERRORS
        MOV (SP),$ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
        SUB #2,$ERRPC
        MOVB @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
        BIT #BIT13,@SWR    ;;SKIP TYPEOUT IF SET
        BNE 20$            ;;SKIP TYPEOUTS
        CMP (SP),#1002     ;;IF RETURN PC LESS THAN 1002
        BHI 12$            ;;ERROR IS ILLEGAL TRAP
        ;;PROCESS UNEXPECTED TRAP OR INTERRUPT
        MOV 4(SP),$ERRPC   ;;GET PC AT TIME OF FALSE TRAP
        SUB #2,$ERRPC     ;;ADJUST PC
        TYPE 10$          ;;TYPE HEADER
        MOV $ERRPC,-(SP)  ;;SAVE $ERRPC FOR TYPEOUT
        TYPOC              ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        TYPE .11$
        SUB #4,(SP)       ;;GET FALSE TRAP VECTOR ADDR
        MOV (SP),$ERRPC
        MOV $ERRPC,-(SP)  ;;SAVE $ERRPC FOR TYPEOUT
        TYPOC              ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        TYPE .$CRLF
        CMP (SP)+,(SP)+  ;;POP FALSE TRAP VECTOR PC&ADDR
        BR 20$
10$:   .ASCIZ <200>'PC= '
11$:   .ASCIZ ' UNEXPECTED TRAP TO '

        .EVEN

```

```
9490 034024 12$:
9491 034024 004737 034136 JSR PC,TYPERR ;;GO TO USER ERROR ROUTINE
9492 034030 104401 001273 TYPE ,SCLF
9493 034034 20$:
9494 034034 122737 000001 001316 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
9495 034042 001007 BNE 2$ ;;NO,SKIP APT ERROR REPORT
9496 034044 113737 001114 034056 MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
9497 034052 004737 033364 JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT
9498 034056 000 21$:
9499 034057 000 .BYTE 0
10000 034060 000777 22$: BR 22$ ;;APT ERROR LOOP
10001 034062 005777 145052 2$: TST @SWR ;;HALT ON ERROR
10002 034066 100002 BPL 3$ ;;SKIP IF CONTINUE
10003 034070 000000 HALT ;;HALT ON ERROR!
10004 034072 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
10005 034074 032777 001000 145036 3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?
10006 034102 001402 BEQ 4$ ;;BR IF NO
10007 034104 013716 001110 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
10008 034110 005737 001264 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
10009 034114 001402 BEQ 5$ ;;BR IF NONE
10010 034116 013716 001264 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
10011 034122 5$:
10012 034122 022737 032162 000042 CMP #SENDAD,@#4? ;;ACT-11 AUTO-ACCEPT?
10013 034130 001001 BNE 6$ ;;BRANCH IF NO
10014 034132 000000 HALT ;;YES
10015 034134 6$:
10016 034134 000002 RTI ;;RETURN
10017 *****
10018 ;SBTTL TYPE ERROR ROUTINE
10019 ;*ENTRY JSR PC,TYPERR
10020 ;*RETURN RTS PC
10021 ;*
10022 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
10023 ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
10024 ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
10025 ;*THE ERROR.
10026 ;*****
10027 TYPERR: SAVREG
10028 034136 104413 MOVB $TSTNM,$TESTN ;GET TEST NUMBER FOR REPORT
10029 034140 113737 001102 001302 BIC #177400,$TESTN ;CLEAR UNUSED BITS
10030 034146 042737 177400 001302 MOVB $ITEMB,R0 ;ENTER ERROR NUMBER
10031 034154 113700 001114 BIC #177400,R0 ;CLEAR UNUSED BITS
10032 034160 042700 177400 DEC R0 ;FORM INDEX FOR ERROR TABLE
10033 034164 005300 ASL R0
10034 034166 006300 ASL R0
10035 034170 006300 ASL R0
10036 034174 062700 001360 1$: ADD #ERRTB,R0 ;FORM ADDRESS OF ERROR ENTRY
10037 034200 012037 034214 MOV (R0)+,2$ ;GET EM POINTER
10038 034204 001404 BEQ 3$ ;BRANCH IF THERE ISN'T ONE
10039 034206 104401 001273 TYPE ,SCLF ;TYPE CARRIAGE RETURN LINE FEED
10040 034212 104401 TYPE ;TYPE ERROR MESSAGE (EM)
10041 034214 000000 .WORD 0 ;EM POINTER GOES HERE
10042 034216 012037 034232 2$: MOV (R0)+,4$ ;GET DH POINTER
10043 034222 001404 BEQ 5$ ;BRANCH IF THERE ISN'T ONE
10044 034224 104401 001273 TYPE ,SCLF ;TYPE CR-LF
10045 034230 104401 TYPE ;TYPE DATA HEADER
```

```

9546 034232 000000      4$: .WORD 0 ;DH POINTER GOES HERE
9547 034234 012001      5$: MOV (R0)+,R1 ;GET DT POINTER
9548 034236 001445      BEQ 20$ ;BRANCH IF THERE ARE NONE
9549 034240 005004      CLR R4 ;SET INDENT SWITCH
9550 034242 012000      MOV (R0)+,R0 ;GET DF POINTER
9551 034244 012002      MOV (R0)+,R2 ;STORE NUMBER OF DH'S
9552 034246 104401 001273      TYPE .$CRLF
9553 034252 112003      10$: MOV (R0)+,R3 ;GET & STORE NUMBER OF DATA WORDS
9554 034254 105720      TSTB (R0)+ ;BUMP PAST FORMAT WORD
9555 034256 005703      TST R3 ;TEST IF ANY DATA FOR THIS HEADER
9556 034260 001416      BEQ 14$ ;NO - SKIP DATA PRINT
9557 034262 005704      TST R4 ;CHECK FOR INDENT
9558 034264 001004      BNE 12$ ;YES, GO INDENT
9559 034266 013146      11$: MOV @ (R1)+, -(SP) ;PUT FIRST DATA WORD ON STACK
9560 034270 104402      TYPOC ;TYPE IT
9561 034272 005303      DEC R3 ;MORE DATA WORDS
9562 034274 001403      BEQ 13$ ;NO-BRANCH
9563 034276 104401 051053      12$: TYPE .SPACE2 ;TYPE SEPARATORS
9564 034302 000771      BR 11$ ;LOOP
9565 034304 104401 001273      13$: TYPE .$CRLF ;TYPE <CR><LF>
9566 034310 005710      TST (R0) ;CHECK IF NEXT HEADER AVAILABLE
9567 034312 001401      BEQ 14$ ;NO, DO NOT CHANGE INDENT
9568 034314 005104      COM R4 ;CHANGE INDENT
9569 034316 005302      14$: DEC R2 ;MORE DH'S?
9570 034320 003414      BLE 20$ ;NO-BRANCH
9571 034322 012037 034342      15$: MOV (R0)+, 18$ ;GET NEXT DH POINTER
9572 034326 001751      BEQ 10$ ;IF 0 GET DATA
9573 034330 005704      TST R4 ;INDENT?
9574 034332 001402      BEQ 17$ ;NO, BRANCH
9575 034334 104401 051053      TYPE .SPACE2 ;TYPE INDENT
9576 034340 104401      17$: TYPE ;TYPE DH
9577 034342 000000      18$: .WORD 0 ;DH POINTER GOES HERE
9578 034344 104401 001273      TYPE .$CRLF
9579 034350 000740      BR 10$ ;GET DATA
9580 034352 104414      20$: RESREG
9581 034354 005237 001632      INC ERRCNT ;INCREMENT THE ERROR COUNT
9582 034360 032777 010000 144552      BIT #SW12,@SWR ;CHECK IF SWITCH 12 SET
9583 034366 001421      BEQ 25$ ;NO, RETURN
9584 034370 022737 000024 001632      CMP #20.,ERRCNT ;CHECK IF ERROR THRESHOLD EXCEEDED
9585 034376 103015      BHS 25$ ;NO, RETURN
9586 034400 104401 053317      TYPE .ABORT ;TYPE "PROGRAM ABORTED BECAUSE
9587      ;ERROR THRESHOLD EXCEEDED"
9588 034404 005737 000042      TST 42 ;CHECK IF CHAIN MODE
9589 034410 001407      BEQ 22$ ;NO, HALT PROCESSOR
9590 034412 012706 001100      MOV #STACK,SP ;INITIALIZE STACK
9591 034416 012737 000001 032020      MOV #1,$EOPCT ;FORCE END OF PROGRAM
9592 034424 000137 031772      JMP $EOP
9593 034430 000000      22$: HALT
9594 034432 000207      25$: RTS PC
9595      .SBTTL NON-EXISTANT MEMORY AND INTERRUPT CHECK HANDLER
9596      ;* THIS ROUTINE SETS THE INTERRUPT FLAG AND DOES AN RTI.
9597      ;* THIS IS THE INDICATION TO THE ROUTINE CHECKING
9598      ;* NON-EXISTANT MEMORY THAT AN INTERRUPT DID OCCUR.
9599
9600 034434 005237 001662      NEXINT: INC INTSET ;BUMP THE INTERRUPT COUNTER
9601 034440 000002      RTI

```

```
9602
9603
9604
9605
9606
9607
9608
9609 034442 000240
9610 034444 005237 001662
9611 034450 000002
9612
9613
9614
9615
9616
9617 034452 032777 020000 144460
9618 034460 001003
9619 034462 104401 053546
9620 034466 104402
9621 034470 012737 000001 032020
9622 034476 012706 001100
9623 034502 000137 031772
9624
9625
9626
9627
9628
9629 034506 005237 001674
9630 034512 042777 000200 145170
9631 034520 000002
9632
9633
9634
9635
9636
9637
9638 034522 104413
9639 034524 013746 000004
9640 034530 013746 000006
9641 034534 012737 034754 000004
9642 034542 012737 000340 000006
9643 034550 005037 001670
9644 034554 042737 000100 001664
9645 034562 005737 170200
9646 034566 000240
9647 034570 000240
9648 034572 005737 177770
9649 034576 000240
9650 034600 000240
9651 034602 012737 177750 001672
9652 034610 052737 002000 001664
9653 034616 052737 000100 001664
9654 034624 000507
9655
9656 034626 013703 001714
9657 034632 012704 000001
```

```

.SBTTL RK611 INTERRUPT HANDLER
:* MOST INTERRUPTS FROM THE RK611 ARE HANDLED BY THIS ROUTINE.  ACTUAL
:* PROCESSING AS A RESULT OF THE INTERRUPT IS LEFT TO THE MAIN
:* PROGRAM.  THE HANDLER JUST SETS A FLAG TO INDICATE THE
:* INTERRUPT OCCURRED.

INTHLR: NOP
        INC     INTSET      ;BUMP THE INTERRUPT FLAG
        RTI      ;RETURN.

.SBTTL MEMORY PARITY ERROR TRAP HANDLER
:* MEMORY PARITY TRAPS WILL BE REPORTED BY THIS ROUTINE.  THE REPORT
:* WILL INCLUDE THE PC AT TIME OF FAILURE AND ABORT THE PROGRAM.

PERHLR: BIT     #BIT13,@SWR  ;TEST IF INHIBIT REPORT
        BNE     1$          ;YES - SKIP
        TYPE    ,EM3        ;TYPE PARITY ERROR MESSAGE
        TYPOC   ;AND PC VALUE
1$:     MOV     #1,$EOPCT    ;FORCE END OF PROGRAM
        MOV     #STACK,SP   ;CLEAN OFF STACK
        JMP     $EOP

.SBTTL LINE CLOCK INTERRUPT HANDLER
:* THE LINE CLOCK INTERRUPT HANDLER WILL INCREMENT THE LCLKTK
:* (LINE CLOCK TICK COUNTER) EACH TIME AN INTERRUPT OCCURS.

LCKHLR: INC     LCLKTK      ;INCREMENT CLOCK TICK COUNTER
        BIC     #BIT7,@KWLADD ;CLEAR MONITOR BIT
        RTI

:*****
.SBTTL OPTION PRESENT TEST AND SETUP
:* THIS ROUTINE CHECKS IF THE MEMORY PARITY OPTION AND THE
:* LINE CLOCK ARE ON THE SYSTEM.  FLAGS ARE SET IF PRESENT; CLEARED
:* OTHERWISE, AND THE APPROPRIATE INTERRUPT VECTORS ARE SET UP.
OPTTST: SAVREG
        MOV     ERRVEC,-(SP) ;STORE OLD NEM CONTENTS
        MOV     ERRVEC+2,-(SP)
        MOV     #20$,ERRVEC  ;DET INTERRUPT FOR NEM
        MOV     #PR7,ERRVEC+2 ;SET PRIORITY
        CLR     MEMPARG      ;CLEAR PARITY WORD FLAGS
        BIC     #PARPRE,OPTFLG ;CLEAR PARITY PRESENT FLAG
        TST     170200       ;TEST IF EITHER 11/70 OR 11/44
        NOP
        NOP
        TST     177770       ;NOW SEE IF IT'S 11/70
        NOP
        NOP
        MOV     #177750,CSRPTR ;SET POINTER FOR 11/70
        BIS     #CP1170,OPTFLG ;SET 11/70 FLAG
        BIS     #PARPRE,OPTFLG ;SET PARITY PRESENT FLAG
        BR     35$          ;GO TO VECTOR SETUP

3$:     MOV     MEMBAS,R3    ;SET UP POINTER TO FIRST PARITY CSR
        MOV     #1,R4       ;INIT MASK
```

9658	034636	012713	000001	6\$:	MOV	#1,(R3)	:SET PARITY DETECT IN THAT CSR
9659	034642	005713			TST	(R3)	
9660	034644	052713	000002		BIS	#BIT1,(R3)	:IS THIS ECC MEMORY?
9661	034650	032713	000002		BIT	#BIT1,(R3)	
9662	034654	001061			BNE	47\$:YES - EXIT
9663	034656	050437	001670		BIS	R4,MEMPAR	:SET PARITY PRESENT BIT
9664				:	BIT	#PARPRE,OPTFLG	:WAS PARITY PRESENT SET BEFORE
9665				:	BNE	10\$:YES - SKIP
9666	034662	013700	001716		MOV	MMVECA,R0	:SET UP FOR PARITY TRAPS
9667	034666	012720	034766		MOV	#40\$,(R0)+	: TO 40\$
9668	034672	012710	000340		MOV	#PR7,(R0)	
9669	034676	012700	072640		MOV	#OBUFF+224,R0	:SET POINTER TO BUFFER WHERE BAD PARITY
9670							: IS USED IN THE TESTS
9671	034702	012713	000004		MOV	#BIT2,(R3)	:SET TO WRITE WRONG PARITY
9672	034706	012710	177777		MOV	#-1,(R0)	:WRITE WORD BAD
9673	034712	012713	000001		MOV	#1,(R3)	:SET TO DETECT PARITY ERROR
9674	034716	005710			TST	(R0)	:READ BAD WORD
9675	034720	042737	000100	001664	BIC	#PARPRE,OPTFLG	:REV 005
9676	034726	012713	000002		MOV	#2,(R3)	:TRUN OFF CZR ERROR ENABLE
9677	034732	012710	000000		MOV	#0,(R0)	:CLEAR BAD PARITY
9678				:	BIC	#7,(R3)	:REV 006
9679				:	MOV	#0,(R0)	:REV 006
9680	034736	000442			BR	35\$:REV 005
9681	034740	062703	000002	10\$:	ADD	#2,R3	:BUMP TO NEXT CSR
9682	034744	000241			CLC		
9683	034746	006104			ROL	R4	:SHIFT MASK
9684	034750	001332			BNE	6\$:TEST IF ALL DONE
9685	034752	000434			BR	35\$:YES - SKIP
9686							
9687	034754	022626		20\$:	CMP	(SP)+,(SP)+	:CLEAR STACK
9688	034756	012737	035040	000004	MOV	#30\$,ERRVEC	:SET NEW NEM TRAP ADDRESS
9689	034764	000720			BR	3\$:GO TO CSR CHECKS
9690							
9691	034766	022626		40\$:	CMP	(SP)+,(SP)+	:CLEAR STACK
9692	034770	010337	001672		MOV	R3,CSRPTR	:SET CSR POINTER FOR CSR TO BE USED
9693	034774	052737	000100	001664	BIS	#PARPRE,OPTFLG	:SET PARITY PRESENT FLAG
9694	035002	012713	000000		MOV	#0,(R3)	:TURN OFF CSR ERROR ENABLE
9695	035006	012710	000000		MOV	#0,(R0)	:WRITE GOOD PARITY
9696	035012	012713	000001		MOV	#1,(R3)	:SET TO DETECT PARITY ERRORS
9697	035016	000412			BR	35\$:EXIT
9698							
9699	035020	012713	000000	47\$:	MOV	#0,(R3)	:CLEAR CSR
9700	035024	042737	000100	001664	BIC	#PARPRE,OPTFLG	:FORGET THEM ALL
9701	035032	005037	001670		CLR	MEMPAR	:FORGET THEM ALL
9702	035036	000402			BR	35\$:EXIT
9703	035040	022626		30\$:	CMP	(SP)+,(SP)+	:CLEAR STACK
9704	035042	000736			BR	10\$:GO CHECK NEXT CSR
9705	035044	013700	001716	35\$:	MOV	MMVECA,R0	:SET UP POINTER TO PARITY VECTOR
9706	035050	005737	001670		TST	MEMPAR	:TEST IF ANY PARITY PRESENT
9707	035054	001004			BNE	37\$:YES - SKIP
9708	035056	032737	002000	001664	BIT	#CP1170,OPTFLG	:TEST IF 11/70
9709	035064	001405			BEQ	39\$:NO - SKIP
9710	035066	012720	034452	37\$:	MOV	#PERHLR,(R0)+	:SET UP PARITY VECTOR
9711	035072	012710	000340		MOV	#PR7,(R0)	:SET PRIORITY
9712	035076	000403			BR	38\$	
9713	035100	012720	000116	39\$:	MOV	#116,(R0)+	:ELSE RESTORE TRAP CATCHER

```

9714 035104 005010          CLR      (R0)
9715 035106 012737 034434 000004 38$:  MOV      #NEXINT,ERRVEC ;SET UP NEM VECTOR FOR LINE CLOCK TEST
9716 035114 005037 001662          CLR      INTSET      ;CLEAR INTERRUPT COUNTER
9717 035120 013700 001712          MOV      KWLVEC,R0     ;SET POINTER TO VECTOR
9718 035124 012720 034506          MOV      #LCKHLR,(R0)+ ;INSERT ADDRESS OF INTERRUPT HDLR
9719 035130 012710 000340          MOV      #PR7,(R0)     ;INSERT PRIORITY
9720 035134 012777 000100 144546  MOV      #BIT6,@KWLADD ;LOAD KW11-L FOR INTERRUPT ENABLE
9721 035142 000240          NOP
9722 035144 000240          NOP
9723 035146 000240          NOP
9724 035150 005737 001662          TST      INTSET      ;TEST IN NEM ON KW11-P REFERENCE
9725 035154 001003          BNE      4$          ;THIS BRANCH WILL BYPASS SET 'JP OF
9726                                     ;CLOCK OPTION
9727 035156 052737 100000 001664 4$:  BIS      #LCLKPR,OPTFLG ;SET CLOCK PRESENT FLAG
9728 035164 012701 000006          MOV      #6,R1        ;RESTORE OLD VECTOR
9729 035170 005037 001662          CLR      INTSET      ;CLEAR INT FLAG
9730 035174 012637 000006          MOV      (SP)+,ERRVEC+2
9731 035200 012637 000004          MOV      (SP)+,ERRVEC
9732 035204 012746 000000          MOV      #PRO,-(SP)   ;RESTORE PRIORITY TO 0
9733 035210 012746 035216          MOV      #12$,-(SP)
9734 035214 000002          RTI
9735                                     12$:
9736 035216 104414          RESREG
9737 035220 000207          RTS      PC
9738
9739 *****
9740 .SBTTL LOOP ON INTERNAL ERROR
9741 :* THIS ROUTINE IS USED TO PROVIDE TIGHT SCOPE LOOPS. THE CALLER
9742 :* IS EXPECTED TO SET $LPERR TO THE START OF THE SCOPE LOOP
9743 :* TO BE EXECUTED ON ERROR.
9744
9745 035222 032777 001000 143710 SCOP1$: BIT      #SW9,@SWR     ;CHECK IF LOOP ON ERROR
9746 035230 001405          BEQ      5$          ;NO, CONTINUE
9747 035232 105737 001103          TSTB    $ERFLG      ;CHECK IF ERROR OCCURRED
9748 035236 001402          BEQ      5$
9749 035240 013716 001110          MOV      $LPERR,(SP) ;LOAD ERROR RETURN
9750 035244 000002          RTI          ;RETURN
9751
9752 .SBTTL LINE CLOCK CALIBRATE
9753 :* WAITS FOR A LINE CLOCK INTERRUPT TO CALIBRATE THE INTERRUPTS
9754 :* TO A MEANINGFUL TIME VALUE. IN ADDITION IT PRESETS
9755 :* THE TIMCNT IF THERE IS NO LINE CLOCK. TIMCNT IS USED IN THE
9756 :* LINE CLOCK SIMULATOR.
9757 035246 005037 001674          CLKCAL: CLR      LCLKTK   ;CLEAR TICK COUNTER
9758 035252 032737 100000 001664  BIT      #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
9759 035260 001004          BNE      1$          ;YES - SKIP
9760 035262 012737 000020 001654  MOV      #16.,TIMCNT  ;ELSE PRESET TIMCNT
9761 035270 000410          BR       2$          ;AND EXIT
9762 035272 005737 001662 1$:  TST      INTSET      ;TEST IF INTERRUPT HAS OCCURRED
9763 035276 001005          BNE      2$          ;YES- ABORT CALIBRATION
9764 035300 005737 001674          TST      LCLKTK     ;WAIT FOR CLOCK TICK
9765 035304 001772          BEQ      1$          ;NOT YET - LOOP
9766 035306 005037 001674          CLR      LCLKTK     ;CLEAR TICK COUNT
9767 035312 000207 2$:  RTS      PC          ;RETURN
9768
9769 .SBTTL LINE CLOCK SIMULATION ROUTINE

```

9770 : * THIS ROUTINE IS USED TO SIMULATE THE LINE CLOCK. TO
9771 : * DO THIS THE VALUE STORED IN MILCNT IS USED AS THE
9772 : * BASE AND REPRESENTS THE NUMBER OF TIMES A DECREMENT
9773 : * AND BRANCH LOOP CAN BE DONE IN 1 MILLISECOND. THE
9774 : * TIMCNT VALUE IS DECREMENTED AND IF IT REACHED 0 THE
9775 : * LINE CLOCK TICK COUNTER IS BUMPED. THEN THE TIMCNT
9776 : * IS RESET TO 16 (REPRESENTS 16 MILLISECONDS PER LINE CLOCK
9777 : * TICK).
9778 : *
9779 : * THUS THE ROUTINE RETURNS TO THE CALLER AFTER 1 MILLISECOND
9780 : * AND BUMPS THE LINE CLOCK TICK COUNTER FOR EACH 16 CALLS.
9781 : *
9782 : *

9783 035314 010046 MYTIME: MOV R0, -(SP) ;SAVE R0
9784 035316 013700 001652 MOV MILCNT, R0 ;SET COUNT
9785 035322 005737 001662 1\$: TST INTSET ;TEST IF INTERRUPT SET
9786 035326 001012 BNE 2\$;YES - SKIP
9787 035330 005300 DEC R0 ;DECREMENT COUNT TO ZERO
9788 035332 001373 BNE 1\$
9789 035334 005337 001654 DEC TIMCNT ;DECREMENT TIMCNT
9790 035340 001005 BNE 2\$;IF NOT ZERO - EXIT
9791 035342 005237 001674 INC LCLKTK ;ELSE BUMP TICK COUNTER
9792 035346 012737 000020 001654 MOV #16, TIMCNT ;RESET TIME COUNT
9793 035354 012600 2\$: MOV (SP)+, R0 ;RESTORE R0
9794 035356 000207 RTS PC ;RETURN

9795
9796 .SBTTL WAIT FOR INTERRUPT ROUTINE
9797 : * THE ROUTINE IS ENTERED BY ONE OF FOURTEEN TRAP CALLS. THE CALL
9798 : * SPECIFIES HOW MANY TICKS OF THE LINE CLOCK ARE TO ELAPSE
9799 : * WHILE WAITING FOR INTERRUPT. IF INTERRUPT DOES NOT OCCUR
9800 : * IN THAT PERIOD OF TIME, AN ERROR MESSAGE IS PREPARED
9801 : * (BUT NOT PRINTED IN THE ROUTINE) AND THEN RETURNS TO THE
9802 : * LOCATION FOLLOWING THE CALL. IF INTERRUPT OCCURS THE
9803 : * RETURN IS BUMPED BY 2. NORMALLY AN ERROR CALL WILL
9804 : * BE IN THE LOCATION AFTER THE CALL TO INTERRUPT WAIT.
9805 : *

9806 035360 104413 IWAT8S: SAVREG ;ENTRY FOR 8 SECOND WAIT
9807 035362 012700 000764 MOV #500, R0
9808 035366 000463 BR WATSRT
9809 035370 104413 IWAT1M: SAVREG ;ENTRY FOR 1 MIN WAIT
9810 035372 012700 007246 MOV #3750, R0
9811 035376 000457 BR WATSRT
9812 035400 104413 IWAT2S: SAVREG ;ENTRY FOR 2 SECOND WAIT
9813 035402 012700 000200 MOV #128, R0
9814 035406 000453 BR WATSRT
9815 035410 104413 IWAT1S: SAVREG ;ENTRY FOR 1 SECOND WAIT
9816 035412 012700 000077 MOV #63, R0
9817 035416 000447 BR WATSRT
9818 035420 104413 IWAT159: SAVREG ;ENTRY FOR 160 MS WAIT
9819 035422 012700 000012 MOV #10, R0
9820 035426 000443 BR WATSRT
9821 035430 104413 IWAT144: SAVREG ;ENTRY FOR 144 MS WAIT
9822 035432 012700 000011 MOV #9, R0
9823 035436 000437 BR WATSRT
9824 035440 104413 IWAT128: SAVREG ;ENTRY FOR 128 MS WAIT
9825 035442 012700 000010 MOV #8, R0

```

9826 035446 000433          BR      WATSRT
9827 035450 104413          IWAT112: SAVREG      ;ENTRY FOR 112 MS WAIT
9828 035452 012700 000007  MOV     #7,R0
9829 035456 000427          BR      WATSRT
9830 035460 104413          IWAT96: SAVREG      ;ENTRY FOR 96 MS WAIT
9831 035462 012700 000006  MOV     #6,R0
9832 035466 000423          BR      WATSRT
9833 035470 104413          IWAT80: SAVREG      ;ENTRY FOR 80 MS WAIT
9834 035472 012700 000005  MOV     #5,R0
9835 035476 000417          BR      WATSRT
9836 035500 104413          IWAT64: SAVREG      ;ENTRY FOR 64 MS WAIT
9837 035502 012700 000004  MOV     #4,R0
9838 035506 000413          BR      WATSRT
9839 035510 104413          IWAT48: SAVREG      ;ENTRY FOR 48 MS WAIT
9840 035512 012700 000003  MOV     #3,R0
9841 035516 000407          BR      WATSRT
9842 035520 104413          IWAT32: SAVREG      ;ENTRY FOR 32 MS WAIT
9843 035522 012700 000002  MOV     #2,R0
9844 035526 000403          BR      WATSRT
9845 035530 104413          IWAT16: SAVREG      ;ENTRY FOR 16 MS WAIT
9846 035532 012700 000001  MOV     #1,R0
9847 035536 012746 000000  WATSRT: MOV     #PRO,-(SP) ;LOAD PRIORITY 0 ON STACK
9848 035542 012746 035550  MOV     #5$,-(SP) ;LOAD ADDRESS
9849 035546 000002          RTI
9850
9851 035550 012737 000020 001654 5$:  MOV     #16.,TIMCNT ;PRESET TIME COUNTER
9852 035556 004737 035246  JSR     PC,CLKCAL ;GO CALIBRATE THE CLOCK
9853 035562 005737 001662  1$:  TST     INTSET ;TEST IF INTERRUPT OCCURRED
9854 035566 001036          BNE     3$ ;YES - EXIT
9855 035570 032737 100000 001664  BIT     #LCLKPR,OPTFLG ;TEST IF KW11-L AVAILABLE
9856 035576 001002          BNE     2$ ;YES - SKIP
9857 035600 004737 035314  JSR     PC,MYTIME ;ELSE CALL SIMULATOR
9858 035604 023700 001674  2$:  CMP     LCLKTK,R0 ;TEST IF ENOUGH TICKS COUNTED
9859 035610 103764          BLO     1$ ;NO - LOOP
9860 035612 104420          TGETRK ;ELSE GET '611 REGS
9861 035614 013701 001540  MOV     T.CS1,R1 ;PUT CS1 IN R1- STRIP ALL BUT
9862 035620 042701 177741  BIC     #177741,R1 ;COMMAND CODE; INDEX INTO TABLE
9863 035624 016137 050210 001372  MOV     CMNDLB(R1),DH2N ;AND SELECT HEADER TO IDENTIFY OPERATION
9864 035632 012737 053616 001370  MOV     #EM4,EM2N ;MESSAGE (NO INTERRUPT OR INTERRUPT LATE)
9865 035640 013700 001302  MOV     $TESTN,R0 ;GET NUMBER OF PRESENT TEST
9866 035644 006300          ASL     R0 ;SHIFT FOR INDEX
9867 035646 016037 033122 001264  MOV     $$SW08TB(R0),$ESCAPE ;LOAD ESCAPE TO ABORT TESTS
9868 035654 162737 000002 001264  SUB     #2,$ESCAPE ;BUT GO TO NEXT SCOPE CALL
9869 035662 000402          BR      4$
9870 035664 062716 000002  3$:  ADD     #2,(SP) ;BUMP RETURN AROUND ERROR
9871 035670 104414          4$:  RESREG ;RESTORE REGS
9872 035672 000002          RTI ;RETURN
9873
9874          .SBTTL  "L" REGISTER LOADING ROUTINE
9875          :*  THE PARAMETERS FOLLOWING THE CALL ARE TRANSFERRED INTO
9876          :*  THE "L" REGISTERS L.CS1-L.DCYL. L.MR1 IS NOT
9877          :*  LOADED IN THIS MANNER SINCE IT IS NOT COMMONLY LOADED
9878          :*  FOR AN OPERATION. L.CS2 IS LOADED FROM DRVNUM.
9879          :*
9880          :*  CALL:  JSR     R4,LRLOAD
9881          :*  COMMAND

```



```
9882          : *          WORD COUNT
9883          : *          BUS ADDRESS
9884          : *          .BYTE          SECTOR ADDRESS
9885          : *          .BYTE          TRACK ADDRESS
9886          : *          CYLINDER ADDRESS
9887
9888 035674      LRLOAD:
9889 035674 010046  MOV      R0,-(SP)          ;;PUSH R0 ON STACK.
9890 035676 010146  MOV      R1,-(SP)          ;;PUSH R1 ON STACK.
9891 035700 012701 001600  MOV      #L.CS1,R1          ;;GET ADDRESS OF L REGS
9892 035704 012700 000004  MOV      #4,R0              ;;PRESET COUNT
9893 035710 012421 1$:    MOV      (R4)+,(R1)+        ;;MOVE FIRST FOUR WORDS INTO "L" REGS
9894 035712 005300      DEC      R0                  ;;CS1, WC, BA, DA
9895 035714 001375      BNE     1$
9896 035716 013721 001626  MOV      DRVNUM,(R1)+      ;;LOAD DRIVE NUMBER
9897 035722 005721      TST     (R1)+              ;;BUMP PAST ASOF
9898 035724 012411      MOV      (R4)+,(R1)        ;;LOAD DCYL
9899 035726 012601      MOV      (SP)+,R1         ;;POP STACK INTO R1
9900 035730 012600      MOV      (SP)+,R0         ;;POP STACK INTO R0
9901 035732 000204      RTS     R4
9902
9903          .SBTTL  LOAD RK611 FOR OPERATION
9904          : *          THE REGISTER SETUP STORAGE IS TRANSFERRED TO THE RK611 REGISTER.
9905          : *          THIS IS A STRAIGHT TRANSFER WITH NO CHECKING OR MANIPULATION
9906          : *          OF THE REGISTER CONTENTS. L.CS1 IS TRANSFERRED LAST AS IT
9907          : *          SHOULD BE IF THE GO BIT IS SET.
9908 035734 005037 001662  LOADRK: CLR      INTSET          ;;CLEAR INTERRUPT FLAG.
9909 035740 010046      MOV      R0,-(SP)          ;;STORE REGISTER
9910 035742 010146      MOV      R1,-(SP)
9911 035744 012700 001602  MOV      #L.WC,R0          ;;GET ADDRESS OF SETUP STORAGE WC
9912 035750 010201      MOV      R2,R1            ;;GET BASE ADDRESS
9913 035752 062701 000002  ADD      #2,R1              ;;PUT R1 PAST RKCS1
9914 035756 012021      MOV      (R0)+,(R1)+      ;;LOAD WORD COUNT
9915 035760 012021      MOV      (R0)+,(R1)+      ;;LOAD BUS ADDRESS
9916 035762 012021      MOV      (R0)+,(R1)+      ;;LOAD DISK ADDRESS
9917 035764 012011      MOV      (R0)+,(R1)        ;;LOAD CS2
9918 035766 062701 000006  ADD      #6,R1              ;;BUMP R1 TO ASOF
9919 035772 012021      MOV      (R0)+,(R1)+      ;;LOAD OFFSET
9920 035774 012021      MOV      (R0)+,(R1)+      ;;LOAD CYLINDER
9921 035776 062701 000004  ADD      #4,R1              ;;BUMP R1 TO MR1
9922 036002 011011      MOV      (R0),(R1)         ;;LOAD MR1
9923 036004 123727 001102 000007  CMPB    $TSTNM,#7          ;;SEE IF TEST 7
9924 036012 001403      BEQ     1$                  ;;BR IF YES
9925 036014 053737 001720 001600  BIS     DTYPE,L.CS1         ;;ELSE SET CORRECT DRIVE TYPE
9926 036022 013712 001600 1$:    MOV      L.CS1,(R2)         ;;LOAD CS1
9927 036026 012601      MOV      (SP)+,R1         ;;RESTORE REGISTER
9928 036030 012600      MOV      (SP)+,R0
9929 036032 000002      RTI                       ;;RETURN
9930
9931
9932          : THIS ROUTINE LOADS CONSTANTS EITHER FOR THR RK06 OR THE RK07
9933
9934 036034 005737 001720  LDCON: TST     DTYPE          ;;SEE IF RK06
9935 036040 001414      BEQ     1$                  ;;BR IF YES
9936 036042 012737 001456 036120  MOV      #1456,LSTCYL       ;;ELSE LOAD RK07 CONSTANTS
9937 036050 012737 001000 036122  MOV      #1000,TSTCYL       ;;MAY NOT BE NEEDED
```

```
9938 036056 005037 036130 CLR HOLD2
9939 036062 012737 000040 036132 MOV #DTYE,HOLD3
9940 036070 000207 RTS PC
9941
9942 036072 012737 000632 036120 1$: MOV #632,LSTCYL ;RK06 CONSTANTS
9943 036100 005037 036122 CLR TSTCYL ;MAY NOT BE NEEDED
9944 036104 012737 002000 036130 MOV #CDT,HOLD2
9945 036112 005037 036132 CLR HOLD3
9946 036116 000207 RTS PC
9947
9948 036120 000000 LSTCYL: 0
9949 036122 000000 TSTCYL: 0 ;FOR T20,57,60 MAY NOT BE NEEDED
9950 036124 000000 HOLD: 0 ;FOR T73,105
9951 036126 000000 HOLD1: 0 ;FOR T105
9952 036130 000000 HOLD2: 0 ;FOR T7
9953 036132 000000 HOLD3: 0 ;FOR T12
9954
9955
9956 ;THESE ROUTINES LOAD THE CORRECT DRIVE TYPE INTO DTYPE
9957
9958 036134 006300 GTYP0: ASL R0
9959 036136 016037 001722 001720 MOV TYPTBL(R0),DTYPE
9960 036144 006200 ASR R0
9961 036146 000207 RTS PC
9962
9963 036150 006301 GTYP1: ASL R1
9964 036152 016137 001722 001720 MOV TYPTBL(R1),DTYPE
9965 036160 006201 ASR R1
9966 036162 000207 RTS PC
9967
9968
9969 .SBTTL STORE RK611 REGISTERS
9970 ;* ALL THE RK611 REGISTERS ARE STORED IN THE TEST TABLE T
9971 ;* WITH THE EXCEPTION OF THE DATA BUFFER WHICH IS NOT STORED IN
9972 ;* THIS ROUTINE.
9973
9974 036164 010046 GETRK: MOV R0,-(SP) ;STORE REGISTERS TO BE USED
9975 036166 010146 MOV R1,-(SP)
9976 036170 010346 MOV R3,-(SP)
9977 036172 012700 001540 MOV #T.CS1,R0 ;SET POINTER TO TEST TABLE
9978 036176 010201 MOV R2,R1 ;SET POINTER TO RK611 BASE
9979 036200 012703 000012 MOV #10,R3 ;SET COUNT FOR 1ST 10 REGS
9980 036204 012120 1$: MOV (R1)+,(R0)+ ;STORE RKCS1 THROUGH RKSPAR
9981 036206 005303 DEC R3
9982 036210 001375 BNE 1$
9983 036212 062701 000002 ADD #2,R1 ;BUMP POINTER PAST RKDB
9984 036216 005720 TST (R0)+ ;BUMP POINTER PAST T.RKDB
9985 036220 012703 000004 MOV #4,R3 ;SET COUNT FOR LAST 5 REGS
9986 036224 012120 2$: MOV (R1)+,(R0)+ ;STORE RKMR1 THROUGH RKMR3
9987 036226 005303 DEC R3
9988 036230 001375 BNE 2$
9989 036232 012603 MOV (SP)+,R3 ;RESTORE REGISTERS
9990 036234 012601 MOV (SP)+,R1
9991 036236 012600 MOV (SP)+,R0
9992 036240 000002 RTI ;RETURN
9993 ;SBTTL BIT COUNTER IN A WORD
```

```
9994      :*      THE WORD WHOSE BITS MUST BE COUNTED IS PLACED ON THE STACK
9995      :*      BY THE CALLING ROUTINE.  THE NUMBER OF BITS FOUND IN THE WORD
9996      :*      ARE PASSED BACK ON THE STACK.
9997
9998 036242 016637 000002 001256 BITCNT: MOV      2(SP), $TMP16 ;GET WORD WHOSE BITS ARE TO BE COUNTED
9999 036250 010346          MOV      R3, -(SP) ;STORE R3
10000 036252 005037 001260          CLR      $TMP17 ;CLEAR $TMP16 FOR COUNTING
10001 036256 012703 000021          MOV      #17., R3 ;SET A SHIFT COUNTER
10002 036262 000241          CLC      ;CLEAR CARRY
10003 036264 006037 001256          1$: ROR      $TMP16 ;ROTATE WORD.
10004 036270 103407          BCS      3$ ;WAS BIT SHIFTED OUT A 1?
10005 036272 005303          2$: DEC      R3 ;NO - DEC COUNT
10006 036274 001373          BNE      1$ ;LOOP IF NOT ZERO
10007 036276 012603          MOV      (SP)+, R3 ;RESTORE R3
10008 036300 013766 001260 000002          MOV      $TMP17, 2(SP) ;PUT COUNT OF BITS ON STACK
10009 036306 000204          RTS      R4 ;RETURN
10010 036310 005237 001260          3$: INC      $TMP17 ;BUMP COUNT
10011 036314 000766          BR       2$ ;LOOP
10012
10013      .SBTTL MAINTENANCE CLOCK ROUTINE
10014      :*      THE PARAMETERS PASSED TO THIS ROUTINE ARE LOCATED IN THE
10015      :*      ADDRESS AFTER THE CALL.  THE FIRST BYTE CONTAINS THE NUMBER
10016      :*      OF PHASES ADDRESSES THE CALLING ROUTINE WANTS THE CONTROLLER
10017      :*      CLOCKED THROUGH AND THE SECOND BYTE CONTAINS THE NUMBER OF
10018      :*      CLOCK TRANSITIONS(PARTIAL PHASES) TO BE DONE.
10019
10020      MCLOCK:
10021 036316 010046          MOV      R0, -(SP) ;:PUSH R0 ON STACK
10022 036320 010146          MOV      R1, -(SP) ;:PUSH R1 ON STACK
10023 036322 112400          MOVB     (R4)+, R0 ;GET NUMBER OF CONTROLLER PHASE ADDRESSES
10024 036324 112401          MOVB     (R4)+, R1 ;GET PARTIAL PHASE ADDRESS COUNT
10025
10026 036326 006300          ASL      R0 ;MULTIPLY PHASE ADDRESS COUNT BY 4
10027 036330 006300          ASL      R0
10028 036332 060100          ADD      R1, R0 ;ADD IN PARTIALS
10029 036334 052762 000400 000026 1$: BIS      #MCLK, RKM1(R2) ;SET CLOCK
10030 036342 042762 000400 000026          BIC      #MCLK, RKM1(R2) ;CLEAR MCLK
10031 036350 005300          DEC      R0 ;DECREMENT COUNT
10032 036352 001370          BNE      1$ ;LOOP IF NOT ZERO
10033 036354 012601          MOV      (SP)+, R1 ;:POP STACK INTO R1
10034 036356 012600          MOV      (SP)+, R0 ;:POP STACK INTO R0
10035 036360 000204          RTS      R4
10036
10037      .SBTTL READ AND SORT HEADERS
10038      :*      THE HEADERS IN THE CYLINDER AND TRACK SPECIFIED BY
10039      :*      THE FIELDS IN THE "L" REGISTERS ARE READ AND STORED IN
10040      :*      ASSCENDING ORDER.  CONTROLLER ERRORS ARE CHECKED IN THE
10041      :*      READ HEADER OPERATION AND DATA LATE IS CHECKED AFTER
10042      :*      EACH READ OF THE DATA BUFFER.
10043
10044      CALL: JSR      R4, RDSTHD
10045      TCHKOP ;RETURN POINT IF CERR IN READ HDR
10046      ERROR 4 ;OR 5, 6, 7
10047      ERROR 13 ;RETURN IF DATA LATE IN DB UNLOAD
10048      ERROR 2 ;RETURN IF TO SLOW OR
10049      ;IF HDR 0 NOT FOUND
```

```
10050 036362 104413 RDSTHD: SAVREG
10051 036364 032737 100000 001664 BIT #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
10052 036372 001402 BEQ 20$ ;NO - SKIP
10053 036374 005077 143310 CLR @KWLADD ;RESET INTERRUPT
10054 036400 012700 000026 20$: MOV #26,R0 ;PRESET FOR 26 SECTOR FORMAT
10055 036404 032737 010000 001600 BIT #CFMT,L.CS1 ;IS 24 SECTOR MODE SET?
10056 036412 001402 BEQ 1$ ;NO - SKIP
10057 036414 012700 000024 MOV #24,R0 ;ELSE CHANGE TO 24 SECTOR MODE
10058 036420 012701 070414 1$: MOV #IBUFF,R1 ;SET POINTER TO INPUT BUFFER
10059 036424 010005 MOV R0,R5 ;SAVE NUMBER OF SECTORS
10060 036426 010104 MOV R1,R4 ;SAVE IBUFF ADDRESS
10061 036430 010203 MOV R2,R3 ;SET UP POINTER TO RKDB
10062 036432 062703 000024 ADD #RKDB,R3
10063 036436 013762 001626 000010 MOV DRVNUM,RKCS2(R2) ;LOAD DRIVE NUM
10064 036444 013762 001614 000020 MOV L.DCYL,RKDCYL(R2) ;LOAD CYLINDER NUM
10065 036452 013762 001606 000006 MOV L.DA,RKDA(R2) ;LOAD TRACK AND SECTOR
10066
10067 036460 012737 000020 001654 2$: MOV #16,TIMCNT ;SET TIME COUNTER
10068 036466 005037 001662 CLR INTSET ;CLEAR INTERRUPT FLAG
10069 036472 005037 001674 CLR LCLKTK ;CLEAR TICK COUNTER
10070 036476 053737 001720 001600 BIS DTYPE,L.CS1 ;SET DRV TYP
10071 036504 013762 001600 000000 MOV L.CS1,RKCS1(R2) ;LOAD COMMAND
10072
10073 036512 005737 001662 3$: TST INTSET ;TEST IF INT OCCURRED
10074 036516 001020 BNE 4$ ;YES - SKIP
10075 036520 004737 035314 JSR PC,MYTIME ;WAIT 1 MS
10076 036524 005737 001674 TST LCLKTK ;HAVE WE WAITED 16 MS?
10077 036530 001770 BEQ 3$ ;NO - LOOP ON WAIT
10078
10079 036532 062766 000006 000006 ADD #6,(SP) ;SET RETURN FOR TO SLOW
10080 036540 104420 TGETRK ;GET RK REGS
10081 036542 012737 053616 001370 MOV #EM4,EM2N ;LOAD MESSAGE "TO SLOW/NOT COMPLETE"
10082 036550 012737 050414 001372 MOV #OPER24,DH2N ;LOAD COMMAND "READ HEADER" FOR REPORT
10083 036556 000466 BR 10$ ;SKIP
10084
10085 036560 005762 000000 4$: TST RKCS1(R2) ;TEST FOR CONTROLLER ERROR
10086 036564 100474 BMI 11$ ;YES - SKIP
10087
10088 036566 011324 MOV (R3),(R4)+ ;STORE HEADERS
10089 036570 011324 MOV (R3),(R4)+
10090 036572 011324 MOV (R3),(R4)+
10091
10092 036574 005762 000010 TST RKCS2(R2) ;TEST IF DATA LATE
10093 036600 100443 BMI 8$ ;YES - SKIP
10094
10095 036602 005300 DEC R0 ;DEC SECTOR COUNT
10096 036604 001325 BNE 2$ ;IF NOT ZERO - LOOP
10097
10098 036606 032737 100000 001664 BIT #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
10099 036614 001403 BEQ 5$ ;NO - SKIP
10100 036616 012777 000100 143064 MOV #BIT6,@KWLADD ;SET INTERRUPT ENABLE
10101 036624 032761 000037 000002 5$: BIT #37,2(R1) ;HEADER AT TOP OF BUFF=HEAD 0?
10102 036632 001413 BEQ 6$ ;YES - SKIP
10103 036634 012124 MOV (R1)+,(R4)+ ;ELSE MOV THIS HEADER TO BOTTOM
10104 036636 012124 MOV (R1)+,(R4)+
10105 036640 012124 MOV (R1)+,(R4)+
```

```
10106  
10107 036642 005305  
10108 036644 001367  
10109 036646 012737 056366 001370  
10110 036654 005037 001372  
10111 036660 000421  
10112  
10113 036662 012700 070414 6$: MOV #IBUFF,RO ;GET TOP OF IBUFF  
10114 036666 012120 7$: MOV (R1)+,(RO)+ ;MOVE HEADERS SO THEY START AT TOP OF IBUFF  
10115 036670 012120  
10116 036672 012120  
10117 036674 020004  
10118 036676 001373  
10119  
10120 036700 062766 000010 000006  
10121 036706 000423  
10122  
10123 036710 012737 054631 001500 8$: MOV #EMDLT,EM13N ;'DATA LATE SET RESULT OF READ DB'  
10124 036716 012737 056777 061112  
10125 036724 062766 000004 000006 9$: ADD #4,6(SP) ;SET ERROR RETURN  
10126 036732 104420  
10127 036734 013700 001302 10$: TGETRK ;GET RK REGS  
10128 036740 006300  
10129 036742 016037 033122 001264  
10130 036750 162737 000002 001264  
10131  
10132 036756 104414 11$: MOV $TESTN,RO ;GET TEST NUMBER  
10133 036760 000204  
10134  
10135  
10136  
10137  
10138  
10139  
10140  
10141  
10142  
10143  
10144  
10145 036762 104413  
10146 036764 012762 100000 000000  
10147 036772 012700 001202  
10148 036776 012701 000010  
10149 037002 012720 177777  
10150 037006 005301  
10151 037010 001374  
10152 037012 012700 001206  
10153 037016 012701 000001  
10154 037022 005037 001230
```

DEC R5 ;TEST FO INSURE HEAD 0 IS FOUND
BNE 5\$;IF ALL HEADERS NOT CHECKED - LOOP
MOV #EM56,EM2N ;ELSE "HEADER 0 NOT FOUND" MESSAGE
CLR DH2N
BR 9\$;SKIP
6\$: MOV #IBUFF,RO ;GET TOP OF IBUFF
7\$: MOV (R1)+,(RO)+ ;MOVE HEADERS SO THEY START AT TOP OF IBUFF
MOV (R1)+,(RO)+
MOV (R1)+,(RO)+
CMP RO,R4 ;ALL HEADERS MOVED?
BNE 7\$;NO - LOOP
ADD #10,6(SP) ;SET UP FOR GOOD RETURN
BR 11\$
8\$: MOV #EMDLT,EM13N ;'DATA LATE SET RESULT OF READ DB'
MOV #EMRDB,DF011A
9\$: ADD #4,6(SP) ;SET ERROR RETURN
TGETRK ;GET RK REGS
10\$: MOV \$TESTN,RO ;GET TEST NUMBER
ASL RO ;SHIFT FOR INDEX
MOV \$SW08TB(RO),\$ESCAPE ;SET ESCAPE
SUB #2,\$ESCAPE ;TO NEXT SCOPE CALL
11\$: RESREG
RTS R4
;SBTTL GET DRIVE STATUS
;* THIS ROUTINE GETS ALL THE DRIVE STATUS AND PLACES IT IN \$REG10
;* THROUGH \$REG17. THESE REGISTORS ARE FIRST CLEARED TO ALL ONES AND
;* THEN IF ERROR OCCURS WHILE GETTING STATUS, THE 1'S ARE LEFT
;* IN THE REGISTERS.
;*CALL: JSR R4,GETDRS
;* BR ERROR PROCESSING ERROR RETURN
;* BR NO ERROR PROCESSING GOOD RETURN
GETDRS: SAVREG
MOV #CCLR,RKCS1(R2) ;CLEAR ANY OLD ERRORS LAYING AROUND
MOV #\$REG10,RO ;PRESET ALL STATUS STORAGE TO
MOV #8,R1 ;ALL ONES
1\$: MOV #177777,(RO)+
DEC R1
BNE 1\$
MOV #\$REG12,RO ;SET POINTER TO REG12 FOR A01 & B01
MOV #1,R1 ;PRESET FOR PAIR ONE.
CLR \$TMP3 ;CLEAR ERROR SWITCH

```
10155 037026 013762 001610 000010 2$: MOV L,CS2,RKCS2(R2) ;LOAD DRIVE #
10156 037034 010162 000026 MOV R1,RKMR1(R2) ;LOAD MR1
10157 037040 012762 000001 000000 MOV #BIT0,RKCS1(R2) ;DO SELECT
10158 037046 012703 000050 MOV #40.,R3 ;WAIT FOR A FEW MICRO RECORDS TO
10159 037052 005303 3$: DEC R3 ;BIT SELECT FINISH
10160 037054 001376 BNE 3$
10161 037056 032762 100000 000000 BIT #CERR,RKCS1(R2) ;ANY ERROR SET AS A RESULT OF SELECT?
10162 037064 001415 BEQ 4$ ;NO - SKIP
10163 037066 032762 024000 000000 BIT #CTO!SPAR,RKCS1(R2) ;CHECK IF TIMEOUT OR PARITY ERROR
10164 037074 001004 BNE 8$ ;YES - SKIP
10165 037076 032762 037400 000010 BIT #37400,RKCS2(R2) ;TEST FOR ERRORS:
10166 ; NED!UPE!MDS!UFE!NEM!PGE
10167 037104 001405 BEQ 4$ ;NO - SKIP
10168 037106 012737 000001 001230 8$: MOV #1,$TMP3 ;SET ERROR FLAG
10169 037114 022020 CMP (R0)+,(R0)+ ;BUMP TO LET THAT PAIR STAY ALL 1'S.
10170 037116 000404 BR 5$ ;SKIP
10171 037120 016220 000034 4$: MOV RKMR2(R2),(R0)+ ;STORE A WORD
10172 037124 016220 000036 MOV RKMR3(R2),(R0)+ ;STORE B WORD
10173 037130 012762 100000 000000 5$: MOV #CCLR,RKCS1(R2) ;CLEAR ANY OLD ERROR IN CONTROLLER
10174 037136 005701 TST R1 ;IS R1 A 0 (LAST TRANSFER, PAIR 0)
10175 037140 001410 BEQ 6$ ;YES - SKIP
10176 037142 005201 INC R1 ;ELSE BUMP TO NEXT PAIR
10177 037144 022701 000004 CMP #4,R1 ;PAIR 3 JUST STORED?
10178 037150 001326 BNE 2$ ;NO - SKIP
10179 037152 005001 CLR R1 ;ELSE SET TO PAIR 0
10180 037154 012700 001202 MOV #$REG10,R0 ;PRESET POINTER FOR PAIR 0
10181 037160 000722 BR 2$ ;GO GET THEM
10182 037162 104414 6$: RESREG ;EXIT HERE
10183 037164 005737 001230 TST $TMP3 ;ANY ERROR IN STATUS GETTING
10184 037170 001001 BNE 7$ ;YES - SKIP
10185 037172 005724 TST (R4)+ ;ELSE BUMP PART ERROR
10186 037174 000204 7$: RTS R4 ;RETURN
10187
10188 ;SBTTL SUBSYSTEM INITIALIZE AND INITIALIZE STATE TEST
10189 ;* THE SUBSYSTEM IS INITIALIZED WITH A SUBSYSTEM CLEAR
10190 ;* COMMAND. CERR AND DI ARE MONITORED FOR A SHORT
10191 ;* PERIOD OF TIME DURING WHICH THEY SHOULD BOTH RESET.
10192 ;*
10193 ;* IF THEY DO RESET, READY IS TESTED TO INSURE IF SETS.
10194 ;*
10195 ;* IF ANY OF THESE THREE CONDITIONS ARE NOT MET AN APPROPRIATE
10196 ;* ERROR MESSAGE IS PREPARED AND REPORTED WHEN THE ROUTINE
10197 ;* RETURN TO THE CALL. IF EVERY THING IS GOOD, THE RETURN
10198 ;* SKIPS OVER THE ERROR CALL AND TEST ABORT.
10199 ;*
10200 ;* THE USUAL CALL TO THIS ROUTINE WILL BE FOLLOWED BY
10201 ;* AN ERROR MESSAGE AND BRANCH TO END OF TEST. THIS
10202 ;* IS DONE BECAUSE FAILURE TO INITIALIZE CORRECTLY IS FATAL TO
10203 ;* THE TEST.
10204
10205 SSINIT:
10206 037176 MOV R0,-(SP) ;:PUSH R0 ON STACK
10207 037176 010046 MOV R1,-(SP) ;:PUSH R1 ON STACK
10208 037200 010146 MOV #7,R1 ;:SET CLEAR COUNT
10209 037202 012701 000007 MOV #L,CS1,R0 ;:GET ADDRESS OF 'L' REGS
10210 037212 012720 000100 MOV #100,(R0)+ ;:PRESET CS1 INTR ENABLE
```

```

10211 037216 005020          7$: CLR (R0)+ ;CLEAR THE NEXT
10212 037220 005301          DEC R1 ;COUNT 0?
10213 037222 001375          BNE 7$ ;NO - LOOP
10214 037224 012762 0C0040 000010 MOV #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
10215 037232 012737 000012 001222 MOV #10, $TMP0 ;SET A COUNTER
10216 037240 016237 000000 001540 1$: MOV RKCS1(R2),T.CS1 ;GET CS1
10217 037246 032737 140000 001540 BIT #CERR!DI,T.CS1 ;TEST IF ERROR OR DI SET
10218 037254 001433          BEQ 2$ ;NO - SKIP TO READY TEST
10219 037256 005337 001222          DEC $TMP0 ;ELSE DECREMENT COUNTER
10220 037262 001366          BNE 1$ ;AND LOOP
10221 037264 032737 100000 001540 BIT #CERR,T.CS1 ;TEST - IS IT CERR STILL SET
10222 037272 001404          BEQ 3$ ;NO - SKIP TO DI MESSAGE
10223 037274 012737 053711 001400 MOV #EM5,EM3N ;MESSAGE (SUBSYS CLR NOT RESET ERROR)
10224 037302 000403          BR 4$
10225 037304 012737 053755 001400 3$: MOV #EM6,EM3N ;MESSAGE (SUBSYS CLEAR NOT RESET DI)
10226 037312 104420          4$: TGETRK
10227 037314 013700 001302          MOV $TESTN,R0 ;GET PRESENT TEST NUMBER
10228 037320 006300          ASL R0 ;SHIFT FOR INDEX
10229 037322 016037 033122 001264 MOV $$SW08TBL(R0), $ESCAPE ;LOAD ESCAPE TO ABORT TEST
10230 037330 162737 000002 001264 SUB #2, $ESCAPE ;SET TO NEXT SCOPE CALL
10231 037336 012601          MOV (SP)+,R1 ;POP STACK INTO R1
10232 037340 012600          MOV (SP)+,R0 ;POP STACK INTO R0
10233 037342 000414          BR 6$ ;SKIP TO EXIT
10234 037344 032737 000200 001540 2$: BIT #RDY,T.CS1 ;TEST READY SET
10235 037352 001004          BNE 5$ ;YES - GOOD EXIT
10236 037354 012737 054034 001400 MOV #EM7,EM3N ;MESSAGE (SUBSYS CLR NOT SET READY)
10237 037362 000753          BR 4$
10238 037364 012601          5$: MOV (SP)+,R1 ;RESTORE REGS
10239 037366 012600          MOV (SP)+,R0
10240 037370 062716 000002          ADD #2,(SP) ;GOOD RETURN
10241 037374 000002          6$: RTI
10242
10243 .SBTTL WORD COUNT AT END OF OPERATION CHECK
10244 :*
10245 :* THIS ROUTINE COMPARES THE CONTENTS OF THE TEST STORAGE FOR
10246 :* THE WORD COUNT AGAINST THE SUPPLIED VALUE. IF UNEQUAL, THE
10247 :* ERROR FLAG (WCERR) IS SET IN GROUP 4 ERROR FLAGS (GRP4ER)
10248 :*
10249 :*CALL: JSR R4,CHKWC
10250 :* .WORD ;EXPECTED WC VALUE
10251 037376 012437 050034          CHKWC: MOV (R4)+,EXPWC ;STORE EXPECTED VALUE
10252
10253 037402 023737 050034 001542          CMP EXPWC,T.WC ;COMPARE
10254 037410 001406          BEQ 1$ ;EQUAL - SKIP
10255 037412 052737 000001 050072          BIS #WCERR,GRP4ER ;SET ERROR FLAG
10256 037420 013737 001542 050050          MOV T.WC,REALWC ;STORE REAL WORD COUNT
10257 037426 000204          1$: RTS R4 ;RETURN.
10258
10259 .SBTTL BUS ADDRESS AT END OF OPERATION CHECK
10260 :*
10261 :* THIS ROUTINE COMPUTES THE EXPECTED BUS ADDRESS AT THE END OF
10262 :* A TRANSFER BY USING THE INITIAL BUS ADDRESS, ADDING IN THE
10263 :* INITIAL WORD COUNT, AND SUBTRACTING ANY RESIDUAL WORD COUNT.
10264 :* IF THIS COMPUTED BA DOES NOT EQUAL THE CONTENTS OF RKBA
10265 :* AN ERROR FLAG (BAERR) IS SET IN GROUP 4 ERROR FIELD (GRP4ER)
10266 :*
10267 :* IF BUS ADDRESS INCREMENT INHIBIT WAS SET, THE EXPECTED BUS

```

```
10267          :* ADDRESS IS THE STARTING BUS ADDRESS.
10268          :*CALL: JSR      R4,CHKBA
10269
10270 037430 010046      CHKBA: MOV      R0,-(SP)
10271 037432 010146      MOV      R1,-(SP)
10272 037434 010346      MOV      R3,-(SP)
10273 037436 032737 000020 001610      BIT      #BAI,L.CS2      ;TEST IF BAI SET
10274 037444 001404      BEQ      4$              ;NO - SKIP
10275 037446 013737 001604 050040      MOV      L.BA,EXPBA      ;STORE EXPECTED BA (SAME AS STARTING BA)
10276 037454 000441      BR       3$
10277 037456 013700 001602      4$:  MOV      L.WC,R0      ;GET INITIAL WORD COUNT
10278 037462 005400      NEG      R0
10279 037464 113703 001601      MOVB     L.CS1+1,R3      ;GET BA16 & BA17
10280 037470 042703 177774      BIC      #177774,R3      ;CLEAR UNWANTED BITS
10281
10282 037474 005700      TST      R0              ;TEST IF INITIAL WORD COUNT 0
10283 037476 001003      BNE     6$              ;NO - SKIP
10284 037500 062703 000002      ADD      #2,R3          ;ADD 2 TO BA16,17 (65K WORD XFER)
10285 037504 000407      BR       9$              ;SKIP
10286 037506 005700      6$:  TST      R0              ;TEST IF INITIAL WC BIT 15 SET
10287 037510 100001      BPL     5$              ;NO - SKIP
10288 037512 005203      INC      R3              ;BUMP BA16,17 (32K WORD XFER)
10289 037514 006300      5$:  ASL      R0              ;SHIFT WORD COUNT TO MAKE MEM ADD CNT
10290 037516 063700 001604      ADD      L.BA,R0          ;ADD IN START BUFFER ADD
10291 037522 005503      ADC      R3              ;IF CARRY - ADD INTO BA16,17
10292 037524 013701 001542      9$:  MOV      T.WC,R1          ;GET END OF OPERATION WORD COUNT
10293 037530 001411      BEQ      1$              ;BRANCH IF ZERO
10294 037532 005401      NEG      R1
10295 037534 005701      TST      R1              ;TEST END OPERATION WC BIT 15 SET
10296 037536 100001      BPL     7$              ;NO - SKIP
10297 037540 005303      DEC      R3              ;DEC BA 16,17 (32K WC LEFT)
10298 037542 006301      7$:  ASL      R1              ;SHIFT WC TO MAKE MEM ADD CNT
10299 037544 160100      SUB      R1,R0           ;SUB FROM COMPUTED BUS ADDRESS
10300 037546 005603      SBC      R3              ;SUB CARRY FROM BA16,17
10301 037550 010337 050036      MOV      R3,EXPUBA      ;STORE EXPECTED UPPER BA BITS
10302 037554 010037 050040      1$:  MOV      R0,EXPBA
10303 037560 020037 001544      3$:  CMP      R0,T.BA          ;EQUAL TO COMPUTED?
10304 037564 001406      BEQ      2$              ;YES - SKIP
10305 037566 052737 000004 050072      BIS      #BAERR,GRP4ER   ;ELSE SET BAERR FLAG
10306 037574 013737 001544 050054      MOV      T.BA,REALBA     ;STORE REAL BUS ADDRESS
10307 037602 113703 001541      2$:  MOVB     T.CS1+1,R3      ;GET REAL UPPER BA
10308 037606 042703 177774      BIC      #177774,R3      ;CLEAR UNWANTED BITS
10309 037612 020337 050036      CMP      R3,EXPUBA      ;CHECK IF EQUAL
10310 037616 001405      BEQ      8$              ;YES - SKIP
10311 037620 052737 000002 050072      BIS      #UBAERR,GRP4ER  ;ELSE SET UBA ERROR
10312 037626 010337 050052      MOV      R3,REALUB      ;STORE REAL UPPER BA
10313 037632 012603      8$:  MOV      (SP)+,R3
10314 037634 012601      MOV      (SP)+,R1
10315 037636 012600      MOV      (SP)+,R0
10316 037640 000204      RTS      R4
10317
10318          .SBTTL CYLINDER,TRACK,SECTOR TEST AT END OF OPERATION
10319          :* THIS ROUTINE CHECKS THAT THE CONTENTS OF THE RKDCYL AND RKDA
10320          :* ARE CORRECT FOR ANY SIZE DATA TRANSFER AT THE END OF THE
10321          :* OPERATION. THE CONTENTS OF THE LOAD REGISTER STORAGE ARE
10322          :* COUNTED ON TO HAVE THE INITIAL VALUES TO MAKE THE
```


10323
10324
10325
10326
10327
10328
10329
10330
10331
10332 037642 104413
10333 037644 013700 001602
10334 037650 005400
10335 037652 013701 001542
10336 037656 001401
10337 037660 005401
10338 037662 160100
10339 037664 005001
10340
10341
10342
10343
10344 037666 022700 000400
10345 037672 003004
10346 037674 005201
10347 037676 162700 000400
10348 037702 000771
10349 037704 005700
10350 037706 001401
10351 037710 005201
10352
10353
10354
10355 037712 012703 000026
10356 037716 032737 010000 001600
10357 037724 001402
10358 037726 012703 000024
10359
10360
10361 037732 013737 001614 050042
10362 037740 113704 001607
10363 037744 042704 177400
10364 037750 113705 001606
10365 037754 042705 177400
10366 037760 005301
10367 037762 005205
10368 037764 020503
10369 037766 001010
10370 037770 005005
10371 037772 005204
10372 037774 022704 000003
10373 040000 001003
10374 040002 005004
10375 040004 005237 050042
10376 040010 005301
10377 040012 001363
10378 040014 010437 050046

```

: *      NECESSARY CALCULATION.
: *
: *      ALL THREE VALUES ARE GENERATED AND STORED IN EXPECTED VALUES
: *      STORAGE EXPCYL, EXPTRK, EXPSEC. ALL 3 ARE CHECKED AND
: *      IF ONE OR MORE ARE WRONG, THE CORRESPONDING BIT IN THE
: *      ERROR FLAGS FIELD (GRP4ER) IS SET.
: *
: *CALL: JSR      R4,CHKCTS
CHKCTS: SAVREG
MOV      L.WC,R0      ;GET SPECIFIED WORD COUNT
NEG      R0           ;NEGATE IT
MOV      T.WC,R1     ;GET END OF OPERATION WORD COUNT
BEQ      10$         ;IF ZERO - SKIP
NEG      R1           ;NEGATE IT
10$: SUB      R1,R0    ;COMPUTE ACTUAL WORDS TRANSFERRED
CLR      R1           ;CLEAR R1 FOR COUNTING
:
: THE FOLLOWING CODE DETERMINES HOW MANY SECTORS OF DATA HAS BEEN
: TRANSFERRED IN THE OPERATION. ONCE IT HAS COMPUTED THAT, THE
: END OF OPERATION VALUES FOR THE CYLINDER, TRACK, AND SECTOR
: IS CALCULATED.
1$: CMP      #400,R0
BGT      2$
INC      R1
SUB      #400,R0
BR       1$
2$: TST      R0
BEQ      3$
INC      R1
:
: AT THIS POINT R1 HAS A COUNT OF THE
: NUMBER OF FULL SECTOR TRANSFER + 1 IF A
: PARTIAL SECTOR WAS TRANSFERRED.
3$: MOV      #26,R3
BIT      #CFMT,L.CS1
BEQ      4$
MOV      #24,R3
:
: R3 HAS BEEN SET UP WTH THE NUMBER
: OF SECTORS IN A TRACK FOR THE FORMAT USED
4$: MOV      L.DCYL,EXPCYL ;GET STARTING VALUES FOR CYLINDER
MOVB     L.DT,R4          ;TRACK
BIC      #177400,R4
MOVB     L.DS,R5          ;SECTOR
BIC      #177400,R5
5$: DEC      R1           ;ADJUST COUNT FOR ZERO DETECT
INC      R5             ;BUMP SECTOR COUNT
CMP      R5,R3         ;DID THIS MAKE SECTOR COUNT > 1 TRACK?
BNE      6$           ;NO - SKIP
CLR      R5             ;ELSE CLEAR SECTOR COUNT
INC      R4             ;BUMP TRACK COUNT
CMP      #3,R4         ;DID THIS MAKE TRK COUNT > 1 CYLINDER?
BNE      6$           ;NO - SKIP
CLR      R4             ;ELSE CLEAR TRACK COUNT
INC      EXPCYL        ;BUMP CYLINDER COUNT
6$: DEC      R1           ;DEC COUNT
BNE      5$           ;IF ZERO - EXIT
MOV      R4,EXPTRK     ;STORE EXPECTED TRACK
```

10379	040020	010537	050044		MOV	R5,EXPSEC	:STORE EXPECTED SECTOR (CYL ALREADY SLOW)
10380	040024	023737	001560	050042	CMP	T.DCYL,EXPCYL	:TEST IF CYLINDER OK
10381	040032	001403			BEQ	7\$:YES - SKIP
10382	040034	052737	000010	050072	BIS	#CYLERR,GRP4ER	:NO - SET ERROR FLAG
10383	040042	120437	001547	7\$:	CMPB	R4,T.DA+1	:TEST TRACK OK
10384	040046	001403			BEQ	8\$:YES - SKIP
10385	040050	052737	000020	050072	BIS	#TRKERR,GRP4ER	:NO - SET ERROR FLAG
10386	040056	120537	001546	8\$:	CMPB	R5,T.DA	:TEST SECTOR COUNT OK
10387	040062	001403			BEQ	9\$:YES - SKIP
10388	040064	052737	000040	050072	BIS	#SECERR,GRP4ER	:USE SET ERROR FLAG
10389	040072	012700	050050	9\$:	MOV	#REALWC,RO	
10390	040076	013720	001542		MOV	T.WC,(R0)+	:STORE REAL WORD COUNT
10391	040102	013720	001544		MOV	T.BA,(R0)+	:STORE REAL BUS ADDRESS
10392	040106	013720	001560		MOV	T.DCYL,(R0)+	:STORE REAL CYLINDER ADDRESS
10393	040112	113710	001547		MOVB	T.DA+1,(R0)	:STORE REAL TRACK ADDRESS
10394	040116	005720			TST	(R0)+	
10395	040120	113710	001546		MOVB	T.DA,(R0)	:STORE REAL SECTOR ADDRESS
10396	040124	104414			RESREG		
10397	040126	000204			RTS	R4	

10398
10399
10400
10401
10402
10403
10404
10405
10406
10407
10408
10409
10410
10411
10412

```
.SBTTL OPERATION CHECK ROUTINE
:* THIS IS WHERE ALL HARDWARE ERROR INDICATORS AND SOME SOFTWARE
:* ERRORS ARE CHECKED. THE GENERAL PROCEDURE FLOW IS AS FOLLOWS:
:* THE ROUTINE IS CALLED WITH A TRAP (TCHKOP). THE LOCATION
:* FOLLOWING THE TRAP CALL WILL HAVE AN ERROR TRAP WHICH
:* THE ROUTINE WILL BYPASS IF NO ERROR IS FOUND. IF AN
:* ERROR IS DETECTED, THE ERROR TRAP CALL IS MODIFIED
:* BY THIS ROUTINE SUCH THAT THE ERROR TABLE ITEM WILL
:* BE THE PROPER ITEM FOR THE FORMAT REQUIRED BY THIS
:* ERROR. THE ERROR TRAP WILL BE MADE EITHER ERROR 4,5,6,
:* 7, OR 10. REFER TO THE ERROR ITEM TABLE FOR A DESCRIPTION
:* OF THE FORMAT AND WHICH ERRORS ARE DISPLAYED IN WHAT
:* FORMAT.
```

10413	040130	104413		
10414	040132	011600		
10415	040134	012037	001242	
10416	040140	012037	001244	
10417	040144	012037	001246	
10418	040150	010016		
10419	040152	012737	177777	001250
10420	040160	000403		
10421				
10422	040162	104413		
10423	040164	005037	001250	
10424				
10425	040170	104420		
10426	040172	005037	050064	
10427	040176	005037	050066	
10428	040202	005037	050070	
10429	040206	005037	050072	
10430	040212	005037	050206	
10431	040216	032737	024000	001540
10432	040224	001111		
10433	040226	032737	177400	001550
10434	040234	001105		

```
CHKWE: SAVREG
MOV (SP),RO :GET POINTER TO ERROR WORDS
MOV (R0)+,$TMP10 :STORE EXPECTED ERROR GROUP 1
MOV (R0)+,$TMP11 :STORE EXPECTED ERROR GROUP 2
MOV (R0)+,$TMP12 :STORE EXPECTED ERROR GROUP 3
MOV RO,(SP) :STORE RETURN
MOV #-1,$TMP13 :SET FLAG - EXPECTED ERROR
BR CHKST

CHKOP: SAVREG
CLR $TMP13 :RESET EXPECTED ERROR FLAG

CHKST: TGETRK :GET 611 REGS IO TRAP
CLR GRP1ER :CLEAR ERROR FLAGS
CLR GRP2ER
CLR GRP3ER
CLR GRP4ER
CLR GPSUMF :CLEAR SUMMARY FLAGS
BIT #CS1ERBIT,T.CS1 :TEST IF ERROR SET IN CS1
BNE 4$ :YES - SKIP
BIT #CS2ERBIT,T.CS2 :TEST IF ERROR SET IN CS2
BNE 4$ :YES - SKIP
```

```
10435 040236 032737 000070 001552 BIT #DSERBIT,T.DS ;TEST IF ERROR SET IN DS
10436 040244 001101 BNE 4$ ;YES - SKIP
10437 040246 005737 001554 TST T.ER ;TEST IF ERROR SET IN ER
10438 040252 001076 BNE 4$ ;YES - SKIP
10439 040254 032737 100000 001540 BIT #CERR,T.CS1 ;COMBINED ERROR SET?
10440 040262 001405 BEQ 9$ ;NO - SKIP
10441 040264 052737 100000 050064 BIS #CERNER,GRP1ER ;SET ERROR FLAG IN GROUP 1
10442 040272 000137 041022 JMP 25$ ;SKIP
10443
10444 : CODE TO CHECK WORD COUNT, BUFFER ADDRESS, CYLINDER, TRACK,
10445 : AND SECTOR AT THE END OF THE OPERATION. THIS IS DONE ONLY
10446 : IF CERR NOT SET BY THE OPERATION.
10447 :
10448 : ALL OF THE ABOVE CONDITIONS ARE CHECKED AND A BIT SET FOR
10449 : EACH CHECK THAT FAILS. HOWEVER, ONLY ONE ERROR IS REPORTED.
10450 : THE ORDER OF PRIORITY FOR REPORTING THE ERROR IS THE ORDER
10451 : LISTED ABOVE.
10452
10453 040276 005737 001250 9$: TST $TMP13 ;TEST IF ERROR EXPECTED
10454 040302 001402 BEQ 62$ ;NO - SKIP
10455 040304 000137 041022 JMP 25$ ;YES - JUMP
10456 040310 013700 001540 62$: MOV T.CS1,R0 ;GET CS1
10457 040314 042700 177741 BIC #177741,R0 ;CHECK IF OPERATION IS READ DATA,
10458 040320 022700 000020 CMP #20,R0 ;WRITE DATA, OR WRITE CHECK. IF
10459 040324 002445 BLT 3$ ;NOT, SKIP ALL CHECKING IN GROUP
10460 040326 022700 000030 CMP #30,R0 ;FOUR
10461 040332 003042 BGT 3$
10462 040334 004437 037376 JSR R4,CHKWC ;CHECK WORD COUNT
10463 040340 000000 .WORD 0 ;EXPECTED WORD COUNT
10464 040342 004437 037430 JSR R4,CHKBA ;CHECK BUS ADDRESS
10465 040346 004437 037642 JSR R4,CHKCTS ;CHECK CYL, TRACK, & SECTOR
10466 040352 005737 050072 TST GRP4ER ;ANY GROUP 4 ERRORS?
10467 040356 001430 BEQ 3$ ;NO - SKIP
10468 040360 016037 050210 061072 MOV CMNDLB(R0),DF010A ;LOAD ADDRESS OF COMMAND MESSAGE
10469 040366 013700 050072 MOV GRP4ER,R0 ;PUT GROUP 4 ERROR FLAG IN R0
10470 040372 005001 CLR R1 ;CLEAR R1 FOR INDEX COUNTER
10471 040374 006200 1$: ASR R0 ;SHIFT FLAGS - FIRST ONE ON RIGHT IS ERROR TO
10472 040376 103402 BCS 2$ ;BE REPORTED, REST ARE IGNORED
10473 040400 005720 TST (R0)+ ;WHEN AN ERROR BIT IS FOUND,
10474 040402 000774 BR 1$ ;GET THE ERROR MESSAGE ASSOCIATED
10475 040404 016037 050074 001450 2$: MOV GRP4MS(R0),EM10N ;WITH IT AND SET ERROR TABLE ITEM TO
10476 040412 016037 050034 001202 MOV EXPWC(R0),$REG10 ;POINT TO THE MESSAGE. LOAD REG10 & 11
10477 040420 016037 050050 001204 MOV REALWC(R0),$REG11 ;WITH EXPECTED & IS VALUES
10478 040426 104414 RESREG ;RESTORE REGISTER
10479 040430 012776 000010 000000 MOV #10,@(SP) ;MAKE THE ERROR CALL POINT TO THE
10480 040436 000002 RTI ;RIGHT TABLE ENTRY & RETURN.
10481 040440 3$:
10482 040440 104414 RESREG
10483 040442 062716 000002 ADD #2,(SP) ;BUMP RETURN PAST ERROR
10484 040446 000002 RTI ;RETURN
10485
10486 ; THE FOLLOWING CODE BUILDS THE GROUP 1,2, & 3 ERROR WORDS.
10487
10488 040450 012700 050064 4$: MOV #GRP1ER,R0 ;SET UP GENERAL REGISTER AS POINTER
10489 040454 012701 001540 MOV #T.CS1,R1 ;CS1
10490 040460 012703 001550 MOV #T.CS2,R3 ;CS2
```

10491	040464	012704	001552		MOV	#T.DS,R4	:DS
10492	040470	012705	001554		MOV	#T.ER,R5	:AND ER
10493							
10494	040474	051510			BIS	(R5),(R0)	:SET ALL BITS IN GRP1ER THAT :CORRESPOND TO ERROR BITS IN RKER
10495							
10496	040476	042710	120701		BIC	#ILF!ECH!BSE!HVRC!OPI!DCK,(R0)	:CLEAR ALL THAT DON'T BELONG GRP1
10497							
10498	040502	032711	020000		BIT	#SPAR,(R1)	:TEST IF SPAR SET
10499	040506	001402			BEQ	5\$:NO - SKIP
10500	040510	052710	000001		BIS	#SPARERR,(R0)	:SET SPAR ERROR FLAG
10501							
10502	040514	032714	000010	5\$:	BIT	#ACLO,(R4)	:TEST ACLO SET
10503	040520	001402			BEQ	6\$:NO - SKIP
10504	040522	052710	000100		BIS	#ACLOERR,(R0)	:SET ACLO ERROR FLAG
10505							
10506	040526	032714	000020	6\$:	BIT	#SPDLSS,(R4)	:TEST SPEED LOSS SET
10507	040532	001402			BEQ	7\$:NO - SKIP
10508	040534	052710	000200		BIS	#SPDERR,(R0)	:SET SPEED LOSS ERROR FLAG
10509							
10510	040540	032714	000040	7\$:	BIT	#DROT,(R4)	:TEST IF DROT SET
10511	040544	001402			BEQ	8\$:NO - SKIP
10512	040546	052710	000400		BIS	#DROTERR,(R0)	:SET DROT ERROR FLAG
10513							
10514	040552	032711	100000	8\$:	BIT	#CERR,(R1)	:TEST CERR ITSELF SET
10515	040556	001002			BNE	10\$:YES - SKIP
10516	040560	032710	020000		BIT	#NCERWE,(R0)	:SET NO CERR WITH ERROR FLAG
10517							
10518	040564	012700	050066	10\$:	MOV	#GRP2ER,R0	:SET POINTER TO GROUP 2 ERROR FLAGS
10519							
10520	040570	032715	000100		BIT	#ECH,(R5)	:TEST IF ECH SET
10521	040574	001402			BEQ	11\$:NO - SKIP
10522	040576	052710	000001		BIS	#ECHERR,(R0)	:SET ECH FLAG
10523							
10524	040602	032715	100000	11\$:	BIT	#DCK,(R5)	:TEST DCK SET
10525	040606	001402			BEQ	12\$:NO - SKIP
10526	040610	052710	000002		BIS	#DCKERR,(R0)	:SET DCK ERROR FLAG.
10527							
10528	040614	032713	040000	12\$:	BIT	#WCE,(R3)	:TEST WRITE CHECK ERROR
10529	040620	001402			BEQ	120\$:NO - SKIP
10530	040622	052710	000004		BIS	#WCKERR,(R0)	:SET WCE BIT
10531	040626	032713	100000	120\$:	BIT	#DLT,(R3)	:TEST DATA LATE
10532	040632	001402			BEQ	13\$:NO - SKIP
10533	040634	052710	000010		BIS	#DLTERR,(R0)	:SET DLT ERROR FLAG
10534							
10535	040640	032715	020000	13\$:	BIT	#OPI,(R5)	:TEST OPI SET
10536	040644	001402			BEQ	14\$:NO - SKIP
10537	040646	052710	000020		BIS	#OPIERR,(R0)	:SET OPI ERROR FLAG
10538							
10539	040652	032715	000400	14\$:	BIT	#HVRC,(R5)	:TEST HVRC SET
10540	040656	001402			BEQ	16\$:NO - SKIP
10541	040660	052710	000040		BIS	#HVRCERR,(R0)	:SET HVRC FLAG
10542							
10543	040664	032715	000200	16\$:	BIT	#BSE,(R5)	:TEST BSE ERROR FLAG
10544	040670	001402			BEQ	17\$:NO - SKIP
10545	040672	052710	000100		BIS	#BSERR,(R0)	:SET BSE FLAG
10546							

10547	040676	012700	050070	17\$:	MOV	#GRP3ER,R0	;SET POINTER TO GROUP 3 FLAGS
10548							
10549	040702	032713	010000		BIT	#NED,(R3)	;TEST NED SET
10550	040706	001402			BEQ	18\$;NO - SKIP
10551	040710	052710	000001		BIS	#NEDERR,(R0)	;SET NED FLAG
10552							
10553	040714	032711	004000	18\$:	BIT	#CTO,(R1)	;TEST CTO SET
10554	040720	001402			BEQ	19\$;NO - SKIP
10555	040722	052710	000002		BIS	#CTOERR,(R0)	;SET CTO FLAG
10556							
10557	040726	032713	000400	19\$:	BIT	#UFE,(R3)	;TEST UFE SET
10558	040732	001402			BEQ	20\$;NO - SKIP
10559	040734	052710	000004		BIS	#UFERR,(R0)	;SET UFE FLAG
10560							
10561	040740	032713	001000	20\$:	BIT	#MDS,(R3)	;TEST MDS SET
10562	040744	001402			BEQ	21\$;NO - SKIP
10563	040746	052710	000010		BIS	#MDSERR,(R0)	;SET MDE FLAG
10564							
10565	040752	032713	002000	21\$:	BIT	#PGE,(R3)	;TEST PGE SET
10566	040756	001402			BEQ	22\$;NO - SKIP
10567	040760	052710	000020		BIS	#PGERR,(R0)	;SET PGE FLAG
10568							
10569	040764	032713	004000	22\$:	BIT	#NEM,(R3)	;TEST NEM SET
10570	040770	001402			BEQ	23\$;NO - SKIP
10571	040772	052710	000040		BIS	#NEMERR,(R0)	;SET NEM FLAG
10572							
10573	040776	032713	020000	23\$:	BIT	#UPE,(R3)	;TEST UPE SET
10574	041002	001402			BEQ	24\$;NO - SKIP
10575	041004	052710	000100		BIS	#UPERR,(R0)	;SET UPE FLAG
10576							
10577	041010	032715	000001	24\$:	BIT	#ILF,(R5)	;TEST ILF SET
10578	041014	001402			BEQ	25\$;NO - SKIP
10579	041016	052710	000200		BIS	#ILFERR,(R0)	;SET ILF FLAG.

10580
 10581 : THE FOLLOWING CODE IS EXECUTED ONLY IF ERRORS WERE EXPECTED.
 10582 : THE FLAG IN \$TMP13 INDICATES IF
 10583 : AN ERROR WAS EXPECTED AND THE CONTENTS OF TMP10,
 10584 : TEMP11, & TEMP12 SPECIFY WHICH ERRORS. THESE ARE COMPARED AGAINST
 10585 : THE ERRORS FOUND AND STORED IN GRP1ER, GRP2ER, AND GRP3ER.
 10586 : THE CONTENTS OF GRP1,2, & 3 ARE MODIFIED TO INDICATE ERRORS THAT
 10587 : OCCURRED BUT WERE NOT EXPECTED. THE CONTENTS OF \$TMP10,11,
 10588 : & 12 ARE MODIFIED TO INDICATE EXPECTED ERRORS THAT DID NOT
 10589 : OCCUR. BOTH CONDITIONS CAN EXIST AT THE SAME TIME AND MUST
 10590 : BE REPORTED.
 10591 :

10592	041022	005737	001250	25\$:	TST	\$TMP13	;CHECK IF AN ERROR WAS EXPECTED
10593	041026	001423			BEQ	110\$;NO - SKIP
10594	041030	012704	050064		MOV	#GRP1ER,R4	;GET ADDRESS OF ERROR
10595	041034	012705	001242		MOV	#\$TMP10,R5	;GET ADDRESS OF EXPECTED ERRORS
10596							
10597	041040	011500		26\$:	MOV	(R5),R0	;GET EXPECTED ERROR
10598	041042	011401			MOV	(R4),R1	;GET GROUP ERROR FLAGS
10599	041044	020001			CMP	R0,R1	;ARE THEY EQUAL?
10600	041046	001003			BNE	27\$;NO - SKIP
10601	041050	005000			CLR	R0	;CLEAR EXPECTED ED
10602	041052	005001			CLR	R1	;CLEAR OCCURED

10603	041054	000403		BR	28\$	
10604						
10605	041056	010003		27\$:	MOV R0,R3	:STORE EXPECTED ERRORS
10606	041060	040100			BIC R1,R0	:RESET EXPECTED THAT OCCURRED
10607	041062	040301			BIC R3,R1	:RESET OCCURRED THAT EXPECTED
10608	041064	010025		28\$:	MOV R0,(R5)+	:STORE EXPECTED THAT DID NOT OCCUR
10609	041066	010124			MOV R1,(R4)+	:STORE OCCURRED THAT WERE NOT EXPECTED
10610	041070	022705	001250		CMP #STMP13,R5	:ALL GROUPS CHECKED.
10611	041074	001361			BNE 26\$:NO - LOOP
10612						
10613						
10614						
10615						
10616						
10617						
10618						
10619						
10620						
10621						
10622						
10623						
10624						
10625						
10626						
10627						
10628						
10629						
10630						
10631						
10632						
10633						
10634						
10635						
10636						
10637						
10638						
10639	041076	005004		110\$:	CLR R4	:CLEAR COUNTERS
10640	041100	005005			CLR R5	
10641	041102	012700	001224		MOV #STMP1,R0	:LOAD POINTERS FOR TEMPORARY STORAGE OF ADDRESS
10642	041106	012701	001226		MOV #STMP2,R1	:WHERE ASCIZ ADDRESSES GO
10643	041112	012703	050205		MOV #GPSUMF,R3	:POINTERS TO GROUP SUMMARY FLAGS
10644	041116	012710	061052		MOV #DF007A,(R0)	:PRESET FOR GRP3 ERR MESSAGE BUILD
10645	041122	012711	001442		MOV #DH7N,(R1)	
10646	041126	013746	050070		MOV GRP3ER,-(SP)	:GET GROUP 3 ERRORS, PUT ON STACK
10647	041132	004437	036242		JSR R4,BITCNT	:GO COUNT NUMBER AT ERRORS
10648	041136	005716			TST (SP)	:ANY ERRORS?
10649	041140	001403			BEQ 29\$:NO - SKIP
10650	041142	061605			ADD (SP),R5	:ADD IN ERROR TOTAL
10651	041144	052713	000004		BIS #GRP3ST,(R3)	:SET BIT TO INDICATE GROUP 3 ERROR
10652						
10653	041150	005726		29\$:	TST (SP)+	:CLEAR OFF STACK
10654	041152	005737	001250		TST \$TMP13	:ERROR EXPECTED
10655	041156	001412			BEQ 31\$:NO - SKIP
10656	041160	013746	001246		MOV \$TMP12,-(SP)	:PUT GROUP 3 NOT RECEIVED ERRORS ON STACK
10657	041164	004437	036242		JSR R4,BITCNT	:COUNT NUMBER OF ERRORS.
10658	041170	005716			TST (SP)	:WERE THERE ANY

THE FOLLOWING CODE:

- A. DETERMINES WHICH FORMAT IS TO BE USED
- B. LOADS THE ADDRESSES OF THE ASCIZ TEXT INTO THE SELECTED ERROR TABLE ITEM AND FORMAT FIELD
- C. COUNTS THE NUMBER OF ERRORS THAT MUST BE REPORTED
- D. GETS DRIVE STATUS IF GROUP 1 ERROR.

THE DECISION OF WHICH ERROR IS TO BE USED IS BASED ON THE ERROR GROUP (OR GROUPS) THAT HAVE FLAGS SET. IF ANY BIT IS SET IN GROUP 1,2, OR 3, GROUP 1 FORMAT (ERROR 4 OR 5) WILL BE USED; ANY SET IN GROUP 2 OR 3, GROUP 2 (ERROR 6) WILL BE USED; AND A FLAG SET IN GROUP 3 ONLY, GROUP 3 (ERROR 7) IS USED.

THE FORMAT TO BE USED IN THE CONTROLLING FACTOR IN HOW THE ERROR TRAP IS CHANGED IN THE MAIN CALL. IF GROUP 1 FORMAT IS USED THE ERROR TRAP WILL BE CHANGED TO ERROR 4 OR 5 (DEPENDING ON AVAILABILITY OF DRIVE STATUS), GROUP 2 FORMAT WILL BE ERROR 6, AND GROUP 3 WILL BE ERROR 7. ONLY THE LOW ORDER BYTE OF THE ERROR TRAP WILL BE ALTERED. THE SP WILL BE POINTING TO THE LOCATION THAT CONTAINS THE ERROR CALL TRAP.

IF THE STATUS IS READ FROM THE DRIVE WITH NO PROBLEM, ERROR 4 IS USED. IF ANY ERROR IS ENCOUNTERED READING STATUS, ERROR 5 IS USED. ERROR 5 INCLUDES A WARNING MESSAGE.

10659	041172	001403		BEQ	30\$:NO - SKIP
10660	041174	052713	000040	BIS	#GP3NR,(R3)		:SET GROUP 3 NOT RECEIVED ERROR FLAG
10661	041200	061604		ADD	(SP),R4		:ADD COUNT TO TOTAL THESE
10662							
10663	041202	005726		30\$: TST	(SP)+		:CLEAR OFF STACK
10664	041204	013746	050066	31\$: MOV	GRP2ER,-(SP)		:GET GROUP 2 ERRORS FOR COUNTING
10665	041210	004437	036242	JSR	R4,BITCNT		:COUNT BITS
10666	041214	005716		TST	(SP)		:ANY SET?
10667	041216	001407		BEQ	32\$:NO - SKIP
10668	041220	052713	000002	BIS	#GRP2ST,(R3)		:SET FLAG FOR GROUP 2 ERRORS
10669	041224	061605		ADD	(SP),R5		:ADD INTO TOTAL
10670	041226	012710	061026	MOV	#DF006A,(R0)		:STORE ADDRESS FOR BUILDING REPORT
10671	041232	012711	001432	MOV	#DH6N,(R1)		
10672							
10673	041236	005726		32\$: TST	(SP)+		:CLEAR OFF STACK
10674	041240	005737	001250	TST	\$TMP13		:ANY EXPECTED ERRORS
10675	041244	001416		BEQ	34\$:NO - SKIP
10676	041246	013746	001244	MOV	\$TMP11,-(SP)		:GET GROUP 2 NOT RECEIVED ERRORS
10677	041252	004437	036242	JSR	R4,BITCNT		:COUNT NUMBER OF BITS
10678	041256	005716		TST	(SP)		:ANY SET?
10679	041260	001407		BEQ	33\$:NO - SKIP
10680	041262	052713	000020	BIS	#GP2NR,(R3)		:SET FLAG FOR GROUP 2 NOT RECEIVED
10681	041266	061604		ADD	(SP),R4		:ADD INTO TOTAL
10682	041270	012710	061026	MOV	#DF006A,(R0)		:STORE ADDRESS FOR BUILDING REPORT
10683	041274	012711	001432	MOV	#DH6N,(R1)		
10684							
10685	041300	005726		33\$: TST	(SP)+		:CLEAR OFF STACK
10686	041302	013746	050064	34\$: MOV	GRP1ER,-(SP)		:GET GROUP 1 ERROR FLAGS
10687	041306	004437	036242	JSR	R4,BITCNT		:COUNT THE NUMBER OF BITS
10688	041312	005716		TST	(SP)		:ANY SET?
10689	041314	001407		BEQ	35\$:NO - SKIP
10690	041316	052713	000001	BIS	#GRP1ST,(R3)		:SET FLAG FOR GROUP 1 ERRORS SET
10691	041322	061605		ADD	(SP),R5		:ADD INTO TOTAL
10692	041324	012710	060742	MOV	#DF004A,(R0)		:LOAD ADDRESS FOR BUILDING REPORT
10693	041330	012711	001412	MOV	#DH4N,(R1)		
10694	041334	005726		35\$: TST	(SP)+		:CLEAR OFF STACK
10695	041336	005737	001250	TST	\$TMP13		:ANY EXPECTED ERRORS?
10696	041342	001416		BEQ	60\$:NO - SKIP
10697	041344	013746	001242	MOV	\$TMP10,-(SP)		:GET GROUP 1 NO RECEIVED ERROR
10698	041350	004437	036242	JSR	R4,BITCNT		:COUNT # OF BITS
10699	041354	005716		TST	(SP)		:ANY SET?
10700	041356	001407		BEQ	36\$:NO - SKIP
10701	041360	052713	000010	BIS	#GP1NR,(R3)		:SET FLAG FOR GROUP 1 NOT RECEIVED
10702	041364	061604		ADD	(SP),R4		:ADD INTO TOTAL
10703	041366	012710	060742	MOV	#DF004A,(R0)		:LOAD ADDRESS FOR BUILDING REPORT
10704	041372	012711	001412	MOV	#DH4N,(R1)		
10705	041376	005726		36\$: TST	(SP)+		:CLEAR OFF STACK
10706	041400	032713	000011	60\$: BIT	#GRP1ST!GP1NR,(R3)		:ANY GROUP 1 ERROR
10707	041404	001414		BEQ	52\$:NO - SKIP
10708	041406	042713	040000	BIC	#DRSTER,(R3)		
10709	041412	004437	036762	JSR	R4,GETDRS		
10710	041416	000401		BR	51\$:ERROR RETURN
10711	041420	000406		BR	52\$:NO ERROR RETURN
10712	041422	012710	060772	51\$: MOV	#DF005A,(R0)		:CHANGE TO FORMAT 5 - STORE ADDRESS
10713	041426	012711	001422	MOV	#DH5N,(R1)		:FOR BUILDING REPORT
10714	041432	052713	040000	BIS	#DRSTER,(R3)		:SET DRIVE STATUS ERROR

10715 041436 52\$:
10716
10717
10718
10719
10720
10721
10722
10723
10724
10725
10726
10727
10728
10729
10730
10731
10732
10733
10734
10735
10736
10737
10738
10739
10740
10741
10742
10743 041436 032777 020000 137474
10744 041444 001402
10745 041446 000137 042034
10746 041452 005737 001250 37\$:
10747 041456 001004
10748
10749
10750
10751 041460 012771 057745 000000
10752 041466 000411
10753 041470 012771 057566 000000 38\$:
10754 041476 032713 000070
10755 041502 001003
10756 041504 012771 057663 000000
10757 041512 013701 001540 39\$:
10758 041516 042701 177741
10759
10760 041522 016170 050210 000000
10761
10762 041530 032713 000007
10763 041534 001462
10764
10765
10766
10767 041536 013701 050070
10768 041542 012700 050110
10769 041546 005037 001252 140\$:
10770 041552 012737 000021 001254 40\$:

THE ERRORS ARE COUNTED, FLAGS SET TO INDICATE WHICH ERRORS ARE TO BE REPORTED, AND THE ERROR FORMAT HAS BEEN SELECTED. THE FOLLOWING CODE WILL TYPE ALL THE ERRORS, LOAD THE PROPER HEADER MESSAGE ADDRESS IN THE ERROR ITEM TABLE AND LOAD THE PROPER HEADER MESSAGE ADDRESS IN THE PROPER DF TABLE.

AT THIS TIME

R5 CONTAINS EITHER THE NUMBER OF ERRORS THAT OCCURRED BUT WERE NOT EXPECTED OR THE NUMBER OF ERRORS THE OCCURRED IF NONE WERE EXPECTED
R4 CONTAINS THE NUMBER OF ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR.
\$TMP10 CONTAINS GROUP 1 ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR
\$TMP11 CONTAINS GROUP 2 ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR
\$TMP12 CONTAINS GROUP 3 ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR
GRP1ER CONTAINS GROUP 1 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECTED
GRP2ER CONTAINS GROUP 2 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECTED
GRP3ER CONTAINS GROUP 3 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECTED

(R1)=\$TMP2 CONTAINS THE ADDRESS OF THE HEADER MESSAGE ADDRESS IN DF THAT MUST BE ALTERED TO IDENTIFY THE OPERATION

(R0)=\$TMP1 CONTAINS THE ADDRESS OF THE HEADER MESSAGE ADDRESS IN THE ERROR ITEM TABLE THAT MUST BE ALTERED TO PROVIDE A PROPER MESSAGE TO REPORT.

(R3)=\$GRSUMF CONTAIN FLAGS TO INDICATE WHICH OF THE GROUP ERROR FLAG FIELDS HAVE ERROR BITS STORED.

BIT #SW13,@SWR ;IS REPORT INHIBITED?
BEQ 37\$;NO - SKIP
JMP 49\$;ELSE EXIT
TST \$TMP13 ;WERE ERRORS EXPECTED?
BNE 38\$;YES - SKIP

IF NO ERRORS WERE EXPECTED, \$TMP10,11, &12 ARE NOT MEANINGFUL

MOV #DH007,@(R1) ;HEADER = ERROR IN OPERATION
BR 39\$
MOV #DH005,@(R1) ;PRESET HDRMSG = EXPECTED NOT SET
BIT #GP1NR!GP2NR!GP3NR,(R3) ;ANY NOT RECEIVED ERRORS?
BNE 39\$;YES - SKIP
MOV #DH006,@(R1) ;SET MESSAGE TO UNEXPECTED ERROR SET
MOV T,CS1,R1 ;GET CS1
BIC #177741,R1 ;CLEAR ALL BUT COMMAND

MOV CMNDLB(R1),@(R0) ;MOVE ADDRESS OF COMMAND MESSAGE
;INTO REPORT
BIT #GRP1ST!GRP2ST!GRP3ST,(R3) ;ANY GPR ERRORS?
BEQ 46\$;NO - SKIP GPR REPORT

PRINT ALL THE ERRORS CONTAINED IN THE GPR1,2,3ER(UNEXPECTED ERRORS)

MOV GRP3ER,R1 ;GET GROUP 3 ERROR FLAGS
MOV #GRP3MS,R0 ;SET POINTER TO GRP3 ERROR MESSAGES
CLR \$TMP14 ;CLEAR GROUP PRINTING INDICATOR
MOV #17,\$TMP15 ;PRESET SHIFT COUNT

10771	041560	000241				CLC			:CLEAR CARRY
10772	041562	006001			41\$:	ROR	R1		:ROTATE ERROR FLAGS
10773	041564	103406				BCS	42\$:WAS BIT SHIFTED OUT SET?
10774	041566	062700	000002		141\$:	ADD	#2,R0		:NO - BUMP POINTER
10775	041572	005337	001254			DEC	\$TMP15		:DEC SHIFT COUNT
10776	041576	001371				BNE	41\$:LOOP IF SHIFT NOT ZERO
10777	041600	000411				BR	44\$		
10778	041602	011037	041614		42\$:	MOV	(R0),43\$:GET ERROR MESSAGE ADDRESS FROM TABLE
10779	041606	104401	001273			TYPE	,\$CRLF		:TYPE CRLF
10780	041612	104401				TYPE			:TYPE ERROR MESSAGE
10781	041614	000000			43\$:	.WORD			:MESSAGE ADDRESS GOES HERE
10782	041616	005305				DEC	R5		:DECEREMENT TOTAL ERROR COUNT.
10783	041620	001362				BNE	141\$:LOOP IF ZERO
10784	041622	000427				BR	46\$:ELSE EXIT GPR ERROR PRINT LOOP
10785									
10786	041624	005713			44\$:	TST	(R3)		:TEST GPSUMF FLAG FOR PRINTING ERROR NOT RECEIVED
10787	041626	100455				BMI	47\$:YES - SKIP
10788	041630	005737	001252			TST	\$TMP14		:PRINTING GROUP 3?
10789	041634	001007				BNE	45\$:NO -SKIP
10790	041636	013701	050066			MOV	GRP2ER,R1		:ELSE SET TO GROUP 2, GET GRP2ER
10791	041642	012700	050130			MOV	#GRP2MS,R0		:& SET POINTER TO GROUP 2 ERROR MESSAGE TABLE
10792	041646	005237	001252			INC	\$TMP14		:BUMP TO INDICATE PRINTING GROUP 2
10793	041652	000737				BR	40\$:GO RESTART PRINT LOOP
10794	041654	022737	000002	001252	45\$:	CMP	#2,\$TMP14		:PRINTING GROUP 1?
10795	041662	001407				BEQ	46\$:YES - EXIT GPR ERROR PRINT LOOP.
10796	041664	013701	050064			MOV	GRP1ER,R1		:ELSE SET TO GROUP 1, GET GROUP 1 ERROR
10797	041670	012700	050146			MOV	#GRP1MS,R0		:SET POINTER TO GROUP 1 ERROR MESSAGE TABLE
10798	041674	005237	001252			INC	\$TMP14		:BUMP TO INDICATE PRINTING GROUP 1
10799	041700	000724				BR	40\$:RESTART PRINT LOOP.
10800									
10801	041702	005737	001250		46\$:	TST	\$TMP13		:EXPECTING ERRORS?
10802	041706	001452				BEQ	49\$:NO - SKIP
10803									
10804					:				PRINT ALL ERRORS CONTAINED IN \$TMP10, 11, 12(NOT RECEIVED ERRORS)
10805									
10806	041710	005713				TST	(R3)		:TEST IF PRINTING NOT RECEIVED ERRORS
10807	041712	100423				BMI	47\$:YES - SKIP
10808	041714	032713	000070			BIT	#GP1NR!GP2NR!GP3NR,(R3)		:ANY NOT RECEIVED ERRORS
10809	041720	001445				BEQ	49\$:NO - SKIP
10810	041722	032713	000007			BIT	#GRP1ST!GRP2ST!GRP3ST,(R3)		:ANY NOT RECEIVED ERRORS?
10811	041726	001404				BEQ	146\$:NO - SKIP LABEL FOR UNEXPECTED ERRORS
10812	041730	104401	001273			TYPE	,\$CRLF		:TYPE CRLF
10813	041734	104401	057663			TYPE	,\$DH006		:TYPE HEADER FOR PREVIOUS ERRORS
10814	041740	052737	100000	050206	146\$:	BIS	#REPNR,GPSUMF		:SET PRINTING NOT RECEIVED ERRORS SWITCH
10815	041746	010405				MOV	R4,R5		:MOVE TOTAL ERRORS TO R5
10816	041750	013701	001246			MOV	\$TMP12,R1		:GET GRP3 NOT RECEIVED ERRORS
10817	041754	012700	050110			MOV	#GRP3MS,R0		:SET POINTER TO GROUP 3 MESSAGES
10818	041760	000672				BR	140\$:GO START PRINT LOOP
10819	041762	005737	001252		47\$:	TST	\$TMP14		:PRINTING GROUP 3?
10820	041766	001007				BNE	48\$:NO - SKIP
10821	041770	013701	001244			MOV	\$TMP11,R1		:ELSE SETUP TO PRINT GROUP 2 - GET ERRORS
10822	041774	012700	050130			MOV	#GRP2MS,R0		:& SET POINTER TO GROUP 2 MESSAGE TABLE
10823	042000	005237	001252			INC	\$TMP14		:BUMP TO INDICATE GROUP 2 PRINTING
10824	042004	000662				BR	40\$:GO START PRINT LOOP
10825	042006	022737	000002	001252	48\$:	CMP	#2,\$TMP14		:PRINTING GROUP 1?
10826	042014	001407				BEQ	49\$:YES - EXIT LOOP

```

10827 042016 013701 001242      MOV      $TMP10,R1      ;SET POINTER TO GROUP 1 MESSAGE
10828 042022 012700 050146      MOV      #GRP1MS,R0    ;TABLE AND GET GROUP 1 ERRORS.
10829 042026 005237 001252      INC      $TMP14        ;BUMP TO INDICATE GROUP 1 PRINTING
10830 042032 000647              BR       40$           ;START LOOP AGAIN.
10831
10832 042034 032713 000077      49$:    BIT      #77,(R3) ;TEST IF ANY ERRORS TO BE REPORTED
10833                                ; GRP1ST!GRP2ST!GRP3ST
10834                                ; GP1NR!GP2NR!GP3NR
10835 042040 001004      BNE      61$           ;YES - SKIP
10836 042042 104414      RESREG              ;ELSE EXIT
10837 042044 062716 000002      ADD      #2,(SP)      ;BUMP FOR GOOD RETURN
10838 042050 000002      RTI
10839
10840 042052 112776 000007 000000 61$:    MOVB     #7,@(SP)      ;PRESET FOR GROUP 3 ERROR RETURN.
10841 042060 032713 000022      BIT      #GRP2ST!GP2NR,(R3) ;ANY GROUP 2 ERRORS?
10842 042064 001403      BEQ      50$           ;NO - SKIP
10843 042066 112776 000006 000000      MOVB     #6,@(SP)      ;ELSE SET FOR GROUP 2 ERROR RETURN
10844
10845 042074 032713 000011      50$:    BIT      #GRP1ST!GP1NR,(R3) ;ANY GROUP 1 ERRORS?
10846 042100 001411      BEQ      53$           ;NO - SKIP
10847 042102 112776 000004 000000      MOVB     #4,@(SP)      ;ELSE SET FOR GROUP 1 ERROR RETURN.
10848 042110 032713 040000      BIT      #DRSTER,(R3) ;CHECK IF ERROR GETTING DRIVE STATUS
10849 042114 001403      BEQ      53$           ;NO - SKIP
10850 042116 112776 000005 000000      MOVB     #5,@(SP)      ;ELSE CHANGE RETURN FORM GROUP 1
10851
10852 042124 005737 001264      53$:    TST      $ESCAPE      ;CHECK IF ESCAPE ALREADY SET
10853 042130 001011      BNE      54$           ;YES - SKIP
10854 042132 013700 001302      MOV      $TESTN,R0     ;SET UP $ESCAPE TO FORCE
10855 042136 006300      ASL      R0            ;ABORT TO PRESENT TEST AFTER
10856 042140 016037 033122 001264      MOV      $$W08TB(R0), $ESCAPE ;ERROR IS REPORTED
10857 042146 162737 000002 001264      SUB      #2,$ESCAPE    ;BUT GO TO NEXT SCOPE STATEMENT
10858 042154 104414      54$:    RESREG
10859 042156 000002      RTI                    ;RETURN
10860
10861
10862
10863
10864
10865
10866
10867
10868
10869
10870
10871
10872
10873
10874
10875
10876 042160 012637 001236      BDSRCK: MOV      (SP)+,$TMP6 ;STORE OLD R4 CONTENTS
10877 042164 010437 001240      MOV      R4,$TMP7     ;GET RETURN ADDRESS
10878 042170 011404      MOV      (R4),R4      ;GET POINTER TO FIELD TO BE CHECKED
10879 042172 005037 001234      CLR      $TMP5        ;CLEAR A COUNTER
10880 042176 005714      1$:    TST      (R4)        ;TEST IF FIELD HAS NO (OR NO MORE) ENTRIES
10881 042200 100417      BMI      4$           ;YES - EXIT
10882 042202 023724 001614      CMP      L.DCYL,(R4)+ ;IS THIS ENTRY FOR THIS CYLINDER?

```

```

*****
:SBTTL BAD SECTOR CHECK
:* THE FIELD WHOSE ADDRESS IS IN THE LOCATION AFTER THE
:* CALL IS CHECKED TO SEE IF ANY SECTORS ARE LISTED THEREIN
:* THAT HAVE THE CYLINDER AND TRACK ADDRESS SPECIFIED IN
:* L.DCYL AND L.DT. IF A SECTOR IS FOUND IN THIS FIELD
:* THAT IS BAD FOR THAT CYLINDER AND TRACK, THE SECTOR NUMBER
:* IS PLACED ON THE STACK. THE TOTAL NUMBER OF BAD SECTORS
:* IS PLACED ON THE STACK AFTER THE ENTIRE
:* FIELD IS SEARCHED.
:*
:* CALL: JSR R4,BDSRCK
:* <ADDRESS OF FIELD TO BE SEARCHED>
*****

```

10883	042206	001012		BNE	3\$:NO - SKIP
10884	042210	005204		INC	R4		:BUMP TO TRACK
10885	042212	123714	001607	CMPB	L.DT,(R4)		:IS ENTRY FOR THIS TRACK?
10886	042216	001005		BNE	2\$:NO - SKIP
10887	042220	005046		CLR	-(SP)		:CLEAR STACK LOCATION
10888	042222	114416		MOVB	-(R4),(SP)		:PUT SECTOR NUMBER ON STACK
10889	042224	005237	001234	INC	\$TMP5		:BUMP COUNTER
10890	042230	000401		BR	3\$:BRANCH
10891							
10892	042232	005304		2\$: DEC	R4		:DECREMENT POINTER TO WORD ALIGN
10893	042234	005724		3\$: TST	(R4)+		:BUMP TO NEXT ENTRY
10894	042236	000757		BR	1\$:TEST NEXT ENTRY
10895							
10896	042240	013746	001234	4\$: MOV	\$TMP5,-(SP)		:PUT COUNT ON STACK
10897	042244	013746	001236	MOV	\$TMP6,-(SP)		:PUT OLD R4 CONTENTS BACK ON STACK
10898	042250	013704	001240	MOV	\$TMP7,R4		:SET UP RETURN
10899	042254	005724		TST	(R4)+		:BUMP PAST PARAMETER
10900	042256	000204		RTS	R4		:RETURN

10901
10902
10903
10904
10905
10906
10907
10908
10909
10910
10911
10912
10913
10914
10915
10916
10917
10918
10919
10920
10921
10922
10923
10924
10925
10926
10927
10928
10929
10930
10931
10932
10933
10934
10935
10936
10937
10938

SBTTL DATA GENERATION AND COMPARE ROUTINE

* CALLS: JSR R4,GENCOM
* CONTROL WORD
* JSR R4,GENCOM
* CONTROL WORD
* LENGTH
* JSR R4,GENCOM
* CONTROL WORD
* RELOCATION CONSTANT
* LENGTH

RETURN: RTS R4

R4 IS ADJUSTED IN THE CODE FOR THE FOLLOWING RETURNS:
THE FIRST CALL RETURNS TO THE LOCATION FOLLOWING THE
CONTROL WORD. THIS IS UNCONDITIONAL.

THE SECOND CALL RETURNS TO THE LOCATION FOLLOWING THE LENGTH IF
THE OPERATION REQUIRES DATA COMPARE AND DATA MISCOMPARED.
IF DATA IS TO BE GENERATED ONLY OR NO DATA COMPARE
ERRORS OCCURRED, THE RETURN IS TO LENGTH +4.

THE THIRD CALL IS IDENTICAL TO THE SECOND.

DEFINITION OF CONTROL WORD:

BIT 15 - DO COMPARE OPERATION OF Ibuff (SOURCE) TO Obuff
(DESTINATION). EXPECTED VALUES ARE IN Obuff (DESTINATION).
BIT 14 - RESUME COMPARE OPERATION FROM POINT LEFT BY LAST COMPARE.
BIT 13 - INVOKE MEMORY MANAGEMENT FOR SOURCE (IBUFF).
BIT 12 - INVOKE MEMORY MANAGEMENT FOR DESTINATION (OBUFF).
BIT 11 - REPEAT FIRST WORD OF SELECTED PATTERN THROUGHOUT OBUFF.
BIT 10 - CLEAR Ibuff TO PATTERN SELECTED.
BIT 9 - BUILD HEADERS, CONSIDERING BS FILES
BIT 8 - BUILD HEADERS, ALL SECTORS INDICATE GOOD SECTORS.
BIT 7 - HEADER OPERATION SPECIFIED (EITHER COMPARE OR BUILD).

```

10939
10940
10941
10942
10943
10944
10945
10946
10947
10948
10949
10950
10951
10952
10953
10954
10955
10956
10957
10958
10959
10960
10961
10962
10963
10964
10965
10966
10967
10968
10969
10970
10971
10972
10973
10974
10975
10976
10977
10978
10979
10980
10981
10982
10983
10984
10985
10986
10987
10988 042260
10989 042260 010046
10990 042262 010146
10991 042264 010346
10992 042266 010546
10993 042270 012400
10994 042272 012737 056312 001520

```

```

:* BIT 6 TO 0 - PATTERN SELECT FIELD, OCTAL ENCODED. 0 INDICATES
:* NO DATA GENERATION, 1 IS ALL ZEROS, AND 7 IS ALL ONES.
:* OTHER PATTERNS PROVIDED ARE PATTERNS 2-6, 8-16.
:*
:* EXPLANATION OF CALLS:
:* THE CALL WITH CONTROL WORD THE ONLY PARAMETER IS USED FOR
:* BUILDING OR COMPARING HEADERS OR RESUMING A COMPARE OPERATION.
:*
:* THE CALL WITH CONTROL WORD AND LENGTH AS PARAMETERS IS USED
:* FOR DATA GENERATION OR COMPARE AND FOR Ibuff INITIALIZATION.
:*
:* THE CALL WITH CONTROL WORD, RELOCATION CONSTANT, AND LENGTH IS
:* USED FOR DATA GENERATION OR COMPARE WITH MEMORY MANAGEMENT.
:*
:* DESCRIPTION:
:*
:* THIS ROUTINE IS MULTI-PURPOSE AND WILL PERFORM THE FOLLOWING:
:* A. BUILD HEADERS, EITHER 20 OR 22 SECTORS/TRACK MODE. THE
:* ROUTINE WILL BUILD THE HEADERS AS ALL GOOD SECTORS (BIT 8)
:* OR TAKE THE BAD SECTOR FILES (HARDWARE OR SOFTWARE) FOR
:* EITHER FORMAT) INTO ACCOUNT AND BUILD THE HEADERS WITH THE
:* SECTORS MARKED BAD IF ANY SECTORS FOR THE CYLINDER - TRACK
:* ARE LISTED THEREIN (BIT 9).
:*
:* B. COMPARE THE CONTENTS OF Ibuff AND Obuff (BIT 15). THE
:* CONTENTS OF THE BUFFER MAY BE HEADERS OR DATA. A
:* HEADER COMPARE OPERATION MAY BE SPECIFIED (BIT 7) WHICH
:* WILL CAUSE THE COMPARE TO BE LIMITED TO 74(8) OR 102(8)
:* WORDS OF HEADERS. THE LENGTH DEPENDS ON THE FORMAT
:* BIT THAT WAS LAST SPECIFIED IN L.CS1. THE HEADERS
:* MAY BE BUILT BEFORE THE COMPARE AS PART OF THE
:* OPERATION (BIT 15 AND BIT 8 OR 9). DATA CAN ALSO BE
:* GENERATED BEFORE THE COMPARE (NON-ZERO BITS 6-0).
:*
:* C. RESUME COMPARE OPERATION. IF A COMPARE OPERATION
:* DETECTS A MISCOMPARE, THE ROUTINE RETURNS TO CALLER
:* BUT STORES PARAMETERS SUCH THAT THE COMPARE CAN BE
:* RESUMED. THIS IS DONE BY CALLING GENCOM WITH BIT 14
:* SET IN THE CONTROL WORD.
:*
:* D. DATA GENERATION OR COMPARE USING MEMORY MANAGEMENT.
:* MEMORY MANAGEMENT CAN BE INVOKED FOR EITHER
:* SOURCE OR DESTINATION BUT NOT FOR BOTH. IN THIS
:* MANNER, DATA GENERATION CAN BE MADE TO
:* PLACE DATA ANYWHERE IN AVAILABLE MEMORY. LIKEWISE
:* DATA COMPARE WILL COMPARE THE CONTENTS OF Ibuff TO
:* ANY AREA OF AVAILABLE MEMORY.

```

```

GENCOM:
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
MOV (R4)+,R0 ;;GET PARAMETER WORD
MOV #EM54,EM15N ;;PRESET FOR HEADER COMPARE ERROR

```

10995	042300	032700	000200		BIT	#BIT7,R0	:HEADER OPERATION SPECIFIED?	
10996	042304	001005			BNE	18\$:YES - SKIP	
10997	042306	012737	056341	001520	MOV	#EM55,EM15N	:CHANGE FOR DATA COMPARE ERROR	
10998	042314	000137	043046		JMP	17\$:ELSE JUMP TO DATA ROUTINE	
10999	042320			18\$:				
11000	042320	010446			MOV	R4,-(SP)	::PUSH R4 ON STACK	
11001	042322	032700	001400		BIT	#BIT8:BIT9,R0	:MUST HEADERS BE BUILT?	
11002	042326	001002			BNE	19\$:YES - SKIP	
11003	042330	000137	042630		JMP	11\$:ELSE JUMP TO HEADER COMPARE	
11004	042334	113701	001607	19\$:	MOVB	L.DT,R1	:START HEADER BUILD ROUTINE	
11005	042340	013703	001614		MOV	L.DCYL,R3	:GET TRACK AND CYL	
11006	042344	012705	000005		MOV	#5,R5	:SET COUNT TO SHIFT TRACK FOR HDR WORD	
11007								
11008	042350	006301		1\$:	ASL	R1	:SHIFT TRACK	
11009	042352	005305			DEC	R5	:DECREMENT TRACK	
11010	042354	001375			BNE	1\$:LOOP UNTIL COUNT 0	
11011								
11012	042356	012704	000026		MOV	#26,R4	:PRESET FOR 26 SECTOR MODE	
11013	042362	032737	010000	001600	BIT	#CFMT,L.CS1	:IS IT 24 SECTOR MODE?	
11014	042370	001404			BEQ	2\$:NO - SKIP	
11015	042372	012704	000024		MOV	#24,R4	:CHANGE COUNT FOR 24 SECTOR MODE	
11016	042376	052701	001000		BIS	#BIT9,R1	:SET 24 SECTOR MODE BIT IN WRD 2 OF HDR	
11017								
11018	042402	052701	140000	2\$:	BIS	#BIT15:BIT14,R1	:SET BS BITS TO INDICATE GOOD SECTOR	
11019	042406	012705	072414		MOV	#OBUF,R5	:SET POINTER TO ADDRESS WHERE HEADERS GO	
11020	042412	010325		3\$:	MOV	R3,(R5)+	:INSERT CYLINDER	
11021	042414	010125			MOV	R1,(R5)+	:INSERT TRACK AND SECTOR	
11022	042416	010337	001224		MOV	R3,\$TMP1	:CALCULATE HVRC WORD	
11023	042422	010115			MOV	R1,(R5)		
11024	042424	040137	001224		BIC	R1,\$TMP1		
11025	042430	040315			BIC	R3,(R5)		
11026	042432	053725	001224		BIS	\$TMP1,(R5)+	:COMPLETE HVRC WORD INSERTION	
11027								
11028	042436	005304			DEC	R4	:DECREMENT HEADER COUNT	
11029	042440	001402			BEQ	4\$:DONE? - YES, SKIP	
11030	042442	005201			INC	R1	:BUMP SECTOR	
11031	042444	000762			BR	3\$:LOOP	
11032								
11033	042446	032700	001000	4\$:	BIT	#BIT9,R0	:MUST HEADERS BE CORRECTED FOR TABLE ENTRIES?	
11034	042452	001003			BNE	5\$:YES - SKIP	
11035	042454	005700		10\$:	TST	R0	:IS THIS A COMPARE OPERATION?	
11036	042456	100464			BMI	11\$:YES-GO DO HDR COMPARE	
11037	042460	000534			BR	50\$:ELSE GET OUT	
11038								
11039	042462	013737	001640	042520	5\$:	MOV	BSF26P,6\$:PRESET FOR BS FACTORY LIST
11040	042470	012737	100000	001224	MOV	#BIT15,\$TMP1	:SET BIT TO BE RESET IN BAD HEADER	
11041	042476	032737	010000	001600	BIT	#CFMT,L.CS1	:IS THIS 26 SECTOR MODE?	
11042	042504	001403			BEQ	8\$:YES - SKIP	
11043	042506	013737	001636	042520	MOV	BSF24P,6\$:ELSE CHANGE FOR 24 SECTOR MODE	
11044								
11045	042514	004437	042160	8\$:	JSR	R4,BDSRCK	:GO CHECK FOR BAD SECTOR THIS ADDRESS	
11046	042520	000000		6\$:	.WORD	0	:POINTER TO FILE TO BE CHECKED GOES HERE	
11047	042522	012605			MOV	(SP)+,R5	:GET # OF BAD SECTORS THIS PACK ADDRESS	
11048	042524	001417			BEQ	9\$:SKIP IF ZERO	
11049								
11050	042526	011601		7\$:	MOV	(SP),R1	:GET 1ST BAD SECTOR NUMBER	

11051	042530	006301			ASL	R1	:MULTIPLY SECTOR NUMBER BY 6 TO	
11052	042532	006301			ASL	R1	:LOACATE SECTOR TO BE MARKED BAD	
11053	042534	061601			ADD	(SP),R1		
11054	042536	062601			ADD	(SP)+,R1		
11055	042540	062701	000002		ADD	#2,R1	:ADD 2 FOR 2ND WORD THAT SECTOR	
11056	042544	043761	001224	072414	BIC	\$TMP1,OBUFF(R1)	: CLEAR BIT FOR BAD SECTOR IN HDR	
11057	042552	043761	001224	072416	BIC	\$TMP1,OBUFF+2(R1)	: CORRECT THE HVRC BIT	
11058	042560	005305			DEC	R5	:DECREMENT BAD SECTOR COUNT	
11059	042562	001361			BNE	7\$:LOOP IF NOT ZERO	
11060								
11061	042564	032737	100000	001224	9\$:	BIT	#BIT15,\$TMP1	:WERE WE DOING BS FACTORY LIST?
11062	042572	001730			BEQ	10\$:NO - GO CHECK IF COMPARE MUST BE DONE	
11063	042574	012737	040000	001224	MOV	#BIT14,\$TMP1	:ELSE SET BIT TO BE RESET IN BAD HDR	
11064	042602	013737	001644	042520	MOV	BSS26P,6\$:PRESET POINTER FOR 26 SECTOR MODE	
11065	042610	032737	010000	001600	BIT	#CFMT,L.CS1	:TEST IF WE ARE DOING 26 SECTOR MODE	
11066	042616	001736			BEQ	8\$:YES - SKIP TO START CHECK	
11067	042620	013737	001642	042520	MOV	BSS24P,6\$:CHANGE POINTER TOR 24 SECTOR MODE	
11068	042626	000732			BR	8\$:SKIP TO START CHECK	
11069								
11070								
11071	042630	012701	000102		11\$:	START OF COMPARE MOV	#102,R1	:PRESET FOR 102 WORDS OF HEADER
11072	042634	032737	010000	001600	BIT	#CFMT,L.CS1	:CHECK IF 26 SECTOR MODE	
11073	042642	001402			BEQ	12\$:YES - SKIP	
11074	042644	012701	000074		MOV	#74,R1	:CHANGE TO 74 WORDS OF HEADER	
11075								
11076	042650	012704	070414		12\$:	MOV	#IBUFF,R4	:SET START OF HEADERS TO BE COMPARED
11077	042654	012705	072414		MOV	#OBUFF,R5	:SET START OF GOOD HEADERS	
11078	042660	005003			CLR	R3	:CLEAR COUNTER	
11079	042662	032700	040000		BIT	#BIT14,R0	:IS THIS A CONTINUATION OF EARLIER COMPARE	
11080	042666	001412			BEQ	13\$:NO - SKIP	
11081	042670	013705	001700		28\$:	MOV	DESHLD,R5	:GET VALUES WHERE PREVIOUS CHECK STOPPED
11082	042674	013704	001702		MOV	SRCHLD,R4	: DESTINATION AND SOURCE	
11083	042700	013703	001704		MOV	WRDNUM,R3	: WORD NUMBER IN ERROR	
11084	042704	013701	001706		MOV	WRDCNT,R1	: WORD COUNT LEFT IN COMPARE	
11085	042710	005701			TST	R1	:TEST IF WORD COUNT LEFT = 0	
11086	042712	001417			BEQ	50\$:YES - EXIT	
11087								
11088	042714	032700	030000		13\$:	BIT	#BIT12!BIT13,R0	:MEM MANAGE REQUIRED?
11089	042720	001402			BEQ	25\$:NO - SKIP	
11090	042722	005237	177572		INC	@#SRO	:TURN IT ON	
11091	042726	022425			25\$:	CMP	(R4)+,(R5)+	:COMPARE THE WORDS
11092	042730	001012			BNE	14\$:SKIP IF NOT EQUAL	
11093	042732	005203			INC	R3	:BUMP WORD NUMBER IN ERROR	
11094	042734	005301			DEC	R1	:DEC WORD COUNT LEFT IN COMPARE	
11095	042736	001373			BNE	25\$:LOOP IF NOT ZERO	
11096	042740	032700	030000		BIT	#BIT12!BIT13,R0	:MEM MANAGE IN USE?	
11097	042744	001402			BEQ	50\$:NO - SKIP	
11098	042746	005337	177572		DEC	@#SRO	:TURN IT OFF	
11099	042752				50\$:			
11100	042752	012604			MOV	(SP)+,R4	:POP STACK INTO R4	
11101	042754	000427			BR	16\$		
11102								
11103					:		ERROR REPORT PREP AND PARAMETER STORAGE FOR CONTINUATION	
11104								
11105	042756	010537	001700		14\$:	MOV	R5,DESHLD	:STORE DESTINATION
11106	042762	010437	001702		MOV	R4,SRCHLD	: SOURCE	

```
11107 042766 014537 001202      MOV      -(R5), $REG10      ;LOAD GOOD WORD FOR REPORT
11108 042772 014437 001204      MOV      -(R4), $REG11      ;      BAD WORD
11109 042776 010337 001206      MOV      R3, $REG12        ;      WORD NUMBER
11110 043002 005301                DEC      R1                  ;DEC COUNT LEFTFOR CONTINUATION
11111 043004 005203                INC      R3                  ;BUMP BAD WORD NUMBER
11112 043006 010137 001706      MOV      R1, WRDCNT         ;STORE COUNT LEFT
11113 043012 010337 001704      MOV      R3, WRDNUM         ;      WORD NUM IN ERROR
11114 043016 032700 030000      BIT      #BIT12!BIT13, R0   ;MEM MANAGE IS USE?
11115 043022 001402                BEQ      15$                ;NO - SKIP
11116 043024 005337 177572      DEC      @#SRC              ;TURN IT OFF
11117
11118 043030                15$:
11119 043030 012604                MOV      (SP)+, R4          ;:POP STACK INTO R4
11120 043032 005724                TST      (R4)+              ;ERROR RETURN
11121
11122 043034                16$:
11123 043034 012605                MOV      (SP)+, R5          ;:POP STACK INTO R5
11124 043036 012603                MOV      (SP)+, R3          ;:POP STACK INTO R3
11125 043040 012601                MOV      (SP)+, R1          ;:POP STACK INTO R1
11126 043042 012600                MOV      (SP)+, R0          ;:POP STACK INTO R0
11127 043044 000204                RTS      R4
11128
11129 ;      DATA PATTERN PROCESSING ROUTINE
11130
11131 043046 032700 040000      17$:  BIT      #BIT14, R0      ;CONTINUE WITH COMPARE?
11132 043052 001402                BEQ      29$                ;NO - SKIP
11133 043054 010446                MOV      R4, -(SP)          ;STORE RETURN
11134 043056 000704                BR       28$                ;GO CONTINUE COMPARE
11135
11136 043060 012705 072414      29$:  MOV      #OBUFF, R5          ;GET DESTINATION
11137 043064 012703 070414      MOV      #IBUFF, R3         ;GET SOURCE
11138 043070 032700 030000      BIT      #BIT12!BIT13, R0   ;USE MEM MANAGE?
11139 043074 001412                BEQ      21$                ;NO - SKIP
11140
11141 043076 012437 172354      MOV      (R4)+, @#KIPAR6    ;LOAD PAR FOR RELOCATION
11142 043102 032700 010000      BIT      #BIT12, R0         ;RELOCATE SOURCE?
11143 043106 001403                BEQ      20$                ;NO - SKIP
11144 043110 012705 140070      MOV      #140070, R5        ;SET DESTINATION TO USE PAR6 + OFFSET
11145 043114 000402                BR       21$                ;SKIP
11146 043116 012703 140070      20$:  MOV      #140070, R3        ;SET SOURCE TO USE PAR6 + OFFSET
11147
11148 043122 012401      21$:  MOV      (R4)+, R1          ;STORE COUNT
11149 043124 010446                MOV      R4, -(SP)          ;STORE RETURN
11150 043126 010304                MOV      R3, R4             ;PUT IN Ibuff POINTER
11151 043130 005003                CLR      R3                  ;CLEAR R3 FOR WORD NUMBER COUNTER
11152 043132 032700 000077      BIT      #77, R0            ;ANY DATA PATTERN SPECIFIED?
11153 043136 001666                BEQ      13$                ;NO - GO DO COMPARE
11154
11155 ;      START OF GENERATION ROUTINE
11156
11157 043140 010537 001700      MOV      R5, DESHLD         ;STORE PARAMETERS FOR COMPARE
11158 043144 010437 001702      MOV      R4, SRCHLD
11159 043150 010337 001704      MOV      R3, WRDNUM
11160 043154 010137 001706      MOV      R1, WRDCNT
11161
11162 ;      CODE TO GENERATE DATA PATTERN IN AREA POINTED TO BY R5.
```

CZR6KGO RK6 FCTNL CTRL DIAG
CZR6KG.P11 28-AUG-81 15:16

MACY11 30(1046) 28-AUG-81 15:17 PAGE 207
DATA GENERATION AND COMPARE ROUTINE

L 16

SEQ 0206

11163
11164
11165
11166 043160 032700 030000
11167 043164 001402

: MEMORY MANAGEMENT WILL BE TURNED ON BUT RELOCATION
: WILL NOT OCCUR UNLESS REQUIRED BY SWITCHES
BIT #BIT12:BIT13,R0 ;MEMORY MANAGEMENT REQUIRED?
BEQ 33\$;NO - SKIP

11168	043166	005237	177572		INC	@#SRO	:TURN IT ON
11169	043172	032700	002000	33\$:	BIT	#BIT10,R0	:GENERATE PATTERN IN Ibuff?
11170	043176	001403			BEQ	32\$:NO - SKIP
11171	043200	010446			MOV	R4,-(SP)	:ELSE SWAP R4 AND R5
11172	043202	010504			MOV	R5,R4	
11173	043204	012605			MOV	(SP)+,R5	
11174							
11175	043206	122700	000001	32\$:	CMPB	#1,R0	:PATTERN 1 (ALL ZEROS)?
11176	043212	001004			BNE	55\$:NO - SKIP
11177	043214	005025		30\$:	CLR	(R5)+	:CLEAR WORD IN BUFF
11178	043216	005301			DEC	R1	:DEC WORD COUNT
11179	043220	001375			BNE	30\$:LOOP UNTIL WORD COUNT ZERO
11180	043222	000550			BR	22\$:EXIT BUILD
11181							
11182	043224	122700	000007	55\$:	CMPB	#7,R0	:PATTERN 7 (ALL ONES)?
11183	043230	001005			BNE	56\$:NO - SKIP
11184	043232	012725	177777	31\$:	MOV	#-1,(R5)+	:LOAD WORD IN BUFF
11185	043236	005301			DEC	R1	:DEC WORD COUNT
11186	043240	001374			BNE	31\$:LOOP UNTIL WORD COUNT ZERO
11187	043242	000540			BR	22\$:EXIT BUILD
11188							
11189	043244	122700	000002	56\$:	CMPB	#2,R0	:PATTERN 2 SET UP
11190	043250	001003			BNE	57\$	
11191	043252	012703	047234		MOV	#PAT02,R3	
11192	043256	000504			BR	70\$	
11193							
11194	043260	122700	000003	57\$:	CMPB	#3,R0	:PATTERN 3 SET UP
11195	043264	001003			BNE	58\$	
11196	043266	012703	047274		MOV	#PAT03,R3	
11197	043272	000476			BR	70\$	
11198							
11199	043274	122700	000004	58\$:	CMPB	#4,R0	:PATTERN 4 SET UP
11200	043300	001003			BNE	59\$	
11201	043302	012703	047334		MOV	#PAT04,R3	
11202	043306	000470			BR	70\$	
11203							
11204	043310	122700	000005	59\$:	CMPB	#5,R0	:PATTERN 5 SET UP
11205	043314	001003			BNE	60\$	
11206	043316	012703	047374		MOV	#PAT05,R3	
11207	043322	000462			BR	70\$	
11208							
11209	043324	122700	000006	60\$:	CMPB	#6,R0	:PATTERN 6 SET UP
11210	043330	001003			BNE	61\$	
11211	043332	012703	047434		MOV	#PAT06,R3	
11212	043336	000454			BR	70\$	
11213							
11214	043340	122700	000010	61\$:	CMPB	#10,R0	:PATTERN 10 SET UP
11215	043344	001003			BNE	62\$	
11216	043346	012703	047474		MOV	#PAT10,R3	
11217	043352	000446			BR	70\$	
11218							
11219	043354	122700	000011	62\$:	CMPB	#11,R0	:PATTERN 11 SET UP
11220	043360	001003			BNE	63\$	
11221	043362	012703	047534		MOV	#PAT11,R3	
11222	043366	000440			BR	70\$	
11223							

11224	043370	122700	000012	63\$:	CMPB	#12,R0	;PATTERN 12 SET UP
11225	043374	001003			BNE	64\$	
11226	043376	012703	047574		MOV	#PAT12,R3	
11227	043402	000432			BR	70\$	
11228							
11229	043404	122700	000013	64\$:	CMPB	#13,R0	;PATTERN 13 SET UP
11230	043410	001003			BNE	65\$	
11231	043412	012703	047634		MOV	#PAT13,R3	
11232	043416	000424			BR	70\$	
11233							
11234	043420	122700	000014	65\$:	CMPB	#14,R0	;PATTERN 14 SET UP
11235	043424	001003			BNE	66\$	
11236	043426	012703	047674		MOV	#PAT14,R3	
11237	043432	000416			BR	70\$	
11238							
11239	043434	122700	000015	66\$:	CMPB	#15,R0	;PATTERN 15 SET UP
11240	043440	001003			BNE	67\$	
11241	043442	012703	047734		MOV	#PAT15,R3	
11242	043446	000410			BR	70\$	
11243							
11244	043450	122700	000016	67\$:	CMPB	#16,R0	;PATTERN 16 SET UP
11245	043454	001003			BNE	68\$	
11246	043456	012703	047774		MOV	#PAT16,R3	
11247	043462	000402			BR	70\$	
11248							
11249	043464	012703	047774	68\$:	MOV	#PAT16,R3	;SET UP FOR 16
11250							
11251	043470	032700	004000	70\$:	BIT	#BIT11,R0	;FIRST WORD REPEAT?
11252	043474	001020			BNE	73\$;YES - SKIP
11253	043476	010446			MOV	R4,-(SP)	;STORE R4
11254	043500	010046			MOV	R0,-(SP)	;STORE R0
11255	043502	012700	000020		MOV	#16,R0	;PRESET COUNT FOR PATTERN LENGTH
11256	043506	010504			MOV	R5,R4	;STORE START OF BUFF
11257							
11258	043510	012325		71\$:	MOV	(R3)+,(R5)+	;MOV WORD TO BUFF
11259	043512	005301			DEC	R1	;DEC WORD COUNT
11260	043514	001405			BEQ	74\$;EXIT IF ZERO
11261	043516	005300			DEC	R0	;DEC PAT LENGTH COUNT
11262	043520	001373			BNE	71\$;LOOP IF NOT ZERO
11263							
11264	043522	012425		72\$:	MOV	(R4)+,(R5)+	;REPEAT PATTERN IN BUFFER
11265	043524	005301			DEC	R1	;DEC WORD COUNT
11266	043526	001375			BNE	72\$;LOOP UNTIL WORD COUNT ZERO
11267							
11268	043530	012600		74\$:	MOV	(SP)+,R0	;RESTORE R0
11269	043532	012604			MOV	(SP)+,R4	;RESTORE R4
11270	043534	000403			BR	22\$;EXIT BUILD
11271							
11272	043536	011325		73\$:	MOV	(R3),(R5)+	;MOV THE SAME WORD INTO BUFFER
11273	043540	005301			DEC	R1	;DEC WORD COUNT
11274	043542	001375			BNE	73\$;LOOP UNTIL ZERO
11275							
11276	043544	032700	030000	22\$:	BIT	#BIT12!BIT13,R0	;MEMORY MANAGEMENT REQUIRED?
11277	043550	001402			BEQ	34\$;NO - SKIP
11278	043552	005337	177572		DEC	@#SR0	;TURN OFF MEM MANAGEMENT
11279	043556	005700		34\$:	TST	R0	;IS COMPARE REQUIRED?

```
11280 043560 100012          BPL      23$          ;NO - SKIP
11281 043562 013705 001700   MOV      DESHLD,R5    ;RESTORE COMPARE PARAMETERS
11282 043566 013704 001702   MOV      SRCHLD,R4
11283 043572 013703 001704   MOV      WRDNUM,R3
11284 043576 013701 001706   MOV      WRDCNT,R1
11285 043602 000137 042714   JMP      13$          ;GO START COMPARE
11286 043606          23$:
11287 043606 012604          MOV      (SP)+,R4     ;;POP STACK INTO R4
11288 043610 000137 043034   JMP      16$          ;GO TO EXIT
11289
11290          ;*****
11291          .SBTTL PHASE LOCK LOOP CLOCK ADJUSTMENT ROUTINE
11292          ;* THIS ROUTINE IS ENTERED VIA A START AT LOCATION 220(8). THE
11293          ;* PROGRAM FIRST RUNS TEST 1, 2, AND 3 TO SET UP THE INTERNAL
11294          ;* PROGRAM VARIABLES AND THEN JUMPS TO THE CLOCK ADJUST ROUTINE.
11295          ;* THE ROUTINE SELECTS THE FIRST AVAILABLE DRIVE AND SETS AND
11296          ;* RESETS DIAGNOSTIC MODE BIT IN MR1. INSTRUCTIONS ON WHERE TO
11297          ;* SCOPE AND WHAT TO ADJUST ARE TYPED ON THE CONSOLE.
11298          ;*
11299          ;* THIS ROUTINE WILL LOOP UNTIL THE PROCESSOR IS HALTED.
11300
11301 043614 104401 052752   ADJCLK: TYPE      ,OPR019          ;TYPE ADJUSTMENT INSTRUCTIONS
11302
11303 043620 012762 000040 000010   MOV      #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
11304 043626 013762 001626 000010   MOV      DRVNUM,RKCS2(R2) ;SELECT DRIVE
11305 043634 012762 000001 000000   MOV      #1,RKCS1(R2)
11306 043642 032762 000200 000000   5$: BIT      #RDY,RKCS1(R2) ;WAIT FOR READY
11307 043650 001774          BEQ      5$
11308 043652 032737 100000 001664   BIT      #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
11309 043660 001402          BEQ      1$          ;NO - SKIP
11310 043662 005077 136022          CLR      @KWLADD      ;CLEAR INTERRUPT ENABLE
11311
11312 043666 012762 000040 000026   1$: MOV      #DMD,RKMR1(R2) ;SET DIAG MODE
11313 043674 012701 000014          MOV      #12.,R1      ;SET A COUNT
11314 043700 005301          2$: DEC      R1          ;DEC COUNT
11315 043702 001376          BNE      2$          ;LOOP UNTIL ZERO
11316 043704 012762 000000 000026   MOV      #0,RKMR1(R2) ;CLEAR MR1
11317 043712 012701 000014          MOV      #12.,R1      ;SET COUNT
11318 043716 005301          3$: DEC      R1          ;DEC COUNT
11319 043720 001376          BNE      3$          ;LOOP UNTIL ZERO
11320 043722 000761          BR       1$          ;RESTART LOOP
11321
11322          .SBTTL CONTROLLED HALT SUBROUTINE
11323          ;* THIS ROUTINE IS ENTERED WHEN A CONTROL C IS TYPED. THE
11324          ;* SUBSYSTEM IS CLEARED, THE DRIVE IS RECALIBRATED, AND, IF
11325          ;* NECESSARY, CERTAIN CYLINDERS ARE REFORMATED. THE REFORMATTING
11326          ;* IS CONTROLLED BY THE LOCATION REFMT WHICH CONTAINS THE ADDRESS
11327          ;* OF THE CYLINDER TO BE REFORMATTED.
11328
11329 043724 012737 000111 001302   CTRHLT: MOV      #$TN,$TESTN      ;SET UP FOR HALT FAIL
11330
11331 043732 104416          TSSINIT          ;CLEAR SUBSYSTEM
11332 043734 104003          ERROR 3          ;BAD INIT ERROR
11333
11334 043736 113700 001102          MOV      $TSTNM,R0    ;GET CURRENT TEST NUMBER
11335 043742 042700 177400          BIC      #177400,R0   ;CLEAR UNUSED BITS
```

11336	043746	022700	000003	CMP	#3,RO	:TEST IF TEST NUMBER 3
11337	043752	001464		BEQ	PROGEND	:GO TO HALT PROG
11338	043754	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
11339	043760	000113		RECAL		:RECAL
11340	043762	000000		0		:0 WORDS
11341	043764	000000		0		:0 IS BUFF ADDRESS
11342	043766	000		.BYTE	0	:SECTOR 0
11343	043767	000		.BYTE	0	:TRACK 0
11344	043770	000000		0		:CYLINDER 0
11345						
11346	043772	104417		TLOADRK		:LOAD RK REGS
11347	043774	104423		TWAT16		:WAIT FOR INTERRUPT
11348	043776	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
11349						
11350	044000	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
11351	044002	104004		ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
11352						
11353	044004	005037	001662	CLR	INTSET	:CLEAR INTERRUPT FLAG
11354	044010	104437		TWAT8S		:WAIT FOR SECOND INTERRUPT
11355	044012	104002		ERROR	2	
11356						
11357	044014	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
11358	044016	104004		ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
11359						
11360	044020	104416		TSSINIT		:CLEAR SUBSYSTEM
11361	044022	104003		ERROR	3	:BAD INIT ERROR
11362						
11363	044024	005737	001676	TST	REFMT	:TEST IF REFORMAT REQUIRED
11364	044030	001435		BEQ	PROGEND	:NO - GO TO HALT
11365	044032	104401	053115	TYPE	,OPR020	:TYPE MESSAGE
11366						
11367	044036	004437	035674	JSR	R4,LRLOAD	:LOAD 'L' REGS
11368	044042	000127		WRHEAD		:WRHEAD
11369	044044	177676		-102		: -102 WORDS
11370	044046	072414		OBUFF		:OBUFF IS BUFF ADDRESS
11371	044050	000		.BYTE	0	:SECTOR 0
11372	044051	000		.BYTE	0	:TRACK 0
11373	044052	000312		312		:CYLINDER 312
11374						
11375	044054	005737	001676	TST	REFMT	:TEST IF CYL 0
11376	044060	100002		BPL	5\$:NO - SKIP
11377	044062	005037	001614	CLR	L.DCYL	:ELSE LOAD FOR CYL 0
11378	044066	004437	042260	JSR	R4,GENCOM	:GENERATE HEADERS
11379	044072	001200		1200		
11380						
11381	044074	104417		TLOADRK		:LOAD RK REGS
11382	044076	104434		TWAT159		:WAIT FOR INTERRUPT
11383	044100	104002		ERROR	2	:TO SLOW/NOT COMPLETE ERROR
11384						
11385	044102	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
11386	044104	104004		ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
11387						
11388	044106	122737	000002 001607	CMPB	#2,L.DT	:TEST IF TRACK 2 FORMATTED
11389	044114	001403		BEQ	PROGEND	:YES - SKIP
11390	044116	105237	001607	INCB	L.DT	:ELSE BUMP TRACK
11391	044122	000761		BR	5\$:DO NEXT TRACK

```
11392
11393 044124 104401 053211
11394 044130 012706 001100
11395 044134 105037 001103
11396 044140 005037 001264
11397 044144 005737 000042
11398 044150 001404
11399 044152 005037 032020
11400 044156 000137 031772
11401
11402 044162 000000
11403 044164 000137 001776
11404
11405
11406
11407
11408 044170 000240
11409 044172 104401 053252
11410 044176 000137 044124
11411
11412
11413
11414
11415
11416
11417
11418
11419
11420
11421
11422
11423
11424
11425
11426
11427
11428 044202 105737 001157
11429 044206 100002
11430 044210 000000
11431 044212 000430
11432 044214 010046
11433 044216 017600 000002
11434 044222 122737 000001 001316
11435 044230 001011
11436 044232 132737 000100 001317
11437 044240 001405
11438 044242 010037 044252
11439 044246 004737 033354
11440 044252 000000
11441 044254 132737 000040 001317
11442 044262 001003
11443 044264 112046
11444 044266 001005
11445 044270 005726
11446 044272 012600
11447 044274 062716 000002
```

```
PROGEND:      TYPE      ,OPR021 ;TYPE HALT MESSAGE
              MOV      #STACK,SP ;CLEAR STACK
              CLRB     $ERFLG     ;CLEAR ERROR FLAG
              CLR      $ESCAPE    ;CLEAR ESCAPE
              TST      @#42       ;TEST IF MONITOR PRESENT
              BEQ      10$        ;NO - SKIP
              CLR      $EOPCT     ;SET FOR END OF PROGRAM
              JMP      $EOP       ;GO TO END OF PASS

10$:          HALT          ;HALT PROGRAM
              JMP      START1    ;GO TO RESTART IF CONTINUE

.SBTTL HALT FAIL ROUTINE
;*      THIS ROUTINE IS ENTERED IF A HARDWARE ERROR IS DETECTED WHEN
;*      THE CARTRIDGE IS BEING REFORMATTED PRIOR TO HALT.
ABTFAIL:     NOP
              TYPE      ,OPR022  ;TYPE HALT FAIL MESSAGE
              JMP      PROGEND    ;GO STOP PROGRAM

.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      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;*OR
;*      TYPE
;*      MESADR
;*

$TYPE:      TSTB     $TPFLG     ;;IS THERE A TERMINAL?
              BPL     1$        ;;BR IF YES
              HALT    ;HALT HERE IF NO TERMINAL
              BR      3$        ;;LEAVE
1$:          MOV     R0,-(SP)    ;;SAVE R0
              MOV     @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
              CMPB   #APTENV,$ENV ;;RUNNING IN APT MODE
              BNE    62$        ;;NO,GO CHECK FOR APT CONSOLE
              BITB   #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
              BEQ    62$        ;;NO,GO CHECK FOR CONSOLE
              MOV    R0,61$     ;;SETUP MESSAGE ADDRESS FOR APT
              JSR   PC,$ATY3    ;;SPOOL MESSAGE TO APT
              .WORD  0          ;;MESSAGE ADDRESS
              BITB   #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
              BNE    60$        ;;YES,SKIP TYPE OUT
              MOVB   (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
              BNE    4$         ;;BR IF IT ISN'T THE TERMINATOR
              TST   (SP)+       ;;IF TERMINATOR POP IT OFF THE STACK
              MOV    (SP)+,R0    ;;RESTORE R0
              ADD    #2,(SP)    ;;ADJUST RETURN PC

61$:         .WORD  0
62$:         BITB   #APTCSUP,$ENVM
              BNE    60$
2$:          MOVB   (R0)+,-(SP)
              BNE    4$
60$:         MOV    (SP)+,R0
3$:          ADD    #2,(SP)
```

```
11448 044300 000002 RTI ;;RETURN
11449 044302 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
11450 044306 001430 BEQ 8$
11451 044310 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
11452 044314 001006 BNE 5$
11453 044316 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
11454 044320 104401 TYPE ;;TYPE A CR AND LF
11455 044322 001273 $CRLF
11456 044324 105037 044532 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
11457 044330 000755 BR 2$ ;;GET NEXT CHARACTER
11458 044332 004737 044414 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
11459 044336 123726 001156 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
11460 044342 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
11461 044344 013746 001154 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
11462 ;;AND THE NULL CHAR.
11463 044350 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
11464 044354 002770 BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
11465 044356 004737 044414 JSR PC,$TYPEC ;;GO TYPE A NULL
11466 044362 105337 044532 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
11467 044366 000770 BR 7$ ;;LOOP
11468
11469 ;HORIZONTAL TAB PROCESSOR
11470
11471 044370 112716 000040 8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
11472 044374 004737 044414 9$: JSR PC,$TYPEC ;;TYPE A SPACE
11473 044400 132737 000007 044532 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
11474 044406 001372 BNE 9$ ;;TAB STOP
11475 044410 005726 TST (SP)+ ;;POP SPACE OFF STACK
11476 044412 000724 BR 2$ ;;GET NEXT CHARACTER
11477 044414 $TYPEC:
11478 044414 105777 134524 TSTB @$TKS ;;CHAR IN KYBD BUFFER?
11479 044420 100022 BPL 10$ ;;BR IF NOT
11480 044422 017746 134520 MOV @$TKB,-(SP) ;;GET CHAR
11481 044426 042716 177600 BIC #177600,(SP) ;;STRIP EXTRANEIOUS BITS
11482 044432 122716 000023 CMPB #$XOFF,(SP) ;;WAS CHAR XOFF
11483 044436 001012 BNE 102$ ;;BR IF NOT
11484 044440 101$:
11485 044440 105777 134500 TSTB @$TKS ;;WAIT FOR CHAR
11486 044444 100375 BPL 101$
11487 044446 117716 134474 MOVB @$TKB,(SP) ;;GET CHAR
11488 044452 042716 177600 BIC #177600,(SP) ;;STRIP IT
11489 044456 122716 000021 CMPB #$XON,(SP) ;;WAS IT XON?
11490 044462 001366 BNE 101$ ;;BR IF NOT
11491 044464 102$:
11492 044464 005726 TST (SP)+ ;;FIX STACK
11493 044466 10$:
11494 044466 105777 134456 TSTB @$TPS ;;WAIT UNTIL PRINTER IS READY
11495 044472 100375 BPL 10$
11496 044474 116677 000002 134450 MOVB 2(SP),@$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11497 044502 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
11498 044510 001003 BNE 1$ ;;BRANCH IF NO
11499 044512 105037 044532 CLRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
11500 044516 000406 BR $TYPEX ;;EXIT
11501 044520 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
11502 044526 001402 BEQ $TYPEX ;;BRANCH IF YES
11503 044530 105227 INCB (PC)+ ;;COUNT THE CHARACTER
```

```
11504 044532 000000 $CHARCNT: .WORD 0          ;; CHARACTER COUNT STORAGE
11505 044534 000207 $TYPEX: RTS PC
11506
11507 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11508
11509 ;;*****
11510 ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11511 ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11512 ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11513 ;;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11514 ;;*REPLACED WITH SPACES.
11515 ;;*CALL:
11516 ;;*      MOV      NUM,-(SP)          ;;PUT THE BINARY NUMBER ON THE STACK
11517 ;;*      TYPDS          ;;GO TO THE ROUTINE
11518
11519 044536 $TYPDS:
11520 044536 010046      MOV      R0,-(SP)          ;;PUSH R0 ON STACK
11521 044540 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
11522 044542 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
11523 044544 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
11524 044546 010546      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
11525 044550 012746 020200      MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
11526 044554 016605 000020      MOV      20(SP),R5        ;;GET THE INPUT NUMBER
11527 044560 100004      BPL      1$              ;;BR IF INPUT IS POS.
11528 044562 005405      NEG      R5              ;;MAKE THE BINARY NUMBER POS.
11529 044564 112766 000055 000001      MOV      #'-,1(SP)        ;;MAKE THE ASCII NUMBER NEG.
11530 044572 005000      1$:      CLR      R0              ;;ZERO THE CONSTANTS INDEX
11531 044574 012703 044752      MOV      #$DBLK,R3        ;;SETUP THE OUTPUT POINTER
11532 044600 112723 000040      MOV      #' ,(R3)+        ;;SET THE FIRST CHARACTER TO A BLANK
11533 044604 005002      2$:      CLR      R2              ;;CLEAR THE BCD NUMBER
11534 044606 016001 044742      MOV      $DTBL(R0),R1     ;;GET THE CONSTANT
11535 044612 160105      3$:      SUB      R1,R5          ;;FORM THIS BCD DIGIT
11536 044614 002402      BLT      4$              ;;BR IF DONE
11537 044616 005202      INC      R2              ;;INCREASE THE BCD DIGIT BY 1
11538 044620 000774      BR      3$
11539 044622 060105      4$:      ADD      R1,R5          ;;ADD BACK THE CONSTANT
11540 044624 005702      TST      R2              ;;CHECK IF BCD DIGIT=0
11541 044626 001002      BNE      5$              ;;FALL THROUGH IF 0
11542 044630 105716      TSTB     (SP)            ;;STILL DOING LEADING 0'S?
11543 044632 100407      BMI      7$              ;;BR IF YES
11544 044634 106316      5$:      ASLB     (SP)            ;;MSD?
11545 044636 103003      BCC      6$              ;;BR IF NO
11546 044640 116663 000001 177777      MOV      1(SP),-1(R3)     ;;YES--SET THE SIGN
11547 044646 052702 000060      BIS      #'0,R2          ;;MAKE THE BCD DIGIT ASCII
11548 044652 052702 000040      6$:      BIS      #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11549 044656 110223      MOV      R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11550 044660 005720      TST      (R0)+          ;;JUST INCREMENTING
11551 044662 020027 000010      CMP      R0,#10         ;;CHECK THE TABLE INDEX
11552 044666 002746      BLT      2$              ;;GO DO THE NEXT DIGIT
11553 044670 003002      BGT      8$              ;;GO TO EXIT
11554 044672 010502      MOV      R5,R2          ;;GET THE LSD
11555 044674 000764      BR      6$              ;;GO CHANGE TO ASCII
11556 044676 105726      8$:      TSTB     (SP)+          ;;WAS THE LSD THE FIRST NON-ZERO?
11557 044700 100003      BPL      9$              ;;BR IF NO
11558 044702 116663 177777 177776      MOV      -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11559 044710 105013      9$:      CLRB     (R3)          ;;SET THE TERMINATOR
```

```
11560 044712 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
11561 044714 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
11562 044716 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
11563 044720 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11564 044722 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
11565 044724 104401 044752  TYPE      $DBLK      ;;NOW TYPE THE NUMBER
11566 044730 016666 000002 000004  MOV      2(SP),4(SP)   ;;ADJUST THE STACK
11567 044736 012616      MOV      (SP)+,(SP)
11568 044740 000002      RTI                          ;;RETURN TO USER
11569 044742 023420      $DTBL: 10000.
11570 044744 001750      1000.
11571 044746 000144      100.
11572 044750 000012      10.
11573 044752 000004      $DBLK: .BLKW 4
11574      .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
11575
11576      ;*****
11577      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11578      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11579      ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11580      ;*CALL:
11581      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11582      ;*      TYPOS      ;;CALL FOR TYPEOUT
11583      ;*      .BYTE  N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11584      ;*      .BYTE  M      ;;M=1 OR 0
11585      ;*                          ;;1=TYPE LEADING ZEROS
11586      ;*                          ;;0=SUPPRESS LEADING ZEROS
11587      ;*
11588      ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11589      ;*$TYPOS OR $TYPOC
11590      ;*CALL:
11591      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11592      ;*      TYPON      ;;CALL FOR TYPEOUT
11593      ;*
11594      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11595      ;*CALL:
11596      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11597      ;*      TYPOC      ;;CALL FOR TYPEOUT
11598      ;*
11599 044762 017646 000000  $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
11600 044766 116637 000001 045205  MOVB     1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
11601 044774 112637 045207  MOVB     (SP)+,$OMODE+1    ;;NUMBER OF DIGITS TO TYPE
11602 045000 062716 000002  ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
11603 045004 000406  BR      $TYPON
11604 045006 112737 000001 045205  $TYPOC: MOVB     #1,$OFILL      ;;SET THE ZERO FILL SWITCH
11605 045014 112737 000006 045207  MOVB     #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
11606 045022 112737 000005 045204  $TYPON: MOVB     #5,$OCNT      ;;SET THE ITERATION COUNT
11607 045030 010346  MOV      R3,-(SP)        ;;SAVE R3
11608 045032 010446  MOV      R4,-(SP)        ;;SAVE R4
11609 045034 010546  MOV      R5,-(SP)        ;;SAVE R5
11610 045036 113704 045207  MOVB     $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
11611 045042 005404  NEG      R4
11612 045044 062704 000006  ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
11613 045050 110437 045206  MOVB     R4,$OMODE        ;;SAVE IT FOR USE
11614 045054 113704 045205  MOVB     $OFILL,R4        ;;GET THE ZERO FILL SWITCH
11615 045060 016605 000012  MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
```



```
11616 045064 005003          CLR      R3          ;;CLEAR THE OUTPUT WORD
11617 045066 006105          1$:    ROL      R5          ;;ROTATE MSB INTO 'C'
11618 045070 000404          BR       3$          ;;GO DO MSB
11619 045072 006105          2$:    ROL      R5          ;;FORM THIS DIGIT
11620 045074 006105          ROL      R5
11621 045076 006105          ROL      R5
11622 045100 010503          MOV      R5,R3
11623 045102 006103          3$:    ROL      R3          ;;GET LSB OF THIS DIG.T
11624 045104 105337 045206    DECB     $OMODE      ;;TYPE THIS DIGIT?
11625 045110 100016          BPL      7$          ;;BR IF NO
11626 045112 042703 177770    BIC      #177770,R3  ;;GET RID OF JUNK
11627 045116 001002          BNE      4$          ;;TEST FOR 0
11628 045120 005704          TST      R4          ;;SUPPRESS THIS 0?
11629 045122 001403          BEQ      5$          ;;BR IF YES
11630 045124 005204          4$:    INC      R4          ;;DON'T SUPPRESS ANYMORE 0'S
11631 045126 052703 000060    BIS      #'0,R3     ;;MAKE THIS DIGIT ASCII
11632 045132 052703 000040    BIS      #' ,R3     ;;MAKE ASCII IF NOT ALREADY
11633 045136 110337 045202    MOV      R3,8$      ;;SAVE FOR TYPING
11634 045142 104401 045202    TYPE     ,8$        ;;GO TYPE THIS DIGIT
11635 045146 105337 045204          7$:    DECB     $OCNT      ;;COUNT BY 1
11636 045152 003347          BGT      2$          ;;BR IF MORE TO DO
11637 045154 002402          BLT      6$          ;;BR IF DONE
11638 045156 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
11639 045160 000744          BR       2$          ;;GO DO THE LAST DIGIT
11640 045162 012605          6$:    MOV      (SP)+,R5  ;;RESTORE R5
11641 045164 012604          MOV      (SP)+,R4  ;;RESTORE R4
11642 045166 012603          MOV      (SP)+,R3  ;;RESTORE R3
11643 045170 016666 000002 000004    MOV      2(SP),4(SP) ;;SET THE STACK FOR RETURNING
11644 045176 012616          MOV      (SP)+,(SP)
11645 045200 000002          RTI
11646 045202 000          8$:    .BYTE    0          ;;RETURN
11647 045203 000          .BYTE    0          ;;STORAGE FOR ASCII DIGIT
11648 045204 000          $OCNT:  .BYTE    0          ;;TERMINATOR FOR TYPE ROUTINE
11649 045205 000          $OFILL: .BYTE    0          ;;OCTAL DIGIT COUNTER
11650 045206 000000          $OMODE: .WORD    0          ;;ZERO FILL SWITCH
11651          .SBTTL  TTY INPUT ROUTINE  ;;NUMBER OF DIGITS TO TYPE
11652
11653          ;;*****
11654          .ENABL  LSB
11655 045210 000000          $TKCNT: .WORD    0          ;;NUMBER OF ITEMS IN QUEUE
11656 045212 000000          $TKQIN: .WORD    0          ;;INPUT POINTER
11657 045214 000000          $TKQOUT: .WORD   0          ;;OUTPUT POINTER
11658 045216 000001          $TKQSRT: .BLKB   1          ;;TTY KEYBOARD QUEUE
11659          $TKQEND=.
11660          .EVEN
11661
11662          ;*TK INITIALIZE ROUTINE
11663          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
11664          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
11665          ;
11666          ;*CALL:
11667          ;*   JSR      PC,$TKINT
11668          ;*   RETURN
11669
11670 045220 005037 045210          $TKINT: CLR      $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
11671 045224 012737 045216 045212    MOV      #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
```

```

11672 045232 013737 045212 045214      MOV      $TKQIN,$TKQOUT  ::QUEUE INTO THE INPUT & OUTPUT POINTERS.
11673 045240 012737 045270 000060      MOV      #$TKSRV,@#TKVEC  ::INITIALIZE THE KEYBOARD VECTOR
11674 045246 012737 000200 000062      MOV      #200,@#TKVEC+2  ::'BR' LEVEL 4
11675 045254 005777 133666      TST      @$TKB           ::CLEAR DONE FLAG
11676 045260 012777 000100 133656      MOV      #100,@$TKS      ::ENABLE TTY KEYBOARD INTERRUPT
11677 045266 000207      RTS      PC              ::RETURN TO CALLER
11678
11679
11680      ;*TK SERVICE ROUTINE
11681      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
11682      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
11683      ;*IT IN THE QUEUE.
11684      ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
11685      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
11686 045270 117746 133652      $TKSRV: MOVB    @$TKB,-(SP)  ::PICKUP THE CHARACTER
11687 045274 042716 177600      BIC      #^C177,(SP)      ::STRIP THE JUNK
11688 045300 042716 177600      BIC      #^C177,(SP)      ::STRIP THE JUNK
11689 045304 021627 000021      CMP      (SP),#$XON      ::IS IT A RANDOM XON?
11690 045310 001002      BNE      30$             ::BRANCH IF NO
11691 045312 005726      TST      (SP)+          ::CLEAN RANDOM XON OFF STACK
11692 045314 000002      RTI
11693 045316      30$:
11694 045316 021627 000003      CMP      (SP),#3        ::IS IT A CONTROL C?
11695 045322 001007      BNE      1$             ::BRANCH IF NO
11696 045324 104401 046422      TYPE    ,%CNTLC        ::TYPE A CONTROL-C (^C)
11697 045330 004737 045220      JSR     PC,$TKINT      ::INIT THE KEYBOARD
11698 045334 005726      TST      (SP)+          ::CLEAN UP STACK
11699 045336 000137 043724      JMP     CTRHLT         ::CONTROL C RESTART
11700 045342 021627 000007      1$:  CMP      (SP),#7        ::IS IT A CONTROL G?
11701 045346 001004      BNE      2$             ::BRANCH IF NO
11702 045350 022737 000176 001140      CMP      #SWREG,SWR     ::IS SOFT-SWR SELECTED?
11703 045356 001500      BEQ     6$             ::GO TO SWR CHANGE
11704
11705 045360      2$:
11706 045360 022737 000001 045210      CMP      #1,$TKCNT     ::IS THE QUEUE FULL?
11707 045366 001004      BNE      3$             ::BRANCH IF NO
11708 045370 104401 001266      TYPE    ,%BELL        ::RING THE TTY BELL
11709 045374 005726      TST      (SP)+          ::CLEAN CHARACTER OFF OF STACK
11710 045376 000451      BR      5$             ::EXIT
11711 045400 021627 000023      3$:  CMP      (SP),#23      ::IS IT A CONTROL-S?
11712 045404 001021      BNE      32$           ::BRANCH IF NO
11713 045406 005077 133532      CLR     @$TKS          ::DISABLE TTY KEYBOARD INTERRUPTS
11714 045412 005726      TST      (SP)+          ::CLEAN CHAR OFF STACK
11715 045414 105777 133524      31$:  TSTB   @$TKS          ::WAIT FOR A CHAR
11716 045420 100375      BPL     31$           ::LOOP UNTIL ITS THERE
11717 045422 117746 133520      MOVB   @$TKB,-(SP)     ::GET THE CHARACTER
11718 045426 042716 177600      BIC    #^C177,(SP)     ::MAKE IT 7-BIT ASCII
11719 045432 022627 000021      CMP    (SP)+,#21      ::IS IT A CONTROL-Q?
11720 045436 001366      BNE    31$           ::BRANCH IF NO
11721 045440 012777 000100 133476      MOV    #100,@$TKS     ::REENABLE TTY KEYBOARD INTERRUPTS
11722 045446 000002      RTI
11723 045450 005237 045210      32$:  INC     $TKCNT        ::COUNT THIS CHARACTER
11724 045454 021627 000140      CMP    (SP),#140      ::IS IT UPPER CASE?
11725 045460 002405      BLT    4$             ::BRANCH IF YES
11726 045462 021627 000175      CMP    (SP),#175     ::IS IT A SPECIAL CHAR?
11727 045466 003002      BGT    4$             ::BRANCH IF YES

```

```
11728 045470 042716 000040      BIC    #40,(SP)      ;;MAKE IT UPPER CASE
11729 045474 112677 177512      4$:   MOVB   (SP)+,@$TKQIN  ;;AND PUT IT IN QUEUE
11730 045500 005237 045212      INC    $TKQIN        ;;UPDATE THE POINTER
11731 045504 023727 045212 045217  CMP    $TKQIN,$$TKQEND ;;GO OFF THE END?
11732 045512 001003          BNE    5$           ;;BRANCH IF NO
11733 045514 012737 045216 045212  MOV    $$TKQSR,$TKQIN ;;RESET THE POINTER
11734 045522 000002      5$:   RTI           ;;RETURN
```

```
11735
11736      ;:*****
11737      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
11738      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
11739      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
11740      ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
```

```
11741 045524 022737 000176 001140 $CKSWR: CMP    #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
11742 045532 001124          BNE    15$           ;;EXIT IF NOT
11743 045534 105777 133404          TSTB  @$TKS         ;;IS A CHAR WAITING?
11744 045540 100121          BPL    15$           ;;IF NOT, EXIT
11745 045542 117746 133400          MOVB  @$TKB,-(SP)    ;;YES
11746 045546 042716 177600          BIC   #^C177,(SP)   ;;MAKE IT 7-BIT ASCII
11747 045552 021627 000007          CMP   (SP),#7       ;;IS IT A CONTROL-G?
11748 045556 001300          BNE   2$            ;;IF NOT, PUT IT IN THE TTY QUEUE
11749
11750
11751
11752
11753
11754      ;:*****
```

```
11755      ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
11756      ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
11757      ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
```

```
11755 045560 123727 001134 000001 6$:   CMPB   $AUTOB,#1    ;;ARE WE RUNNING IN AUTO-MODE?
11756 045566 001674          BEQ    2$            ;;BRANCH IF YES
11757 045570 005726          TST   (SP)+        ;;CLEAR CONTROL-G OFF STACK
11758 045572 004737 045220          JSR   PC,$TKINT    ;;FLUSH THE TTY INPUT QUEUE
11759 045576 005077 133342          CLR   @$TKS        ;;DISABLE TTY KEYBOARD INTERRUPTS
11760 045602 112737 000001 001135  MOVB  #1,$INTAG     ;;SET INTERRUPT MODE INDICATOR
```

```
11761
11762 045610 104401 046434      $GTSWR: TYPE   , $CNTLG    ;;ECHO THE CONTROL-G (^G)
11763 045614 104401 046441      TYPE   , $MSWR      ;;TYPE CURRENT CONTENTS
11764 045620 013746 000176      MOV    SWREG,-(SP)  ;;SAVE SWREG FOR TYPEOUT
11765 045624 104402          TYPOC             ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11766 045626 104401 046452      TYPE   , $MNEW     ;;PROMPT FOR NEW SWR
11767 045632 005046      19$:  CLR    -(SP)      ;;CLEAR COUNTER
11768 045634 005046          CLR    -(SP)      ;;THE NEW SWR
11769 045636 105777 133302      7$:   TSTB  @$TKS        ;;CHAR THERE?
11770 045642 100375          BPL    7$          ;;IF NOT TRY AGAIN
```

```
11771
11772 045644 117746 133276      MOVB  @$TKB,-(SP)  ;;PICK UP CHAR
11773 045650 042716 177600          BIC   #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
```

```
11774
11775 045654 021627 000003          CMP   (SP),#3      ;;IS IT A CONTROL-C?
11776 045660 001015          BNE   9$           ;;BRANCH IF NOT
11777 045662 104401 046422          TYPE  , $CNTLC    ;;YES, ECHO CONTROL-C (^C)
11778 045666 062706 000006          ADD   #6,SP        ;;CLEAN UP STACK
11779 045672 123727 001135 000001  CMPB  $INTAG,#1    ;;REENABLE TTY KEYBOARD INTERRUPTS?
11780 045700 001003          BNE   8$           ;;BRANCH IF NO
11781 045702 012777 000100 133234  MOV   #100,@$TKS   ;;ALLOW TTY KEYBOARD INTERRUPTS
11782 045710 000137 043724      8$:   JMP    CTRHLT     ;;CONTROL-C RESTART
11783
```

```
11784
11785 045714 021627 000025 9$: CMP (SP),#25 ::IS IT A CONTROL-U?
11786 045720 001005 BNE 10$ ::BRANCH IF NOT
11787 045722 104401 046427 TYPE ,SCNTLU ::YES, ECHO CONTROL-U (^U)
11788 045726 062706 000006 20$: ADD #6,SP ::IGNORE PREVIOUS INPUT
11789 045732 000737 BR 19$ ::LET'S TRY IT AGAIN
11790
11791
11792 045734 021627 000015 10$: CMP (SP),#15 ::IS IT A <CR>?
11793 045740 001022 BNE 16$ ::BRANCH IF NO
11794 045742 005766 000004 TST 4(SP) ::YES, IS IT THE FIRST CHAR?
11795 045746 001403 BEQ 11$ ::BRANCH IF YES
11796 045750 016677 000002 133162 MOV 2(SP),@SWR ::SAVE NEW SWR
11797 045756 062706 000006 11$: ADD #6,SP ::CLEAR UP STACK
11798 045762 104401 001273 14$: TYPE ,SCRLF ::ECHO <CR> AND <LF>
11799 045766 123727 001135 000001 CMPB $INTAG,#1 ::RE-ENABLE TTY KBD INTERRUPTS?
11800 045774 001003 BNE 15$ ::BRANCH IF NOT
11801 045776 012777 000100 133140 MOV #100,@$TKS ::RE-ENABLE TTY KBD INTERRUPTS
11802 046004 000002 RTI ::RETURN
11803 046006 004737 044414 15$: JSR PC,$TYPEC ::ECHO CHAR
11804 046012 021627 000060 16$: CMP (SP),#60 ::CHAR < 0?
11805 046016 002420 BLT 18$ ::BRANCH IF YES
11806 046020 021627 000067 CMP (SP),#67 ::CHAR > 7?
11807 046024 003015 BGT 18$ ::BRANCH IF YES
11808 046026 042726 000060 BIC #60,(SP)+ ::STRIP-OFF ASCII
11809 046032 005766 000002 TST 2(SP) ::IS THIS THE FIRST CHAR
11810 046036 001403 BEQ 17$ ::BRANCH IF YES
11811 046040 006316 ASL (SP) ::NO, SHIFT PRESENT
11812 046042 006316 ASL (SP) :: CHAR OVER TO MAKE
11813 046044 006316 ASL (SP) :: ROOM FOR NEW ONE.
11814 046046 005266 000002 17$: INC 2(SP) ::KEEP COUNT OF CHAR
11815 046052 056616 177776 BIS -2(SP),(SP) ::SET IN NEW CHAR
11816 046056 000667 BR 7$ ::GET THE NEXT ONE
11817 046060 104401 001272 18$: TYPE ,SQUES ::TYPE ?<CR><LF>
11818 046064 000720 BR 20$ ::SIMULATE CONTROL-U
11819 .DSABL LSB
11820
11821
11822 *****
11823 ::THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11824 ::CALL:
11825 :: RDCHR ::GET A CHARACTER FROM THE QUEUE
11826 :: RETURN HERE ::CHARACTER IS ON THE STACK
11827 :: ::WITH PARITY BIT STRIPPED OFF
11828
11829
11830 SRDCHR: MOV (SP),-(SP) ::PUSH DOWN THE PC AND
11831 046070 016666 000004 000002 MOV 4(SP),2(SP) ::THE PS
11832 046076 005066 000004 CLR 4(SP) ::GET READY FOR A CHARACTER
11833 046102 005046 CLR -(SP) ::PUT NEW PS ON STACK
11834 046104 012746 046112 MOV #64$,-(SP) ::PUT NEW PC ON STACK
11835 046110 000002 RTI ::POP NEW PC AND PS
11836 046112 64$:
11837 046112 005737 045210 1$: TST $TKCNT ::WAIT ON A CHARACTER
11838 046116 001775 BEQ 1$
11839 046120 005337 045210 DEC $TKCNT ::DECREMENT THE COUNTER
```

```
11840 046124 117766 177064 000004      MOVB   @STKQOUT,4(SP)  ;;GET ONE CHARACTER
11841 046132 005237 045214              INC    $TKQOUT        ;;UPDATE THE POINTER
11842 046136 023727 045214 045217      CMP    $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
11843 046144 001003              BNE    2$             ;;BRANCH IF NO
11844 046146 012737 045216 045214      MOV    #$TKQSRT,$TKQOUT ;;RESET THE POINTER
11845 046154 000002              RTI                    ;;RETURN
11846                                     ;;*****
11847                                     ;;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
11848                                     ;;*CALL:
11849                                     ;;*      RDLIN                ;;INPUT A STRING FROM THE TTY
11850                                     ;;*      RETURN HERE         ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
11851                                     ;;*                          ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
11852
11853 046156 010346      $RDLIN: MOV    R3,-(SP)      ;;SAVE R3
11854 046160 005046              CLR    -(SP)          ;;CLEAR THE RUBOUT KEY
11855 046162 012703 046412      1$:  MOV    #$TTYIN,R3    ;;GET ADDRESS
11856 046166 022703 046422      2$:  CMP    #$TTYIN+8.,R3  ;;BUFFER FULL?
11857 046172 101456              BLOS   4$             ;;BR IF YES
11858 046174 104410              RDCHR  ;;GO READ ONE CHARACTER FROM THE TTY
11859 046176 112613              MOVB   (SP)+,(R3)     ;;GET CHARACTER
11860 046200 122713 000177      10$: CMPB   #177,(R3)    ;;IS IT A RUBOUT
11861 046204 001022              BNE    5$             ;;BR IF NO
11862 046206 005716              TST   (SP)           ;;IS THIS THE FIRST RUBOUT?
11863 046210 001007              BNE    6$             ;;BR IF NO
11864 046212 112737 000134 046410      MOVB   #' \,9$       ;;TYPE A BACK SLASH
11865 046220 104401 046410              TYPE  ,9$
11866 046224 012716 177777              MOV    #-1,(SP)      ;;SET THE RUBOUT KEY
11867 046230 005303      6$:  DEC    R3           ;;BACKUP BY ONE
11868 046232 020327 046412      CMP    R3,$$TTYIN    ;;STACK EMPTY?
11869 046236 103434              BLO   4$             ;;BR IF YES
11870 046240 111337 046410      MOVB   (R3),9$       ;;SETUP TO TYPEOUT THE DELETED CHAR.
11871 046244 104401 046410              TYPE  ,9$
11872 046250 000746              BR    2$             ;;GO READ ANOTHER CHAR.
11873 046252 005716      5$:  TST   (SP)           ;;RUBOUT KEY SET?
11874 046254 001406              BEQ   7$             ;;BR IF NO
11875 046256 112737 000134 046410      MOVB   #' \,9$       ;;TYPE A BACK SLASH
11876 046264 104401 046410              TYPE  ,9$
11877 046270 005016              CLR   (SP)           ;;CLEAR THE RUBOUT KEY
11878 046272 122713 000025      7$:  CMPB   #25,(R3)     ;;IS CHARACTER A CTRL U?
11879 046276 001003              BNE   8$             ;;BR IF NO
11880 046300 104401 046427              TYPE  ,$CNTLU        ;;TYPE A CONTROL 'U'
11881 046304 000726              BR    1$             ;;GO START OVER
11882 046306 122713 000022      8$:  CMPB   #22,(R3)    ;;IS CHARACTER A '^R'?
11883 046312 001011              BNE   3$             ;;BRANCH IF NO
11884 046314 105013              CLRB  (R3)           ;;CLEAR THE CHARACTER
11885 046316 104401 001273              TYPE  ,$CRLF         ;;TYPE A 'CR' & 'LF'
11886 046322 104401 046412              TYPE  ,$TTYIN        ;;TYPE THE INPUT STRING
11887 046326 000717              BR    2$             ;;GO PICKUP ANOTHER CHARACTER
11888 046330 104401 001272      4$:  TYPE  ,$QUES        ;;TYPE A '?'
11889 046334 000712              BR    1$             ;;CLEAR THE BUFFER AND LOOP
11890 046336 111337 046410      3$:  MOVB   (R3),9$     ;;ECHO THE CHARACTER
11891 046342 104401 046410              TYPE  ,9$
11892 046346 122723 000015              CMPB   #15,(R3)+    ;;CHECK FOR RETURN
11893 046352 001305              BNE   2$             ;;LOOP IF NOT RETURN
11894 046354 105063 177777              CLRB  -1(R3)        ;;CLEAR RETURN (THE 15)
11895 046360 104401 001274              TYPE  ,.LF          ;;TYPE A LINE FEED
```

```

11896 046364 005726          TST      (SP)+          ;;CLEAN RUBOUT KEY FROM THE STACK
11897 046366 012603          MOV      (SP)+,R3      ;;RESTORE R3
11898 046370 011646          MOV      (SP),-(SP)    ;;ADJUST THE STACK AND PUT ADDRESS OF THE
11899 046372 016666 000004 000002  MOV      4(SP),2(SP)    ;; FIRST ASCII CHARACTER ON IT
11900 046400 012766 046412 000004  MOV      #$TTYIN,4(SP)
11901 046406 000002          RTI                    ;;RETURN
11902 046410 000          9$: .BYTE 0             ;;STORAGE FOR ASCII CHAR. TO TYPE
11903 046411 000          .BYTE 0             ;;TERMINATOR
11904 046412 000010          $TTYIN: .BLKB 8        ;;RESERVE 8 BYTES FOR TTY INPUT
11905 046422 041536 005015 000          $CNTLC: .ASCIZ /^C/<15><12> ;;CONTROL 'C'
11906 046427 136 006525 000012 $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
11907 046434 043536 005015 000          $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
11908 046441 015 051412 051127 $MSWR: .ASCIZ <15><12>/SWR = /
11909 046446 036440 000040          $MNEW: .ASCIZ / NEW = /
11910 046452 020040 042516 020127
11911 046460 020075 000
11912 046464
11913 .EVEN
11914 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
11915
11916 ;;*****
11917 ;;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
11918 ;;*CHANGE IT TO BINARY.
11919 ;;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
11920 ;;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
11921 ;;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
11922 ;;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
11923 ;;*CALL:
11924 ;;* RDOCT ;;READ AN OCTAL NUMBER
11925 ;;* RETURN HERE ;;LOW ORDER BITS ARE ON TOP OF THE STACK
11926 ;;* ;;HIGH ORDER BITS ARE IN $HIOCT
11927 046464 011646          $RDOCT: MOV      (SP),-(SP) ;;PROVIDE SPACE FOR THE
11928 046466 016666 000004 000002  MOV      4(SP),2(SP)    ;;INPUT NUMBER
11929 046474 010046          MOV      R0,-(SP)     ;;PUSH R0 ON STACK
11930 046476 010146          MOV      R1,-(SP)     ;;PUSH R1 ON STACK
11931 046500 010246          MOV      R2,-(SP)     ;;PUSH R2 ON STACK
11932 046502 104411          1$: RDLIN           ;;READ AN ASCIZ LINE
11933 046504 012600          MOV      (SP)+,R0     ;;GET ADDRESS OF 1ST CHARACTER
11934 046506 010037 046612          MOV      R0,5$        ;;AND SAVE IT
11935 046512 005001          CLR      R1           ;;CLEAR DATA WORD
11936 046514 005002          CLR      R2
11937 046516 112046          2$: MOVB      (R0)+,-(SP) ;;PICKUP THIS CHARACTER
11938 046520 001420          BEQ      3$           ;;IF ZERO GET OUT
11939 046522 122716 000060          CMPB     #'0,(SP)     ;;MAKE SURE THIS CHARACTER
11940 046526 003026          BGT      4$           ;;IS AN OCTAL DIGIT
11941 046530 122716 000067          CMPB     #'7,(SP)
11942 046534 002423          BLT
11943 046536 006301          ASL      R1           ;;*2
11944 046540 006102          ROL      R2
11945 046542 006301          ASL      R1           ;;*4
11946 046544 006102          ROL      R2
11947 046546 006301          ASL      R1           ;;*8
11948 046550 006102          ROL      R2
11949 046552 042716 177770          BIC      #^(7,(SP)    ;;STRIP THE ASCII JUNK
11950 046556 062601          ADD      (SP)+,R1     ;;ADD IN THIS DIGIT
11951 046560 000756          BR       2$          ;;LOOP

```

```
11952 046562 005726          3$:   TST      (SP)+          ;;CLEAN TERMINATOR FROM STACK
11953 046564 010166 000012   MOV      R1,12(SP)        ;;SAVE THE RESULT
11954 046570 010237 046622   MOV      R2,$SHIOCT
11955 046574 012602          MOV      (SP)+,R2         ;;POP STACK INTO R2
11956 046576 012601          MOV      (SP)+,R1         ;;POP STACK INTO R1
11957 046600 012600          MOV      (SP)+,R0         ;;POP STACK INTO R0
11958 046602 000002          RTI                          ;;RETURN
11959 046604 005726          4$:   TST      (SP)+          ;;CLEAN PARTIAL FROM STACK
11960 046606 105010          (LRB    (R0)              ;;SET A TERMINATOR
11961 046610 104401          TYPE                          ;;TYPE UP THRU THE BAD CHAR.
11962 046612 000000          5$:   .WORD    0
11963 046614 104401 001272   TYPE    ,SQUES            ;;'"?' 'CR' & 'LF'
11964 046620 000730          BR      1$                ;;TRY AGAIN
11965 046622 000000          $SHIOCT: .WORD 0          ;;HIGH ORDER BITS GO HERE
11966                                     .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11967
11968                                     ;*****
11969                                     ;*SAVE R0-R5
11970                                     ;*CALL:
11971                                     ;*   SAVREG
11972                                     ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11973                                     ;*
11974                                     ;*TOP---(+16)
11975                                     ;* +2---(+18)
11976                                     ;* +4---R5
11977                                     ;* +6---R4
11978                                     ;* +8---R3
11979                                     ;*+10---R2
11980                                     ;*+12---R1
11981                                     ;*+14---R0
11982
11983 046624          $SAVREG:
11984 046624 010046          MOV      R0,-(SP)         ;;PUSH R0 ON STACK
11985 046626 010146          MOV      R1,-(SP)         ;;PUSH R1 ON STACK
11986 046630 010246          MOV      R2,-(SP)         ;;PUSH R2 ON STACK
11987 046632 010346          MOV      R3,-(SP)         ;;PUSH R3 ON STACK
11988 046634 010446          MOV      R4,-(SP)         ;;PUSH R4 ON STACK
11989 046636 010546          MOV      R5,-(SP)         ;;PUSH R5 ON STACK
11990 046640 016646 000022   MOV      22(SP),-(SP)     ;;SAVE PS OF MAIN FLOW
11991 046644 016646 000022   MOV      22(SP),-(SP)     ;;SAVE PC OF MAIN FLOW
11992 046650 016646 000022   MOV      22(SP),-(SP)     ;;SAVE PS OF CALL
11993 046654 016646 000022   MOV      22(SP),-(SP)     ;;SAVE PC OF CALL
11994 046660 000002          RTI
11995
11996                                     ;*RESTORE R0-R5
11997                                     ;*CALL:
11998                                     ;*   RESREG
11999 046662          $RESREG:
12000 046662 012666 000022   MOV      (SP)+,22(SP)     ;;RESTORE PC OF CALL
12001 046666 012666 000022   MOV      (SP)+,22(SP)     ;;RESTORE PS OF CALL
12002 046672 012666 000022   MOV      (SP)+,22(SP)     ;;RESTORE PC OF MAIN FLOW
12003 046676 012666 000022   MOV      (SP)+,22(SP)     ;;RESTORE PS OF MAIN FLOW
12004 046702 012605          MOV      (SP)+,R5         ;;POP STACK INTO R5
12005 046704 012604          MOV      (SP)+,R4         ;;POP STACK INTO R4
12006 046706 012603          MOV      (SP)+,R3         ;;POP STACK INTO R3
12007 046710 012602          MOV      (SP)+,R2         ;;POP STACK INTO R2
```

```
12008 046712 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
12009 046714 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
12010 046716 000002          RTI
12011                               .SBTTL  POWER DOWN AND UP ROUTINES
12012
12013                               ;:*****
12014                               ;:POWER DOWN ROUTINE
12015 046720 012737 047060 000024 $PWRDN: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12016 046726 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
12017 046734 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
12018 046736 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
12019 046740 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
12020 046742 010346          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
12021 046744 010446          MOV      R4,-(SP)          ;;PUSH R4 ON STACK
12022 046746 010546          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
12023 046750 017746 132164          MOV      @SWR,-(SP)        ;;PUSH @SWR ON STACK
12024 046754 010637 047064          MOV      SP,$SAVR6        ;;SAVE SP
12025 046760 012737 046772 000024      MOV      #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12026 046766 000000          HALT
12027 046770 000776          BR      .-2                ;;HANG UP
12028
12029                               ;:*****
12030                               ;:POWER UP ROUTINE
12031 046772 012737 047060 000024 $PWRUP: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12032 047000 013706 047064          MOV      $SAVR6,SP        ;;GET SP
12033 047004 005037 047064          CLR      $SAVR6          ;;WAIT LOOP FOR THE TTY
12034 047010 005237 047064          1$:   INC      $SAVR6        ;;WAIT FOR THE INC
12035 047014 001375          BNE     1$                ;;OF WORD
12036 047016 012677 132116          MOV      (SP)+,@SWR        ;;POP STACK INTO @SWR
12037 047022 012605          MOV      (SP)+,R5          ;;POP STACK INTO R5
12038 047024 012604          MOV      (SP)+,R4          ;;POP STACK INTO R4
12039 047026 012603          MOV      (SP)+,R3          ;;POP STACK INTO R3
12040 047030 012602          MOV      (SP)+,R2          ;;POP STACK INTO R2
12041 047032 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
12042 047034 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
12043 047036 012737 046720 000024      MOV      #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12044 047044 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
12045 047052 104401          TYPE
12046 047054 047066          $PWRMG: .WORD  $POWER      ;;REPORT THE POWER FAILURE
12047 047056 000002          RTI                        ;;POWER FAIL MESSAGE POINTER
12048 047060 000000          $ILLUP: HALT                ;;THE POWER UP SEQUENCE WAS STARTED
12049 047062 000776          BR      .-2                ;; BEFORE THE POWER DOWN WAS COMPLETE
12050 047064 000000          $SAVR6: 0                    ;;PUT THE SP HERE
12051 047066 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
12052 047074 000122
12053                               .EVEN
12054                               .SBTTL  TRAP DECODER
12055
12056                               ;:*****
12057                               ;:*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12058                               ;:*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12059                               ;:*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12060                               ;:*GO TO THAT ROUTINE.
12061
12062 047076 010046          $TRAP: MOV      R0,-(SP)      ;;SAVE R0
12063 047100 016600 000002          MOV      2(SP),R0          ;;GET TRAP ADDRESS
```



```

12064 047104 005740          TST      -(RO)          ;;BACKUP BY 2
12065 047106 111000          MOV      (RO),RO       ;;GET RIGHT BYTE OF TRAP
12066 047110 006300          ASL      RO            ;;POSITION FOR INDEXING
12067 047112 016000 047132  MOV      $TRPAD(RO),RO  ;;INDEX TO TABLE
12068 047116 000200          RTS      RO            ;;GO TO ROUTINE
12069
12070
12071          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12072
12073 047120 011646          $TRAP2: MOV      (SP),-(SP)  ;;MOVE THE PC DOWN
12074 047122 016666 000004 000002  MOV      4(SP),2(SP)  ;;MOVE THE PSW DOWN
12075 047130 000002          RTI                    ;;RESTORE THE PSW
12076
12077          .SBTTL TRAP TABLE
12078
12079          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12080          ;*BY THE "TRAP" INSTRUCTION.
12081
12082          :          ROUTINE
12083          :          -----
12084 047132 047120          $TRPAD: .WORD      $TRAP2
12085 047134 044202          $TYPE      ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
12086 047136 045006          $TYPOC     ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12087 047140 044762          $TYPOS     ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12088 047142 045022          $TYPON     ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
12089 047144 044536          $TYPDS     ;;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12090
12091 047146 045614          $GTSWR     ;;CALL=GTSWR     TRAP+6(104406)  GET SOFT-SWR SETTING
12092
12093 047150 045524          $CKSWR     ;;CALL=CKSWR     TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
12094 047152 046066          $RDCHR     ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
12095 047154 046156          $RDLIN     ;;CALL=RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
12096 047156 046464          $RDOCT     ;;CALL=RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
12097 047160 046624          $$SAVREG   ;;CALL=SAVREG     TRAP+13(104413) SAVE R0-R5 ROUTINE
12098 047162 046662          $RESREG    ;;CALL=RESREG     TRAP+14(104414) RESTORE R0-R5 ROUTINE
12099 047164 035222          $SCOPI$    ;;CALL=SCOPI     TRAP+15(104415) INTERNAL LOOP ON ERROR
12100 047166 037176          $SSINIT    ;;CALL=TSSINIT    TRAP+16(104416) INITIALIZE SUBSYSTEM
12101 047170 035734          $LOADRK    ;;CALL=TLOADRK   TRAP+17(104417) LOAD RK611 FOR OPERATION
12102 047172 036164          $GETRK     ;;CALL=TGETRK     TRAP+20(104420) GET RK611 REGISTERS
12103 047174 040162          $CHKOP     ;;CALL=TCHKOP     TRAP+21(104421) CHECK OPREATION FOR ANY ERRORS
12104 047176 040130          $CHKWE     ;;CALL=TCHKWE     TRAP+22(104422) CHECK OPERATION FOR EXPECTED ERRORS
12105 047200 035530          $IWAT16    ;;CALL=TWAT16     TRAP+23(104423) WAIT 16 MS
12106 047202 035520          $IWAT32    ;;CALL=TWAT32     TRAP+24(104424) WAIT 32 MS
12107 047204 035510          $IWAT48    ;;CALL=TWAT48     TRAP+25(104425) WAIT 48 MS
12108 047206 035500          $IWAT64    ;;CALL=TWAT64     TRAP+26(104426) WAIT 64 MS
12109 047210 035470          $IWAT80    ;;CALL=TWAT80     TRAP+27(104427) WAIT 80 MS
12110 047212 035460          $IWAT96    ;;CALL=TWAT96     TRAP+30(104430) WAIT 96 MS
12111 047214 035450          $IWAT112   ;;CALL=TWAT112    TRAP+31(104431) WAIT 112 MS
12112 047216 035440          $IWAT128   ;;CALL=TWAT128    TRAP+32(104432) WAIT 128 MS
12113 047220 035430          $IWAT144   ;;CALL=TWAT144    TRAP+33(104433) WAIT 144 MS
12114 047222 035420          $IWAT159   ;;CALL=TWAT159    TRAP+34(104434) WAIT 160 MS
12115 047224 035410          $IWAT1S    ;;CALL=TWAT1S     TRAP+35(104435) WAIT FOR 1 SECOND
12116 047226 035400          $IWAT2S    ;;CALL=TWAT2S     TRAP+36(104436) WAIT FOR 2 SECONDS
12117 047230 035360          $IWAT8S    ;;CALL=TWAT8S     TRAP+37(104437) WAIT FOR 8 SECONDS
12118 047232 035370          $IWAT1M    ;;CALL=TWAT1M     TRAP+40(104440) WAIT FOR 1 MIN
12119          $TERM=-$TRPAD

```

```
12120 .SBTTL DATA PATTERNS
12121 :DATA PATTERN 1
12122 : PATTERN IS ALL ZEROS
12123
12124 :DATA PATTERN 2
12125 : HI-LO FREQ. MIX
12126 e PAT02:
12127 047234 177777 177777
12128 047236 177777 177777
12129 047240 177777 177777
12130 047242 052525 052525
12131 047244 052525 052525
12132 047246 052525 052525
12133 047250 177777 177777
12134 047252 177777 177777
12135 047254 052525 052525
12136 047256 052525 052525
12137 047260 177777 177777
12138 047262 052525 052525
12139 047264 177252 177252
12140 047266 177252 177252
12141 047270 172765 172765
12142 047272 172765 172765
12143
12144 :DATA PATERN 3
12145 : HI FREQ. PHASE MIX
12146 PAT03:
12147 047274 000000 000000
12148 047276 000000 000000
12149 047300 000000 000000
12150 047302 177777 177777
12151 047304 177777 177777
12152 047306 177777 177777
12153 047310 000000 000000
12154 047312 000000 000000
12155 047314 177777 177777
12156 047316 177777 177777
12157 047320 000000 000000
12158 047322 177777 177777
12159 047324 000000 000000
12160 047326 177777 177777
12161 047330 000000 000000
12162 047332 177777 177777
12163
12164 :DATA PATTERN 4
12165 : LO FREQ. PHASE MIX
12166 PAT04:
12167 047334 052525 052525
12168 047336 052525 052525
12169 047340 052525 052525
12170 047342 125252 125252
12171 047344 125252 125252
12172 047346 125252 125252
12173 047350 052525 052525
12174 047352 052525 052525
12175 047354 125252 125252
```

12176	047356	125252	125252
12177	047360	052525	052525
12178	047362	125252	125252
12179	047364	052525	052525
12180	047366	125252	125252
12181	047370	052525	052525
12182	047372	125252	125252

:DATA PATTERN 5
: MAX PRECOMP. PHASE MIX

12183			
12184			
12185			
12186	047374		
12187	047374	133333	133333
12188	047376	066666	066666
12189	047400	155555	155555
12190	047402	155555	155555
12191	047404	133333	133333
12192	047406	066666	066666
12193	047410	066666	066666
12194	047412	155555	155555
12195	047414	155555	155555
12196	047416	133333	133333
12197	047420	133333	133333
12198	047422	133333	133333
12199	047424	133333	133333
12200	047426	133333	133333
12201	047430	133333	133333
12202	047432	133333	133333

:DATA PATTERN 6
: ROTATING BOUNDARY PULSE PRECOMP.

12203			
12204			
12205			
12206	047434		
12207	047434	121105	121105
12208	047436	150442	150442
12209	047440	064221	064221
12210	047442	132110	132110
12211	047444	055044	055044
12212	047446	026422	026422
12213	047450	013211	013211
12214	047452	105504	105504
12215	047454	042642	042642
12216	047456	021321	021321
12217	047460	110550	110550
12218	047462	044264	044264
12219	047464	022132	022132
12220	047466	011055	011055
12221	047470	104426	104426
12222	047472	042213	042213

:DATA PATTERN 7
: FIELD OF ALL ONES

:DATA PATTERN 10
: ROTATING CELL PULSE PRECOMP.

12223			
12224			
12225			
12226			
12227			
12228			
12229	047474		
12230	047474	026455	026455
12231	047476	113226	113226

12232	047500	045513	045513
12233	047502	122645	122645
12234	047504	151322	151322
12235	047506	064551	064551
12236	047510	132264	132264
12237	047512	055132	055132
12238	047514	026455	026455
12239	047516	113226	113226
12240	047520	045513	045513
12241	047522	122645	122645
12242	047524	151322	151322
12243	047526	064551	064551
12244	047530	132264	132264
12245	047532	055132	055132

:DATA PATTERN 11
: SHIFTED 1 IN A FIELD OF ZEROS

12246			
12247			
12248			
12249	047534		
12250	047534	000001	000001
12251	047536	000002	000002
12252	047540	000004	000004
12253	047542	000010	000010
12254	047544	000020	000020
12255	047546	000040	000040
12256	047550	000100	000100
12257	047552	000200	000200
12258	047554	000400	000400
12259	047556	001000	001000
12260	047560	002000	002000
12261	047562	004000	004000
12262	047564	010000	010000
12263	047566	020000	020000
12264	047570	040000	040000
12265	047572	100000	100000

:DATA PATTERN 12
: SHIFTED 0 IN A FIELD OF ONES

12266			
12267			
12268			
12269	047574		
12270	047574	177776	177776
12271	047576	177775	177775
12272	047600	177773	177773
12273	047602	177767	177767
12274	047604	177757	177757
12275	047606	177737	177737
12276	047610	177677	177677
12277	047612	177577	177577
12278	047614	177377	177377
12279	047616	176777	176777
12280	047620	175777	175777
12281	047622	173777	173777
12282	047624	167777	167777
12283	047626	157777	157777
12284	047630	137777	137777
12285	047632	077777	077777

:DATA PATTERN 13

12286
12287

12288			:	PAT13:	ALTERNATING 0-1
12289	047634		:		052525
12290	047634	052525	:		052525
12291	047636	052525	:		052525
12292	047640	052525	:		052525
12293	047642	052525	:		052525
12294	047644	052525	:		052525
12295	047646	052525	:		052525
12296	047650	052525	:		052525
12297	047652	052525	:		052525
12298	047654	052525	:		052525
12299	047656	052525	:		052525
12300	047660	052525	:		052525
12301	047662	052525	:		052525
12302	047664	052525	:		052525
12303	047666	052525	:		052525
12304	047670	052525	:		052525
12305	047672	052525	:		052525

12306			:		
12307			:		
12308			:		
12309	047674		:		
12310	047674	125252	:		
12311	047676	125252	:		
12312	047700	125252	:		
12313	047702	125252	:		
12314	047704	125252	:		
12315	047706	125252	:		
12316	047710	125252	:		
12317	047712	125252	:		
12318	047714	125252	:		
12319	047716	125252	:		
12320	047720	125252	:		
12321	047722	125252	:		
12322	047724	125252	:		
12323	047726	125252	:		
12324	047730	125252	:		
12325	047732	125252	:		

12326			:		
12327			:		
12328			:		
12329	047734		:		
12330	047734	000001	:		
12331	047736	000003	:		
12332	047740	000007	:		
12333	047742	000017	:		
12334	047744	000037	:		
12335	047746	000077	:		
12336	047750	000177	:		
12337	047752	000377	:		
12338	047754	000777	:		
12339	047756	001777	:		
12340	047760	003777	:		
12341	047762	007777	:		
12342	047764	017777	:		
12343	047766	037777	:		

12344	047770	077777
12345	047772	177777
12346		
12347		
12348		
12349	047774	
12350	047774	072307
12351	047776	135143
12352	050000	156461
12353	050002	167230
12354	050004	073514
12355	050006	035646
12356	050010	016723
12357	050012	107351
12358	050014	143564
12359	050016	061672
12360	050020	030735
12361	050022	114356
12362	050024	046167
12363	050026	123073
12364	050030	151453
12365	050032	164616
12366		

	077777
	177777
:DATA PATTERN 16	
: COMPOSITE ROTATING	
PAT16:	072307
	135143
	156461
	167230
	073514
	035646
	016723
	107351
	143564
	061672
	030735
	114356
	046167
	123073
	151453
	164616

```
12367          .SBTTL FIELDS AND VARIABLES FOR OPERATION CHECKING
12368          CS1ERBIT=24000          :CS1 ERROR BITS SPAR & CTO
12369          177400                  :CS2 ERROR BITS
12370          :DLT,WCE,UPE,NED,NEM
12371          :PGE,MOS,UFE
12372          000070                  DSERBIT=70          :DS ERROR BITS
12373          :SPDLSS,DROT,ACLO
12374
12375          050034 000000          EXPWC: .WORD 0          :EXPECTED WORD COUNT (GIVEN)
12376          050036 000000          EXPUBA: .WORD 0        :EXPECTED UPPER BA (COMPUTED)
12377          050040 000000          EXPBA: .WORD 0        :EXPECTED BUS ADDRESS (COMPUTED)
12378          050042 000000          EXPCYL: .WORD 0       :EXPECTED CYLINDER (COMPUTED)
12379          050044 000000          EXPSEC: .WORD 0       :EXPECTED SECTOR (COMPUTED)
12380          050046 000000          EXPTRK: .WORD 0       :EXPECTED TRACK (COMPUTED)
12381
12382          050050 000000          REALWC: .WORD 0       :WORD COUNT AT END OF OPERATION
12383          050052 000000          REALUB: .WORD 0       :REAL UPPER BA
12384          050054 000000          REALBA: .WORD 0       :BUS ADDRESS
12385          050056 000000          REALCY: .WORD 0       :CYLINDER
12386          050060 000000          REALTRK: .WORD 0      :TRACK
12387          050062 000000          REALSEC: .WORD 0      :SECTOR
12388
12389          050064 000000          GRP1ER: .WORD 0       :GROUP 1 ERROR FIELDS
12390          000001          SPARERR=BIT0          :CONTROLLER DETECTED DRIVE BUS PARITY ERR
12391          000002          SKIERR= BIT1          :SEEK INCOMPLETE
12392          000004          NXFERR= BIT2          :NON-EXECUTABLE DRIVE FUNCTION
12393          000010          DRPARERR=BIT3        :DRIVE DETECTED DRIVE BUS PARITY ERROR
12394          000020          FMTERR= BIT4          :FORMAT ERROR
12395          000040          DTYERR= BIT5          :DRIVE TYPE ERROR
12396          000100          ACLOERR=BIT6         :AC LOW ERROR
12397          000200          SPDERR= BIT7          :SPEED LOSS ERROR
12398          000400          DROTERR=BIT8         :DRIVE OFF TRACK ERROR
12399          001000          COERR= BIT9          :CYLINDER OVER FLOW ERROR
12400          002000          IDAERR= BIT10        :ILLEGAL DISK ADDRESS ERROR
12401          004000          WLERR= BIT11         :WRITE LOCK ERROR
12402          010000          DTERR= BIT12         :DRIVE TIMING ERROR
12403          020000          NCERWE= BIT13        :NO CERR WITH ERROR SET ERROR
12404          040000          UNSERR= BIT14        :DRIVE UNSAFE ERROR
12405          100000          CERNER= BIT15        :CERR BUT NO ERROR SET ERROR
12406
12407          050066 000000          GRP2ER: .WORD 0       :GROUP 2 ERROR FIELD
12408          000001          ECHERR= BIT0          :ECC HARD ERROR
12409          000002          DCKERR= BIT1          :DATA CHECK ERROR
12410          000004          WCKERR= BIT2          :WRITE CHECK ERROR
12411          000010          DLTERR= BIT3          :DATA LATE ERROR
12412          000020          OPIERR= BIT4          :OPERATION INCOMPLETE ERROR
12413          000040          HVRCERR=BIT5         :HEADER VRC ERROR
12414          000100          BSERR= BIT6          :BAD SECTOR ERROR
12415
12416          050070 000000          GRP3ER: .WORD 0       :GROUP 3 ERROR FIELD
12417          000001          NEDERR= BIT0          :NON-EXISTANT DRIVE ERROR
12418          000002          CTOERR= BIT1          :CONTROLLER TIME OUT ERROR
12419          000004          UFERR= BIT2          :UNIT FIELD ERROR
12420          000010          MDSERR= BIT3          :MULTIPLE DRIVE SELECT ERROR
12421          000020          PGERR= BIT4          :PROGRAMMING ERROR
12422          000040          NEMERR= BITS         :NON-EXISTANT MEMORY ERROR
```

12423		000100	UPERR= BIT6	:UNIBUS PARITY ERROR
12424		000200	ILFERR= BIT7	:ILLEGAL FUNCTION ERROR.
12425				
12426	050072	000000	GRP4ER: .WORD 0	:GROUP 4 ERROR FIELD
12427		000001	WCERR= BIT0	:WORD COUNT ERROR FLAG
12428		000002	UBAERR= BIT1	:UPPER BA ERROR
12429		000004	BAERR= BIT2	:BUS ADDRESS ERROR FLAG
12430		000010	CYLERR= BIT3	:CYL ADDRESS ERROR FLAG
12431		000020	TRKERR= BIT4	:TRACK ADDRESS ERROR FLAG
12432		000040	SECERR= BIT5	:SECTOR ADDRESS ERROR FLAG
12433				
12434	050074	054076	GRP4MS: .WORD EM10	
12435	050076	054151	.WORD EM11A	
12436	050100	054123	.WORD EM11	
12437	050102	054225	.WORD EM12	
12438	050104	054260	.WORD EM13	
12439	050106	054310	.WORD EM14	
12440				
12441	050110	054342	GRP3MS: .WORD EM15	
12442	050112	054365	.WORD EM16	
12443	050114	054410	.WORD EM17	
12444	050116	054431	.WORD EM18	
12445	050120	054456	.WORD EM19	
12446	050122	054500	.WORD EM20	
12447	050124	054524	.WORD EM21	
12448	050126	054550	.WORD EM22	
12449				
12450	050130	054571	GRP2MS: .WORD EM23	
12451	050132	054602	.WORD EM24	
12452	050134	054615	.WORD EM25	
12453	050136	054631	.WORD EM26	
12454	050140	054643	.WORD EM27	
12455	050142	054670	.WORD EM28	
12456	050144	054703	.WORD EM29	
12457				
12458	050146	054724	GRP1MS: .WORD EM30	
12459	050150	054777	.WORD EM31	
12460	050152	055017	.WORD EM32	
12461	050154	055055	.WORD EM33	
12462	050156	055123	.WORD EM34	
12463	050160	055140	.WORD EM35	
12464	050162	055161	.WORD EM36	
12465	050164	055170	.WORD EM37	
12466	050166	055213	.WORD EM38	
12467	050170	055233	.WORD EM39	
12468	050172	055255	.WORD EM40	
12469	050174	055307	.WORD EM41	
12470	050176	055330	.WORD EM42	
12471	050200	055353	.WORD EM43	
12472	050202	055415	.WORD EM44	
12473	050204	055432	.WORD EM45	
12474				
12475	050206	000000	GPSUMF: .WORD 0	:GROUP ERROR SUMMARY FLAGS
12476		000001	GRP1ST= BIT0	:GROUP 1 ERROR SET
12477		000002	GRP2ST= BIT1	:GROUP 2 ERROR SET
12478		000004	GRP3ST= BIT2	:GROUP 3 ERROR SET

12479	000010	GP1NR= BIT3	:GROUP 1 ERROR NOT RECEIVED
12480	000020	GP2NR= BIT4	:GROUP 2 ERROR NOT RECEIVED
12481	000040	GP3NR= BIT5	:GROUP 3 ERROR NOT RECEIVED
12482	040000	DRSTER= BIT14	:ERROR IN GETTING DRIVE STATUS FLAG.
12483	100000	REPNR= BIT15	:REPORTING NOT RECEIVED SWITCH

12484
12485 .SBTTL TABLE OF OPERATION MESSAGE ADDRESS
12486 :* THIS TABLE CONTAINS THE ADDRESS OF ASCIZ FIELDS THAT ARE
12487 :* USED IN REPORTING TO IDENTIFY THE OPERATION BEING PERFORMED.
12488

12489	050210	050250	CMNDLB: .WORD	OPER00	:ADDRESS OF SELECT MESSAGE
12490	050212	050265	.WORD	OPER02	PACK ACK
12491	050214	050276	.WORD	OPER04	DRIVE CLEAR
12492	050216	050312	.WORD	OPER06	UNLOAD
12493	050220	050321	.WORD	OPER10	START SPINDLE
12494	050222	050337	.WORD	OPER12	RECALIBRATE
12495	050224	050353	.WORD	OPER14	OFFSET
12496	050226	050362	.WORD	OPER16	SEEK
12497	050230	050367	.WORD	OPER20	READ DATA
12498	050232	050401	.WORD	OPER22	WRITE DATA
12499	050234	050414	.WORD	OPER24	READ HEADER
12500	050236	050431	.WORD	OPER26	WRITE HEADER
12501	050240	050447	.WORD	OPER30	WRITE CHECK
12502	050242	050463	.WORD	OPER32	:ILLEGAL OPERATION 33
12503	050244	050507	.WORD	OPER34	35
12504	050246	050533	.WORD	OPER36	37
12505					

12506 .SBTTL OPERATION MESSAGES

12507	050250	051104	053111	020105	OPER00: .ASCIZ /DRIVE SELECT/
12508	050256	042523	042514	052103	
12509	050264	000			
12510	050265	120	041501	020113	OPER02: .ASCIZ /PACK ACK/
12511	050272	041501	000113		
12512	050276	051104	053111	020105	OPER04: .ASCIZ /DRIVE CLEAR/
12513	050304	046103	040505	000122	
12514	050312	047125	047514	042101	OPER06: .ASCIZ /UNLOAD/
12515	050320	000			
12516	050321	123	040524	052122	OPER10: .ASCIZ /START SPINDLE/
12517	050326	051440	044520	042116	
12518	050334	042514	000		
12519	050337	122	041505	046101	OPER12: .ASCIZ /RECALIBRATE/
12520	050344	041111	040522	042524	
12521	050352	000			
12522	050353	117	043106	042523	OPER14: .ASCIZ /OFFSET/
12523	050360	000124			
12524	050362	042523	045505	000	OPER16: .ASCIZ /SEEK/
12525	050367	122	040505	020104	OPER20: .ASCIZ /READ DATA/
12526	050374	040504	040524	000	
12527	050401	127	044522	042524	OPER22: .ASCIZ /WRITE DATA/
12528	050406	042040	052101	000101	
12529	050414	042522	042101	044040	OPER24: .ASCIZ /READ HEADERS/
12530	050422	040505	042504	051522	
12531	050430	000			
12532	050431	127	044522	042524	OPER26: .ASCIZ /WRITE HEADERS/
12533	050436	044040	040505	042504	
12534	050444	051522	000		

12535	050447	127	044522	042524	OPER30: .ASCIZ /WRITE CHECK/
12536	050454	041440	042510	045503	
12537	050462	000			
12538	050463	111	046114	043505	OPER32: .ASCIZ /ILLEGAL FUNCTION 33/
12539	050470	046101	043040	047125	
12540	050476	052103	047511	020116	
12541	050504	031463	000		
12542	050507	111	046114	043505	OPER34: .ASCIZ /ILLEGAL FUNCTION 35/
12543	050514	046101	043040	047125	
12544	050522	052103	047511	020116	
12545	050530	032463	000		
12546	050533	111	046114	043505	OPER36: .ASCIZ /ILLEGAL FUNCTION 37/
12547	050540	046101	043040	047125	
12548	050546	052103	047511	020116	
12549	050554	033463	000		
12550	050557	127	044522	042524	OPER37: .ASCIZ /WRITE DATA ABORTED WITH BSE/
12551	050564	042040	052101	020101	
12552	050572	041101	051117	042524	
12553	050600	020104	044527	044124	
12554	050606	041040	042523	000	
12555	050613				OPER40:
12556	050613	127	044522	042524	OPER41: .ASCIZ /WRITE CHECK ABORTED WITH WCE/
12557	050620	041440	042510	045503	
12558	050626	040440	047502	052122	
12559	050634	042105	053440	052111	
12560	050642	020110	041527	000105	
12561	050650	051127	052111	020105	OPER42: .ASCIZ /WRITE DATA ABORTED WITH NON-EXISTANT MEMORY/
12562	050656	040504	040524	040440	
12563	050664	047502	052122	042105	
12564	050672	053440	052111	020110	
12565	050700	047516	026516	054105	
12566	050706	051511	040524	052116	
12567	050714	046440	046505	051117	
12568	050722	000131			
12569	050724	042522	042101	042040	OPER43: .ASCIZ /READ DATA ABORTED WITH NON-EXISTANT MEMORY/
12570	050732	052101	020101	041101	
12571	050740	051117	042524	020104	
12572	050746	044527	044124	047040	
12573	050754	047117	042455	044530	
12574	050762	052123	047101	020124	
12575	050770	042515	047515	054522	
12576	050776	000			
12577	050777	127	044522	042524	OPER44: .ASCIZ /WRITE DATA ABORTED WITH UNIBUS PARITY ERROR/
12578	051004	042040	052101	020101	
12579	051012	041101	051117	042524	
12580	051020	020104	044527	044124	
12581	051026	052440	044516	052502	
12582	051034	020123	040520	044522	
12583	051042	054524	042440	051122	
12584	051050	051117	000		
12585					

```
12586 .SBTTL ASCII MESSAGES
12587
12588 051053 040 000040 SPACE2: .ASCIZ / /
12589 051056 005015 045522 030466 OPR001: .ASCIZ <15><12>/RK611 BUS ADDRESS ( /
12590 051064 020061 052502 020123
12591 051072 042101 051104 051505
12592 051100 020123 020050 000
12593 051105 040 020051 020075 OPR002: .ASCIZ / ) = /
12594 051112 00
12595 051113 02 033113 030461 OPR003: .ASCIZ /RK611 VECTOR ADDRESS ( /
12596 051120 053040 041505 047524
12597 051126 020122 042101 051104
12598 051134 051505 020123 020050
12599 051142 000
12600 051143 122 033113 030461 OPR004: .ASCIZ /RK611 PRIORITY ( /
12601 051150 050040 044522 051117
12602 051156 052111 020131 020050
12603 051164 000
12604 051165 111 051516 043125 OPR005: .ASCIZ /INSUFFICIENT MEMORY. PROGRAM ABORTING./<15><12>
12605 051172 044506 044503 047105
12606 051200 020124 042515 047515
12607 051206 054522 020056 051120
12608 051214 043517 040522 020115
12609 051222 041101 051117 044524
12610 051230 043516 006456 000012
12611 051236
12612 OPR006:
: .ASCII <15><12>/TO BYPASS TESTING DRIVE 0, PLACE IT OFF-LINE/
12613 : .ASCIZ <15><12>/TO TEST DRIVE 0, REPLACE PROGRAM PACK WITH SCRATCH PACK/
12614 051236 OPR007:
: .ASCII <CR><LF>/CHANGE XXDP PACK/
12615 051236 005015 044103 047101 .ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
12616 051244 042507 054040 042130
12617 051252 020120 040520 045503
12618 051260 005015 046103 040505
12619 051266 020122 047514 020103
12620 051274 030064 051054 051505
12621 051302 040524 052122 050040
12622 051310 047522 051107 046501
12623 051316 000
12624 : .ASCII <15><12>/DRIVE 0 WILL NOT BE TESTED. TO TEST DRIVE 0,/<15><12>
12625 : .ASCIZ /RESTART AT 214 AND MOUNT SCRATCH PACK AS DIRECTED./
12626 051317 015 047012 020117 OPR008: .ASCIZ <15><12>/NO DRIVES AVAILABLE FOR TESTING. PROGRAM ABORTED/
12627 051324 051104 053111 051505
12628 051332 040440 040526 046111
12629 051340 041101 042514 043040
12630 051346 051117 052040 051505
12631 051354 044524 043516 020056
12632 051362 051120 043517 040522
12633 051370 020115 041101 051117
12634 051376 042524 000104
12635 051402 005015 044124 020105 OPR009: .ASCIZ <15><12>/THE FOLLOWING DRIVES WILL BE TESTED/<15><12>
12636 051410 047506 046114 053517
12637 051416 047111 020107 051104
12638 051424 053111 051505 053440
12639 051432 046111 020114 042502
12640 051440 052040 051505 042524
12641 051446 006504 000012
```

```
12642 051452 005015 047516 050040 OPR010: .ASCII <15><12>/NO PARITY OPTION FOR MEMORY AREA IN USE/<15><12>
12643 051460 051101 052111 020131
12644 051466 050117 044524 047117
12645 051474 043040 051117 046440
12646 051502 046505 051117 020131
12647 051510 051101 040505 044440
12648 051516 020116 051525 006505
12649 051524 012
12650 051525 055 052440 044516 .ASCIZ /- UNIBUS PARITY ERROR TEST BYPASSED/<15><12>
12651 051532 052502 020123 040520
12652 051540 044522 054524 042440
12653 051546 051122 051117 052040
12654 051554 051505 020124 054502
12655 051562 040520 051523 042105
12656 051570 005015 000
12657 051573 015 046412 046505 OPR011: .ASCII <15><12>/MEMORY SIZE NOT LARGE ENOUGH FOR BUS ADDRESS BITS 16 & 17 TESTS
12658 051600 051117 020131 044523
12659 051606 042532 047040 052117
12660 051614 046040 051101 042507
12661 051622 042440 047516 043525
12662 051630 020110 047506 020122
12663 051636 052502 020123 042101
12664 051644 051104 051505 020123
12665 051652 044502 051524 030440
12666 051660 020066 020046 033461
12667 051666 052040 051505 051524
12668 051674 005015
12669 051676 046101 020114 040504 .ASCIZ /ALL DATA XFER TESTS WITH ADDR >/
12670 051704 040524 054040 042506
12671 051712 020122 042524 052123
12672 051720 020123 044527 044124
12673 051726 040440 042104 020122
12674 051734 000076
12675 051736 031063 000113 OPR012: .ASCIZ /32K/
12676 051742 032066 000113 OPR013: .ASCIZ /64K/
12677 051746 033071 000113 OPR014: .ASCIZ /96K/
12678 051752 041040 050131 051501 OPR015: .ASCIZ / BYPASSED/<15><12>
12679 051760 042523 006504 000012
12680 051766 005015 020012 020040 OPR016: .ASCII <15><12><12>/ *** CAUTION ***/<15><12><12>
12681 051774 020040 020040 025052
12682 052002 020052 040503 052125
12683 052010 047511 020116 025052
12684 052016 006452 005012
12685 052022 044124 051511 050040 .ASCII /THIS PROGRAM SHOULD BE HALTED BY TYPING ^C./<15><12>
12686 052030 047522 051107 046501
12687 052036 051440 047510 046125
12688 052044 020104 042502 044040
12689 052052 046101 042524 020104
12690 052060 054502 052040 050131
12691 052066 047111 020107 041536
12692 052074 006456 012
12693 052077 111 020106 040510 .ASCII /IF HALTED USING THE HALT KEY, THE STATE OF THE DRIVE/<15><12>
12694 052104 052114 042105 052440
12695 052112 044523 043516 052040
12696 052120 042510 044040 046101
12697 052126 020124 042513 026131
```

12698	052134	052040	042510	051440
12699	052142	040524	042524	047440
12700	052150	020106	044124	020105
12701	052156	051104	053111	006505
12702	052164	012		
12703	052165	117	020122	040503
12704	052172	052122	044522	043504
12705	052200	020105	040503	047116
12706	052206	052117	041040	020105
12707	052214	051120	042105	041511
12708	052222	042524	027104	005015
12709	052230	012		
12710	052231	101	046114	042040
12711	052236	044522	042526	020123
12712	052244	047524	041040	020105
12713	052252	042524	052123	042105
12714	052260	046440	051525	020124
12715	052266	042502	047440	026516
12716	052274	044514	042516	006454
12717	052302	012		
12718	052303	122	040505	054504
12719	052310	020054	047101	020104
12720	052316	051127	052111	020105
12721	052324	047514	045503	051040
12722	052332	051505	052105	006456
12723	052340	012		
12724	052341	101	054516	042040
12725	052346	044522	042526	047040
12726	052354	052117	052040	020117
12727	052362	042502	052040	051505
12728	052370	042524	020104	052515
12729	052376	052123	041040	020105
12730	052404	043117	026506	044514
12731	052412	042516	006456	005012
12732	052420	047516	042524	020072
12733	052426	047062	020104	047101
12734	052434	020104	052523	051502
12735	052442	050505	042525	052116
12736	052450	050040	051501	020123
12737	052456	052522	020116	044524
12738	052464	042515	044440	006523
12739	052472	012		
12740	052473	040	020040	020040
12741	052500	040440	050120	047522
12742	052506	020130	020062	044515
12743	052514	020116	030063	051440
12744	052522	041505	043040	051117
12745	052530	042440	041501	020110
12746	052536	051104	053111	027105
12747	052544	005015	000	
12748	052547	015	043012	051111
12749	052554	052123	031040	033065
12750	052562	051440	041505	047524
12751	052570	051522	047040	052117
12752	052576	041040	042523	042440
12753	052604	051122	051117	043040

.ASCII /OR CARTRIDGE CANNOT BE PREDICTED./<15><12><12>

.ASCII /ALL DRIVES TO BE TESTED MUST BE ON-LINE./<15><12>

.ASCII /READY, AND WRITE LOCK RESET./<15><12>

.ASCII /ANY DRIVE NOT TO BE TESTED MUST BE OFF-LINE./<15><12><12>

.ASCII /NOTE: 2ND AND SUBSEQUENT PASS RUN TIME IS/<15><12>

.ASCIZ / APPROX 2 MIN 30 SEC FOR EACH DRIVE./<15><12>

OPR017: .ASCII <15><12>/FIRST 256 SECTORS NOT BSE ERROR FREE./

12754 052612 042522 027105
12755 052616 040515 044530
12756 052624 020115 040504
12757 052632 052040 040522
12758 052640 042506 020122
12759 052646 052123 041040
12760 052654 051501 042523
12761 052662 000012
12762 052664 020040 020040
12763 052672 047412 046116
12764 052700 020061 051104
12765 052706 027105 047440
12766 052714 046122 050101
12767 052722 020104 050117
12768 052730 052101 047511
12769 052736 041040 050131
12770 052744 042523 006504
12771 052752 005015 041523
12772 052760 035105 041440
12773 052766 024040 051124
12774 052774 026051 042440
12775 053002 034055 020073
12776 053010 026062 042440
12777 053016 031055 024040
12778 053024 041440 052517
12779 053032 026105 027040
12780 053040 041457 024515
12781 053044 005015 042101
12782 053052 052123 051040
12783 053060 043040 051117
12784 053066 047117 052123
12785 053074 020124 042514
12786 053102 020114 047117
12787 053110 031110 005015
12788 053115 015 050012
12789 053122 051107 046501
12790 053130 046101 020124
12791 053136 042116 047111
12792 053144 020055 040503
12793 053152 044522 043504
12794 053160 047506 046522
12795 053166 041040 044505
12796 053174 041440 051117
12797 053202 052103 042105
12798 053210 000
12799 053211 015 025012
12800 053216 025052 020040
12801 053224 043517 040522
12802 053232 040510 052114
12803 053240 020040 025052
12804 053246 006452 000012
12805 053252 040503 052122
12806 053260 043504 020105
12807 053266 046522 052101
12808 053274 051117 042522
12809 053302 047511 020116

.ASCIIZ /MAXIMUM DATA TRANSFER TEST BYPASSED/<15><12>
OPR018: .ASCIIZ / /<15><12>/ONLY 1 DRIVE. OVERLAPPED OPERATIONS BYPASSED/<15><12>
OPR019: .ASCII <15><12>@SCOPE: CH1 (TRIG), E53-8; CH2, E49-2 (AC COUPLE, .2V/CM)@
.ASCIIZ <15><12>/ADJUST R72 FOR CONSTANT LEVEL ON CH2/<15><12>
OPR020: .ASCIIZ <15><12>/PROGRAM HALT PENDING - CARTRIDGE FORMAT BEING CORRECTED/<15><12>
OPR021: .ASCIIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
OPR022: .ASCIIZ /CARTRIDGE FORMAT CORRECTION FAILED/<15><12>

12810	053310	046111	042105	005015
12811	053316	000		
12812	053317	015	050012	047522
12813	053324	051107	046501	040440
12814	053332	047502	052122	047111
12815	053340	020107	042502	040503
12816	053346	051525	020105	051105
12817	053354	047522	020122	044124
12818	053362	042522	044123	046117
12819	053370	020104	054105	042503
12820	053376	042105	042105	005015
12821	053404	000		
12822				
12823				
12824	053405	106	052101	046101
12825	053412	047055	047117	042440
12826	053420	044530	052123	047101
12827	053426	020124	042515	047515
12828	053434	054522	040440	020124
12829	053442	045522	030466	020061
12830	053450	040502	042523	040440
12831	053456	042104	042522	051523
12832	053464	000		
12833	053465	106	052101	046101
12834	053472	053455	044522	042524
12835	053500	051040	040505	054504
12836	053506	040440	042116	044440
12837	053514	020105	044504	020104
12838	053522	047516	020124	040503
12839	053530	051525	020105	047111
12840	053536	042524	051122	050125
12841	053544	000124		
12842	053546	040506	040524	026514
12843	053554	040520	044522	054524
12844	053562	042440	051122	051117
12845	053570	052040	040522	027120
12846	053576	050040	020103	052101
12847	053604	042440	051122	051117
12848	053612	036440	000040	
12849	053616	054105	042520	052103
12850	053624	042105	044440	052116
12851	053632	051105	052522	052120
12852	053640	042040	042111	047040
12853	053646	052117	047440	041503
12854	053654	051125	047440	020122
12855	053662	040527	020123	040514
12856	053670	042524	020056	047503
12857	053676	046515	047101	020104
12858	053704	040527	035123	000
12859	053711	123	041125	054523
12860	053716	052123	046505	041440
12861	053724	042514	051101	042040
12862	053732	042111	047040	052117
12863	053740	051040	051505	052105
12864	053746	042440	051122	051117
12865	053754	000		

ABORT: .ASCIZ <15><12>/PROGRAM ABORTING BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>

.SBTTL ERROR MESSAGES

EM1: .ASCIZ /FATAL-NON EXISTANT MEMORY AT RK611 BASE ADDRESS/

EM2: .ASCIZ /FATAL-WRITE READY AND IE DID NOT CAUSE INTERRUPT/

EM3: .ASCIZ /FATAL-PARITY ERROR TRAP. PC AT ERROR = /

EM4: .ASCIZ /EXPECTED INTERRUPT DID NOT OCCUR OR WAS LATE. COMMAND WAS:/

EM5: .ASCIZ /SUBSYSTEM CLEAR DID NOT RESET ERROR/

12866	053755	123	041125	054523	EM6:	.ASCIZ /SUBSYSTEM CLEAR DID NOT RESET DEVICE INTERRUPT/
12867	053762	052123	046505	041440		
12868	053770	042514	051101	042040		
12869	053776	042111	047040	052117		
12870	054004	051040	051505	052105		
12871	054012	042040	053105	041511		
12872	054020	020105	047111	042524		
12873	054026	051122	050125	000124		
12874	054034	052523	051502	051531	EM7:	.ASCIZ /SUBSYSTEM CLEAR DID NOT SET READY/
12875	054042	042524	020115	046103		
12876	054050	040505	020122	044504		
12877	054056	020104	047516	020124		
12878	054064	042523	020124	042522		
12879	054072	042101	000131			
12880	054076	047527	042122	041440	EM10:	.ASCIZ /WORD COUNT INCORRECT/
12881	054104	052517	052116	044440		
12882	054112	041516	051117	042522		
12883	054120	052103	000			
12884	054123	102	051525	040440	EM11:	.ASCIZ /BUS ADDRESS INCORRECT/
12885	054130	042104	042522	051523		
12886	054136	044440	041516	051117		
12887	054144	042522	052103	000		
12888	054151	125	050120	051105	EM11A:	.ASCIZ /UPPER BUS ADDRESS BITS INCORRECT (BA16, 17)/
12889	054156	041040	051525	040440		
12890	054164	042104	042522	051523		
12891	054172	041040	052111	020123		
12892	054200	047111	047503	051122		
12893	054206	041505	020124	041050		
12894	054214	030501	026066	030440		
12895	054222	024467	000			
12896	054225	103	046131	047111	EM12:	.ASCIZ /CYLINDER ADDRESS INCORRECT/
12897	054232	042504	020122	042101		
12898	054240	051104	051505	020123		
12899	054246	047111	047503	051122		
12900	054254	041505	000124			
12901	054260	051124	041501	020113	EM13:	.ASCIZ /TRACK ADDRESS INCORRECT/
12902	054266	042101	051104	051505		
12903	054274	020123	047111	047503		
12904	054302	051122	041505	000124		
12905	054310	042523	052103	051117	EM14:	.ASCIZ /SECTOR ADDRESS INCORRECT./
12906	054316	040440	042104	042522		
12907	054324	051523	044440	041516		
12908	054332	051117	042522	052103		
12909	054340	000056				
12910	054342	047516	026516	054105	EM15:	.ASCIZ /NON-EXISTANT DRIVE/
12911	054350	051511	040524	052116		
12912	054356	042040	044522	042526		
12913	054364	000				
12914	054365	103	047117	051124	EM16:	.ASCIZ /CONTROLLER TIMEOUT/
12915	054372	046117	042514	020122		
12916	054400	044524	042515	052517		
12917	054406	000124				
12918	054410	047125	052111	043040	EM17:	.ASCIZ /UNIT FIELD ERROR/
12919	054416	042511	042114	042440		
12920	054424	051122	051117	000		
12921	054431	115	046125	050111	EM18:	.ASCIZ /MULIPLE DRIVE SELECT/

12922	054436	042514	042040	044522		
12923	054444	042526	051440	046105		
12924	054452	041505	000124			
12925	054456	051120	043517	040522	EM19:	.ASCIZ /PROGRAMMING ERROR/
12926	054464	046515	047111	020107		
12927	054472	051105	047522	000122		
12928	054500	047516	026516	054105	EM20:	.ASCIZ /NON-EXISTANT MEMORY/
12929	054506	051511	040524	052116		
12930	054514	046440	046505	051117		
12931	054522	000131				
12932	054524	047125	041111	051525	EM21:	.ASCIZ /UNIBUS PARITY ERROR/
12933	054532	050040	051101	052111		
12934	054540	020131	051105	047522		
12935	054546	000122				
12936	054550	046111	042514	040507	EM22:	.ASCIZ /ILLEGAL FUNCTION/
12937	054556	020114	052506	041516		
12938	054564	044524	047117	000		
12939	054571	105	041503	044040	EM23:	.ASCIZ /ECC HARD/
12940	054576	051101	000104			
12941	054602	040504	040524	041440	EM24:	.ASCIZ /DATA CHECK/
12942	054610	042510	045503	000		
12943	054615	127	044522	042524	EM25:	.ASCIZ /WRITE CHECK/
12944	054622	041440	042510	045503		
12945	054630	000				
12946	054631	104	052101	020101	EM26:	.ASCIZ /DATA LATE/
12947	054636	040514	042524	000		
12948	054643	117	042520	040522	EM27:	.ASCIZ /OPERATION INCOMPLETE/
12949	054650	044524	047117	044440		
12950	054656	041516	046517	046120		
12951	054664	052105	000105			
12952	054670	042510	042101	051105	EM28:	.ASCIZ /HEADER VRC/
12953	054676	053040	041522	000		
12954	054703	102	042101	051440	EM29:	.ASCIZ /BAD SECTOR ERROR/
12955	054710	041505	047524	020122		
12956	054716	051105	047522	000122		
12957	054724	047503	052116	047522	EM30:	.ASCIZ /CONTROLLER DETECTED DRIVE BUS PARITY ERROR/
12958	054732	046114	051105	042040		
12959	054740	052105	041505	042524		
12960	054746	020104	051104	053111		
12961	054754	020105	052502	020123		
12962	054762	040520	044522	054524		
12963	054770	042440	051122	051117		
12964	054776	000				
12965	054777	123	042505	020113	EM31:	.ASCIZ /SEEK INCOMPLETE/
12966	055004	047111	047503	050115		
12967	055012	042514	042524	000		
12968	055017	116	047117	042455	EM32:	.ASCIZ /NON-EXECUTABLE DRIVE FUNCTION/
12969	055024	042530	052503	040524		
12970	055032	046102	020105	051104		
12971	055040	053111	020105	052506		
12972	055046	041516	044524	047117		
12973	055054	000				
12974	055055	104	044522	042526	EM33:	.ASCIZ /DRIVE DETECTED DRIVE BUS PARITY ERROR/
12975	055062	042040	052105	041505		
12976	055070	042524	020104	051104		
12977	055076	053111	020105	052502		

12978	055104	020123	040520	044522					
12979	055112	054524	042440	051122					
12980	055120	051117	000						
12981	055123	106	051117	040515	EM34:	.ASCIZ	/FORMAT ERROR/		
12982	055130	020124	051105	047522					
12983	055136	000122							
12984	055140	051104	053111	020105	EM35:	.ASCIZ	/DRIVE TYPE ERROR/		
12985	055146	054524	042520	042440					
12986	055154	051122	051117	000					
12987	055161	101	020103	047514	EM36:	.ASCIZ	/AC LOW/		
12988	055166	000127							
12989	055170	050123	047111	046104	EM37:	.ASCIZ	/SPINDLE SPEED LOSS/		
12990	055176	020105	050123	042505					
12991	055204	020104	047514	051523					
12992	055212	000							
12993	055213	104	044522	042526	EM38:	.ASCIZ	/DRIVE OFF TRACK/		
12994	055220	047440	043106	052040					
12995	055226	040522	045503	000					
12996	055233	103	046131	047111	EM39:	.ASCIZ	/CYLINDER OVERFLOW/		
12997	055240	042504	020122	053117					
12998	055246	051105	046106	053517					
12999	055254	000							
13000	055255	111	046114	043505	EM40:	.ASCIZ	/ILLEGAL DISK PACK ADDRESS/		
13001	055262	046101	042040	051511					
13002	055270	020113	040520	045503					
13003	055276	040440	042104	042522					
13004	055304	051523	000						
13005	055307	127	044522	042524	EM41:	.ASCIZ	/WRITE LOCK ERROR/		
13006	055314	046040	041517	020113					
13007	055322	051105	047522	000122					
13008	055330	051104	053111	020105	EM42:	.ASCIZ	/DRIVE TIMING ERROR/		
13009	055336	044524	044515	043516					
13010	055344	042440	051122	051117					
13011	055352	000							
13012	055353	116	020117	042503	EM43:	.ASCIZ	/NO CERR WITH SOME OTHER ERROR SET/		
13013	055360	051122	053440	052111					
13014	055366	020110	047523	042515					
13015	055374	047440	044124	051105					
13016	055402	042440	051122	051117					
13017	055410	051440	052105	000					
13018	055415	104	044522	042526	EM44:	.ASCIZ	/DRIVE UNSAFE/		
13019	055422	052440	051516	043101					
13020	055430	000105							
13021	055432	042503	051122	051440	EM45:	.ASCIZ	/CERR SET BUT NO OTHER ERROR SET/		
13022	055440	052105	041040	052125					
13023	055446	047040	020117	052117					
13024	055454	042510	020122	051105					
13025	055462	047522	020122	042523					
13026	055470	000124							
13027									
13028	055472	053126	042040	042111	EM46:	.ASCIZ	/VV DID NOT SET WITH PACK ACK/		
13029	055500	047040	052117	051440					
13030	055506	052105	053440	052111					
13031	055514	020110	040520	045503					
13032	055522	040440	045503	000					
13033	055527	123	040524	052524	EM47:	.ASCIZ	/STATUS VALID SET ON SELECT TO NON-EXISTANT DRIVE/		

13034	055534	020123	040526	044514	
13035	055542	020104	042523	020124	
13036	055550	047117	051440	046105	
13037	055556	041505	020124	047524	
13038	055564	047040	047117	042455	
13039	055572	044530	052123	047101	
13040	055600	020124	051104	053111	
13041	055606	000105			
13042	055610	052123	052101	051525	EM48: .ASCIZ /STATUS VALID RESET ON SELECT TO EXISTANT DRIVE/
13043	055616	053040	046101	042111	
13044	055624	051040	051505	052105	
13045	055632	047440	020116	042523	
13046	055640	042514	052103	052040	
13047	055646	020117	054105	051511	
13048	055654	040524	052116	042040	
13049	055662	044522	042526	000	
13050	055667	123	040524	052524	EM49: .ASCIZ /STATUS VALID NOT RESET ON DRIVE RELEASE/
13051	055674	020123	040526	044514	
13052	055702	020104	047516	020124	
13053	055710	042522	042523	020124	
13054	055716	047117	042040	044522	
13055	055724	042526	051040	046105	
13056	055732	040505	042523	000	
13057	055737	105	050130	041505	EM50: .ASCIZ /EXPECTED 2ND INTERRUPT DID NOT OCCUR OR WAS LATE. COMMAND WAS:/
13058	055744	042524	020104	047062	
13059	055752	020104	047111	042524	
13060	055760	051122	050125	020124	
13061	055766	044504	020104	047516	
13062	055774	020124	041517	052503	
13063	056002	020122	051117	053440	
13064	056010	051501	046040	052101	
13065	056016	027105	041440	046517	
13066	056024	040515	042116	053440	
13067	056032	051501	000072		
13068	056036	040503	047116	052117	EM51: .ASCII /CANNOT READ BAD SECTOR FILE/<15><12>
13069	056044	051040	040505	020104	
13070	056052	040502	020104	042523	
13071	056060	052103	051117	043040	
13072	056066	046111	006505	012	
13073	056073	124	051505	044524	.ASCIZ /TESTING ABORTED ON THIS DRIVE/
13074	056100	043516	040440	047502	
13075	056106	052122	042105	047440	
13076	056114	020116	044124	051511	
13077	056122	042040	044522	042526	
13078	056130	000			
13079	056131	101	044514	047107	EM52: .ASCII /ALIGNMENT PACK ON THIS DRIVE./<15><12>
13080	056136	042515	052116	050040	
13081	056144	041501	020113	047117	
13082	056152	052040	044510	020123	
13083	056160	051104	053111	027105	
13084	056166	005015			
13085	056170	042524	052123	047111	.ASCIZ /TESTING ABORTED ON THIS DRIVE./
13086	056176	020107	041101	051117	
13087	056204	042524	020104	047117	
13088	056212	052040	044510	020123	
13089	056220	051104	053111	027105	

13090	056226	000				
13091	056227	124	020117	040515	EM53:	.ASCIZ /TO MANY BAD SECTORS. TESTING ABORTED ON THIS DRIVE/
13092	056234	054516	041040	042101		
13093	056242	051440	041505	047524		
13094	056250	051522	020056	042524		
13095	056256	052123	047111	020107		
13096	056264	041101	051117	042524		
13097	056272	020104	047117	052040		
13098	056300	044510	020123	051104		
13099	056306	053111	000105			
13100	056312	042510	042101	051105	EM54:	.ASCIZ /HEADER WORD MISCOMPARE/
13101	056320	053440	051117	020104		
13102	056326	044515	041523	046517		
13103	056334	040520	042522	000		
13104	056341	104	052101	020101	EM55:	.ASCIZ /DATA WORD MISCOMPARE/
13105	056346	047527	042122	046440		
13106	056354	051511	047503	050115		
13107	056362	051101	000105			
13108						
13109	056366	040503	047116	052117	EM56:	.ASCIZ /CANNOT FIND HEADER 0 IN READ AND SORT HEADERS/
13110	056374	043040	047111	020104		
13111	056402	042510	042101	051105		
13112	056410	030040	044440	020116		
13113	056416	042522	042101	040440		
13114	056424	042116	051440	051117		
13115	056432	020124	042510	042101		
13116	056440	051105	000123			
13117	056444	052123	052101	051525	EM5VAL:	.ASCIZ /STATUS VALID/
13118	056452	053040	046101	042111		
13119	056460	000				
13120	056461	123	046105	041505	EMNZPR:	.ASCIZ /SELECT WITH NON-ZERO STATUS PAIR/
13121	056466	020124	044527	044124		
13122	056474	047040	047117	055055		
13123	056502	051105	020117	052123		
13124	056510	052101	051525	050040		
13125	056516	044501	000122			
13126	056522	051127	052111	047111	EMWCS2:	.ASCIZ /WRITING COMMAND STATUS REGISTER 2/
13127	056530	020107	047503	046515		
13128	056536	047101	020104	052123		
13129	056544	052101	051525	051040		
13130	056552	043505	051511	042524		
13131	056560	020122	000062			
13132		055140				
13133	056564	051104	053111	020105	EMDTPE= EM35	:DRIVE TYPE ERROR
13134	056572	047111	042524	051122	EMDI:	.ASCIZ /DRIVE INTERRUPT/
13135	056600	050125	000124			
13136		055055				
13137	056604	051104	053111	020105	EMDPAR= EM33	:DRIVE DETECTED DRIVE BUS PARITY ERROR
13138	056612	052123	052101	051525	EMDSC:	.ASCIZ /DRIVE STATUS CHANGE/
13139	056620	041440	040510	043516		
13140	056626	000105				
13141	056630	051104	053111	020105	EMDA:	.ASCIZ /DRIVE ATTENTION/
13142	056636	052101	042524	052116		
13143	056644	047511	000116			
13144	056650	047503	052116	047522	EMCCLR:	.ASCIZ /CONTROLLER CLEAR/
13145	056656	046114	051105	041440		

13146	056664	042514	051101	000	
13147		050250			EMSELD= OPER00 ;DRIVE SELECT
13148		050276			EMDCLR= OPER04 ;DRIVE CLEAR
13149		050337			EMRCAL= OPER12 ;RECALIBRATE
13150	056671	123	041505	047117	EM2INT: .ASCIZ /SECOND INTERRUPT/
13151	056676	020104	047111	042524	
13152	056704	051122	050125	000124	
13153	056712	042523	045505	052040	EMSKSF: .ASCIZ /SEEK TO SELF/
13154	056720	020117	042523	043114	
13155	056726	000			
13156		050362			EMSK= OPER16 ;SEEK
13157	056727	125	042516	050130	EMUXIT: .ASCIZ /UNEXPECTED INTERRUPT/
13158	056734	041505	042524	020104	
13159	056742	047111	042524	051122	
13160	056750	050125	000124		
13161	056754	047125	041111	051525	EMUR: .ASCIZ /UNIBUS RESET/
13162	056762	051040	051505	052105	
13163	056770	000			
13164	056771	122	051505	052105	EMRSET: .ASCIZ /RESET/
13165	056776	000			
13166		054631			EMDLT= EM26 ;DATA LATE
13167	056777	122	040505	044504	EMRDB: .ASCIZ /READING DATA BUFFER/
13168	057004	043516	042040	052101	
13169	057012	020101	052502	043106	
13170	057020	051105	000		
13171	057023	103	047117	051124	EMCERR: .ASCIZ /CONTROLLER ERROR/
13172	057030	046117	042514	020122	
13173	057036	051105	047522	000122	
13174	057044	052523	051502	051531	EMSCLR: .ASCIZ /SUBSYSTEM CLEAR/
13175	057052	042524	020115	046103	
13176	057060	040505	000122		
13177	057064	052515	052114	050111	EMMI: .ASCIZ /MULTIPLE INTERRUPTS/
13178	057072	042514	044440	052116	
13179	057100	051105	052522	052120	
13180	057106	000123			
13181	057110	040502	020104	042523	DRVABT: .ASCII /BAD SECTORS ON PACK IN AREAS USED BY TEST (CYL 312(8))/<15><12>
13182	057116	052103	051117	020123	
13183	057124	047117	050040	041501	
13184	057132	020113	047111	040440	
13185	057140	042522	051501	052440	
13186	057146	042523	020104	054502	
13187	057154	052040	051505	020124	
13188	057162	041450	046131	031440	
13189	057170	031061	034050	024451	
13190	057176	005015			
13191	057200	042524	052123	047111	.ASCIZ /TESTING ABORTED ON THIS DRIVE/
13192	057206	020107	041101	051117	
13193	057214	042524	020104	047117	
13194	057222	052040	044510	020123	
13195	057230	051104	053111	000105	
13196					
13197					
13198	057236	051524	020124	052516	.SBTTL DATA HEADERS FOR ERROR REPORTS
13199	057244	020115	051105	020122	DH001: .ASCIZ /TST NUM ERR PC DRIVE/
13200	057252	041520	020040	051104	
13201	057260	053111	000105		

13202 057264 051524 020124 052516
13203 057272 020115 051105 020122
13204 057300 041520 020040 051104
13205 057306 053111 000105
13206 057312 045522 051503 020061
13207 057320 020040 045522 051503
13208 057326 020062 020040 045522
13209 057334 051504 020040 020040
13210 057342 045522 051105 020040
13211 057350 020040 045522 051501
13212 057356 043117 020040 045522
13213 057364 041504 046131 020040
13214 057372 045522 040504 000
13215 057377 122 041113 020101
13216 057404 020040 051040 053513
13217 057412 000103
13218 057414 030101 020060 020040
13219 057422 020040 030102 020060
13220 057430 020040 020040 030101
13221 057436 020061 020040 020040
13222 057444 030102 020061 020040
13223 057452 020040 030101 020062
13224 057460 020040 020040 030102
13225 057466 020062 020040 020040
13226 057474 030101 020063 020040
13227 057502 020040 030102 000063
13228 057510 045522 051503 020061
13229 057516 020040 045522 051503
13230 057524 020062 020040 045522
13231 057532 051504 020040 020040
13232 057540 045522 051105 020040
13233 057546 020040 045522 051501
13234 057554 043117 020040 045522
13235 057562 051115 000061
13236 057566 044124 020105 041101
13237 057574 053117 020105 051101
13238 057602 020105 054105 042520
13239 057610 052103 042105 042440
13240 057616 051122 051117 020123
13241 057624 044124 052101 042040
13242 057632 042111 047040 052117
13243 057640 051440 052105 044440
13244 057646 020116 050117 051105
13245 057654 052101 047511 035116
13246 057662 000
13247 057663 124 042510 040440
13248 057670 047502 042526 040440
13249 057676 042522 052440 042516
13250 057704 050130 041505 042524
13251 057712 020104 051105 047522
13252 057720 051522 051440 052105
13253 057726 044440 020116 050117
13254 057734 051105 052101 047511
13255 057742 035116 000
13256 057745 124 042510 040440
13257 057752 047502 042526 040440

DH002A: .ASCIZ /TST NUM ERR PC DRIVE/

DH002B: .ASCIZ /RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA/

DH002C: .ASCIZ /RKBA RKWC/

DH002D: .ASCIZ /A00 B00 A01 B01 A02 B02 A03 B03/

DH003B: .ASCIZ /RKCS1 RKCS2 RKDS RKER RKASOF RKMR1/

DH005: .ASCIZ /THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:/

DH006: .ASCIZ /THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:/

DH007: .ASCIZ /THE ABOVE ARE ERRORS SET IN OPERATION:/

13258	057760	042522	042440	051122	
13259	057766	051117	020123	042523	
13260	057774	020124	047111	047440	
13261	060002	042520	040522	044524	
13262	060010	047117	000072		
13263	060014	047101	020131	044506	DH005A: .ASCIZ /ANY FIELD WITH ALL ONES MUST BE CONSIDERED INVALID/
13264	060022	046105	020104	044527	
13265	060030	044124	040440	046114	
13266	060036	047440	042516	020123	
13267	060044	052515	052123	041040	
13268	060052	020105	047503	051516	
13269	060060	042111	051105	042105	
13270	060066	044440	053116	046101	
13271	060074	042111	000		
13272	060077	105	051122	051117	DH010: .ASCIZ /ERROR AT COMPLETION OF OPERATION/
13273	060104	040440	020124	047503	
13274	060112	050115	042514	044524	
13275	060120	047117	047440	020106	
13276	060126	050117	051105	052101	
13277	060134	047511	000116		
13278	060140	054105	052120	020104	DH010A: .ASCIZ /EXPTD IS/
13279	060146	020040	051511	000	
13280	060153	122	040505	020104	DH010B: .ASCIZ /READ DATA WITH FORCED CYLINDER OVERFLOW/
13281	060160	040504	040524	053440	
13282	060166	052111	020110	047506	
13283	060174	041522	042105	041440	
13284	060202	046131	047111	042504	
13285	060210	020122	053117	051105	
13286	060216	046106	053517	000	
13287	060223	116	052117	051440	DH011: .ASCIZ /NOT SET AS A RESULT OF/
13288	060230	052105	040440	020123	
13289	060236	020101	042522	052523	
13290	060244	052114	047440	000106	
13291	060252	047516	020124	042522	DH012: .ASCIZ /NOT RESET AS A RESULT OF/
13292	060260	042523	020124	051501	
13293	060266	040440	051040	051505	
13294	060274	046125	020124	043117	
13295	060302	000			
13296	060303	123	052105	040440	DH013: .ASCIZ /SET AS A RESULT OF/
13297	060310	020123	020101	042522	
13298	060316	052523	052114	047440	
13299	060324	000106			
13300	060326	042522	042523	020124	DH014: .ASCIZ /RESET AS A RESULT OF/
13301	060334	051501	040440	051040	
13302	060342	051505	046125	020124	
13303	060350	043117	000		
13304	060353	107	047517	020104	DH015: .ASCIZ /GOOD BAD WORD NUM/
13305	060360	020040	041040	042101	
13306	060366	020040	020040	053440	
13307	060374	051117	020104	052516	
13308	060402	000115			
13309	060404	052523	051502	051531	DH016: .ASCIZ /SUBSYSTEM CLEAR TO RESET SEEK LIMIT ALLOWING HEADS TO RELOAD/
13310	060412	042524	020115	046103	
13311	060420	040505	020122	047524	
13312	060426	051040	051505	052105	
13313	060434	051440	042505	020113	

13314	060442	044514	044515	020124					
13315	060450	046101	047514	044527					
13316	060456	043516	044040	040505					
13317	060464	051504	052040	020117					
13318	060472	042522	047514	042101					
13319	060500	000							
13320	060501	123	046105	041505	DH017:	.ASCIZ	/SELECT AFTER RECAL/		
13321	060506	020124	043101	042524					
13322	060514	020122	042522	040503					
13323	060522	000114							
13324	060524	042522	040503	020114	DH018:	.ASCIZ	/RECAL COMPLETE ATTN AFTER SELECT/		
13325	060532	047503	050115	042514					
13326	060540	042524	040440	052124					
13327	060546	020116	043101	042524					
13328	060554	020122	042523	042514					
13329	060562	052103	000						
13330									
13331					.SBTTL	DATA TABLES FOR ERROR REPORTS			
13332		060566			.EVEN				
13333	060566	001302	001116	001626	DT003:	.WORD	\$TESTN,\$ERRPC,DRVNUM,T.CS1,T.CS2,T.DS,T.ER,T.ASOF,T.MR1		
13334	060574	001540	001550	001552					
13335	060602	001554	001556	001566					
13336	060610	001302	001116	001626	DT001:	.WORD	\$TESTN,\$ERRPC,DRVNUM		
13337	060616				DT002:				
13338	060616				DT004:				
13339	060616				DT005:				
13340	060616				DT006:				
13341	060616				DT007:				
13342	060616	001302	001116	001626	DT010:	.WORD	\$TESTN,\$ERRPC,DRVNUM		
13343	060624	001540	001550	001552	.WORD	T.CS1,T.CS2,T.DS,T.ER,T.ASOF			
13344	060632	001554	001556						
13345	060636	001560	001546		.WORD	T.DCYL,T.DA			
13346	060642	001544	001542		.WORD	T.BA,T.WC			
13347	060646	001202	001204	001206	.WORD	\$REG10,\$REG11,\$REG12,\$REG13,\$REG14,\$REG15,\$REG16,\$REG17			
13348	060654	001210	001212	001214					
13349	060662	001216	001220						
13350	060666	001302	001116	001626	DT015:	.WORD	\$TESTN,\$ERRPC,DRVNUM		
13351	060674	001202	001204	001206	DT015A:	.WORD	\$REG10,\$REG11,\$REG12		
13352					.SBTTL	DATA FORMATS FOR ERROR REPORTS			
13353	060702	000001			DF001:	.WORD	1		
13354	060704	003	000		.BYTE	3,0			
13355									
13356	060706	000004			DF002:	.WORD	4		
13357	060710	000	000		.BYTE	0,0			
13358	060712	057264			.WORD	DH002A			
13359	060714	003	000		.BYTE	3,0			
13360	060716	057312			.WORD	DH002B			
13361	060720	007	000		.BYTE	7,0			
13362	060722	057377			.WORD	DH002C			
13363	060724	002	000		.BYTE	2,0			
13364									
13365									
13366	060726	000002			DF003:	.WORD	2		
13367	060730	003	000		.BYTE	3,0			
13368	060732	057510			.WORD	DH003B			
13369	060734	006	000		.BYTE	6,0			

13370					
13371	060736	000006		DF004:	.WORD 6
13372	060740	000	000		.BYTE 0,0
13373	060742	000000		DF004A:	.WORD 0
13374	060744	000	000		.BYTE 0,0
13375	060746	057264			.WORD DH002A
13376	060750	003	000		.BYTE 3,0
13377	060752	057312			.WORD DH002B
13378	060754	007	000		.BYTE 7,0
13379	060756	057377			.WORD DH002C
13380	060760	002	000		.BYTE 2,0
13381	060762	057414			.WORD DH002D
13382	060764	010	000		.BYTE 10,0
13383					
13384	060766	000007		DF005:	.WORD 7
13385	060770	000	000		.BYTE 0,0
13386	060772	000000		DF005A:	.WORD 0
13387	060774	000	000		.BYTE 0,0
13388	060776	057264			.WORD DH002A
13389	061000	003	000		.BYTE 3,0
13390	061002	057312			.WORD DH002B
13391	061004	007	000		.BYTE 7,0
13392	061006	057377			.WORD DH002C
13393	061010	002	000		.BYTE 2,0
13394	061012	057414			.WORD DH002D
13395	061014	010	000		.BYTE 10,0
13396	061016	060014			.WORD DH005A
13397	061020	000	000		.BYTE 0,0
13398					
13399	061022	000005		DF006:	.WORD 5
13400	061024	000	000		.BYTE 0,0
13401	061026	000000		DF006A:	.WORD 0
13402	061030	000	000		.BYTE 0,0
13403	061032	057264			.WORD DH002A
13404	061034	003	000		.BYTE 3,0
13405	061036	057312			.WORD DH002B
13406	061040	007	000		.BYTE 7,0
13407	061042	057377			.WORD DH002C
13408	061044	002	000		.BYTE 2,0
13409					
13410	061046	000004		DF007:	.WORD 4
13411	061050	000	000		.BYTE 0,0
13412	061052	000000		DF007A:	.WORD 0
13413	061054	000	000		.BYTE 0,0
13414	061056	057264			.WORD DH002A
13415	061060	003	000		.BYTE 3,0
13416	061062	057510			.WORD DH003B
13417	061064	006	000		.BYTE 6,0
13418					
13419	061066	000004		DF010:	.WORD 4
13420	061070	000	000		.BYTE 0,0
13421	061072	000000		DF010A:	.WORD 0
13422	061074	000	000		.BYTE 0,0
13423	061076	057264			.WORD DH002A
13424	061100	003	000		.BYTE 3,0
13425	061102	060140			.WORD DH010A

13426	061104	002	000		.BYTE	2,0		
13427								
13428	061106	000004			DF011:	.WORD	4	
13429	061110	000	000			.BYTE	0,0	
13430	061112	000000			DF011A:	.WORD	0	
13431	061114	000	000			.BYTE	0,0	
13432	061116	057264				.WORD	DH002A	
13433	061120	003	000			.BYTE	3,0	
13434	061122	057510				.WORD	DH003B	
13435	061124	000006	000000			.WORD	6,0	
13436								
13437	061130	000002			DF015:	.WORD	2	
13438	061132	003	000			.BYTE	3,0	
13439	061134	060353				.WORD	DH015	
13440	061136	003	000			.BYTE	3,0	
13441								
13442	061140	000001			DF016:	.WORD	1	
13443	061142	003	000			.BYTE	3,0	
13444								
13445	061144	000052			BS26:	.BLKW	52	
13446	061270	000052			BS24:	.BLKW	52	
13447		070376				.=70376		
13448	070376	000240				NOP		
13449	070400	005014			SPCHLR:	CLR	(R4)	:CLEAR MEM MAINT REG
13450	070402	005237	001662			INC	INTSET	:COUNT THE INTERRUPT
13451	070406	000240				NOP		
13452	070410	000002			SPCPAR:	RTI		:RETURN
13453	070412	000240				NOP		
13454	070414	001000			IBUFF:	.BLKW	1000	
13455	072414	001000			OBUFF:	.BLKW	1000	
13456		000001			.END			

Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value
ABASE =	177440	1963#	2285	2326			
ABORT	053317	9586	12812#				
ABTFAI	044170	9381	11408#				
ACDW1 =	000000	2285	2328				
ACDW2 =	000000	2285					
ACLO =	000010	2058#	10502				
ACLOER=	000100	10504	12396#				
ACPUOP=	000000	2285	2300				
ADDW0 =	000000	2285					
ADDW1 =	000000	2285					
ADDW10=	000000	2285					
ADDW11=	000000	2285					
ADDW12=	000000	2285					
ADDW13=	000000	2285					
ADDW14=	000000	2285					
ADDW15=	000000	2285					
ADDW2 =	000000	2285					
ADDW3 =	000000	2285					
ADDW4 =	000000	2285					
ADDW5 =	000000	2285					
ADDW6 =	000000	2285					
ADDW7 =	000000	2285					
ADDW8 =	000000	2285					
ADDW9 =	000000	2285					
ADEVCT=	000000	2285	2291				
ADEVN =	000000	2285	2327				
ADJCLK	043614	3264	11301#				
AENV =	000000	2285	2296				
AENVN =	000000	2285	2297				
AFATAL=	000000	2285	2288				
AMADR1=	000000	2285	2313				
AMADR2=	000000	2285	2317				
AMADR3=	000000	2285	2320				
AMADR4=	000000	2285	2323				
AMAMS1=	000000	2285	2307				
AMAMS2=	000000	2285	2315				
AMAMS3=	000000	2285	2318				
AMAMS4=	000000	2285	2321				
AMSGAD=	000000	2285	2293				
AMSGLG=	000000	2285	2294				
AMSGTY=	000000	2285	2287				
AMTYP1=	000000	2285	2308				
AMTYP2=	000000	2285	2316				
AMTYP3=	000000	2285	2319				
AMTYP4=	000000	2285	2322				
APASS =	000000	2285	2290				
APRIOR=	000240	1962#	2285				
APTCSU=	000040	9439#	11441				
APTENV=	000001	9138	9395	9437#	9494	11434	
APTSIZ=	000200	2799	9436#				
APTSPO=	000100	9397	9438#	11436			
ASWREG=	000000	2285	2298				
ATESTN=	000000	2285	2289				
AUNIT =	000000	2285	2292				
AUSWR =	000000	2285	2299				
AVECT1=	000210	1961#	2285	2324			

IDAE = 002000	2046#																		
IDAERR= 002000	3938	3946	4007	12400#															
IE = 000100	2004#	2970	3868	7817	7920	8118	8490												
ILF = 000001	2036#	10496	10577																
ILFERR= 000200	10579	12424#																	
INTHLR 034442	2917	3041	5731	9609#															
INTR = 000300	1999#																		
INTSET 001662	2693#	2944*	2949	2969*	2974	3741*	3823	3859*	3914*	3977*	5652	7428*	7465*						
	7623*	7684*	7764*	7816*	7853*	7922*	8017	8076*	8119*	8223*	8259*	8356*	8483*						
	8632*	8701*	8789*	8809*	8942*	8964*	9600*	9610*	9716*	9724	9729*	9762	9785						
	9853	9908*	10068*	10073	11353*	13450*													
	1919#	2768*	2769*																
IOTVEC= 000020	2023#																		
IR = 000100	9809#	12118																	
IWAT1M 035370	9815#	12115																	
IWAT1S 035410	9827#	12111																	
IWAT11 035450	9824#	12112																	
IWAT12 035440	9821#	12113																	
IWAT14 035430	9818#	12114																	
IWAT15 035420	9845#	12105																	
IWAT16 035530	9812#	12116																	
IWAT2S 035400	9842#	12106																	
IWAT32 035520	9839#	12107																	
IWAT48 035510	9836#	12108																	
IWAT64 035500	9806#	12117																	
IWAT8S 035360	9833#	12109																	
IWAT80 035470	9830#	12110																	
IWAT96 035460	1952#	9151	9183																
KIPAR0= 172340	1953#																		
KIPAR1= 172342	1954#																		
KIPAR2= 172344	1955#																		
KIPAR3= 172346	1956#																		
KIPAR4= 172350	1957#																		
KIPAR5= 172352	1958#	11141*																	
KIPAR6= 172354	1959#	9199																	
KIPAR7= 172356	1941#																		
KIPDR0= 172300	1942#																		
KIPDR1= 172302	1943#																		
KIPDR2= 172304	1944#																		
KIPDR3= 172306	1945#																		
KIPDR4= 172310	1946#																		
KIPDR5= 172312	1947#																		
KIPDR6= 172314	1948#																		
KIPDR7= 172316	2721#	5660*	5676*	9630*	9720*	10053*	10100*	11310*											
KWLADD 001710	2722#	9717																	
KWLVEC 001712	9629#	9718																	
LCKHLR 034506	2706#	9727	9758	9855	10051	10098	11308												
LCLKPR= 100000	2712#	3863*	3865	9629*	9757*	9764	9766*	9791*	9858	10069*	10076								
LCLKTK 001674	3304	9934#																	
LDCON 036034	1825#	11501	11507	12615	12618														
LF = 000012	9908#	12101																	
LOADRK 035734	4067	4225	4253	4350	4378	4426	4459	4505	4525	4557	4578	4611	4639						
LRLOAD 035674	4672	4698	4733	4762	4781	4827	4847	4914	4938	4989	5006	5068	5086						
	5137	5154	5214	5233	5288	5305	5350	5369	5425	5443	5494	5547	5619						
	5753	5776	5791	5837	5860	5879	5923	5951	5965	6008	6032	6047	6093						
	6117	6132	6191	6248	6306	6362	6418	6464	6492	6616	6631	6689	6704						

S.HDFL= 000200	2131#														
S.HDHM= 000040	2115#	7914	8477												
S.ICYL= 000040	2101#														
S.ILF = 000400	2104#														
S.LIMD= 020000	2137#														
S.LOAD= 010000	2122#														
S.MHD = 000400	2132#														
S.NMOV= 010000	2136#														
S.OFF = 002000	2093#														
S.PAR = 001000	2105#														
S.PIP = 020000	2096#														
S.PLO = 004000	2135#														
S.REV = 004000	2121#														
S.RTZ = 020000	2123#														
S.SECT= 000020	2128#														
S.SKI = 002000	2106#														
S.SPIN= 010000	2095#														
S.SPLS= 010000	2108#														
S.SPOK= 001000	2119#														
S.TYPE= 000400	2091#														
S.UNLD= 040000	2124#														
S.UNS = 040000	2110#														
S.VV = 000100	2089#														
S.WCLK= 000040	2129#														
S.WGAT= 000100	2130#														
S.WLE = 004000	2107#														
S.WRL = 004000	2094#														
S.XDOK= 000020	2114#														
S.XERR= 001000	2133#														
TBITVE= 000014	1916#														
TCHKOP= 104421	3084	3150	3220	3234	3287	3296	3326	3342	3377	3509	3572	3599	3642		
	3690	3778	3829	4078	4244	4269	4295	4373	4406	4452	4486	4522	4575		
	4636	4695	4709	4748	4778	4793	4844	4867	4930	4948	5004	5017	5080		
	5097	5152	5166	5230	5245	5303	5317	5366	5386	5440	5457	5769	5788		
	5803	5853	5876	5891	5939	5963	6024	6044	6059	6110	6129	6144	6203		
	6260	6318	6374	6430	6476	6545	6565	6628	6643	6701	6716	6777	6841		
	6931	6961	7004	7019	7061	7076	7197	7212	7326	7343	7390	7421	7447		
	7466	7492	7517	7607	7625	7631	7646	7671	7686	7691	7706	7729	7761		
	7767	7782	7857	7911	8000	8044	8135	8168	8194	8240	8256	8264	8279		
	8353	8360	8375	8390	8433	8474	8509	8527	8578	8649	8718	8786	8800		
	8952	8972	9000	11350	11357	11385	12103#								
TCHKWE= 104422	3203	3421	3465	3530	3659	3709	3937	3945	4006	4393	4474	4537	4590		
	4651	5505	5558	5694	6504	6800	6876	7096	7234	7809	7887	7961	8036		
	8109	8316	8450	12104#											
TGETRK= 104420	3070	3120	3389	3606	3742	3757	3809	3850	3878	4273	8661	8728	8818		
	9860	10080	10126	10226	10425	12102#									
TIMCNT 001654	2690#	9760*	9789*	9792*	9851*	10067*									
TKVEC = 000060	1923#	11673*	11674*												
TLOADR= 104417	3065	3115	3197	3229	3283	3292	3323	3338	3373	3417	3461	3506	3526		
	3561	3568	3595	3638	3648	3686	3696	3738	3774	3804	3846	3909	3928		
	3972	4002	4075	4240	4265	4369	4386	4449	4467	4518	4533	4571	4586		
	4632	4647	4691	4705	4744	4774	4789	4840	4863	4926	4945	5000	5014		
	5076	5093	5148	5162	5226	5241	5299	5313	5362	5382	5436	5454	5502		
	5555	5656	5765	5784	5799	5849	5872	5887	5935	5959	5973	6020	6040		
	6055	6105	6125	6140	6199	6256	6314	6370	6426	6472	6500	6541	6624		
	6639	6697	6712	6773	6796	6837	6872	6927	6957	7000	7015	7057	7072		

MSG	2927#	2929	2958#	2960	2987#	2990	3268#	3270	3305#	3307	3357#	3359	3401#	3403	3434#
	3436	3620#	3622	3667#	3669	3717#	3719	3788#	3790	3831#	3833	3889#	3891	3955#	3957
	4014#	4016	4208#	4210	4328#	4330	4410#	4412	4490#	4492	4543#	4545	4595#	4597	4656#
	4658	4715#	4717	4812#	4814	4886#	4888	4967#	4969	5035#	5037	5116#	5118	5186#	5188
	5266#	5268	5337#	5339	5410#	5412	5482#	5484	5535#	5537	5589#	5591	5740#	5742	5823#
	5825	5910#	5912	5994#	5996	6079#	6081	6164#	6166	6223#	6225	6280#	6282	6337#	6339
	6393#	6395	6449#	6451	6478#	6480	6513#	6515	6594#	6596	6665#	6667	6739#	6741	6899#
	6901	6975#	6977	7021#	7023	7157#	7159	7295#	7297	7348#	7350	7555#	7557	7733#	7735
	7825#	7828	7933#	7935	7967#	7970	8048#	8051	8140#	8142	8196#	8198	8321#	8323	8530#
	8532	8602#	8604	8671#	8673	8738#	8740	8904#	8906						
MULT	1#	1926#													
NEWST	1#	1926#	2927	2958	2988	3268	3305	3357	3401	3434	3620	3667	3717	3788	3831
	3889	3955	4014	4208	4328	4410	4490	4543	4595	4656	4715	4812	4886	4967	5035
	5116	5186	5266	5337	5410	5482	5535	5589	5740	5823	5910	5994	6079	6164	6223
	6280	6337	6393	6449	6478	6513	6594	6665	6739	6899	6975	7021	7157	7295	7348
	7555	7733	7826	7933	7968	8049	8140	8196	8321	8530	8602	8671	8738	8904	
OPCHK	2740#	4844	4867	4930	4948	5004	5017	5080	5097	5152	5166	5230	5245	5303	5317
	5440	5457	5769	5788	5803	5853	5876	5891	5939	5963	6024	6044	6059	6110	6129
	6144	6203	6260	6318	6374	6430	6476	6545	6628	6643	6701	6716	6777	6841	6931
	6961	7004	7019	7061	7076	7197	7212	7326	7343	7390	7421	7446	7466	7491	7517
	7606	7625	7631	7646	7670	7686	7691	7706	7728	7761	7767	7782	7857	7911	8000
	8043	8135	8168	8194	8240	8256	8264	8279	8353	8360	8375	8390	8433	8474	8509
	8527	8578	8648	8717	8786	8800	8951	8971	9000	11350	11357	11385			
POP	1#	1926#	9429	9430	9899	10033	10231	11099	11118	11122	11286	11560	11955	12004	12036
	12037														
PUSH	1#	1926#	9390	9392	9413	9888	10020	10205	10988	10999	11519	11929	11984	12017	12023
REPORT	1#	1926#													
RESDC	2737#	4875	4956	5025	5106	5175	5255	5326	5471	5812	5900	5983	6068	6153	6212
	6269	6327	6383	6439	6577	6652	6725	6850	6888	7528					
RKLOAD	2739#	4840	4863	4926	4945	5000	5014	5076	5093	5148	5162	5226	5241	5299	5313
	5436	5454	5502	5555	5765	5784	5799	5849	5872	5887	5935	5959	5973	6020	6040
	6055	6105	6125	6140	6199	6256	6314	6370	6426	6472	6500	6624	6639	6697	6712
	6773	6796	6837	6872	6927	6957	7000	7015	7057	7072	7092	7193	7208	7230	7322
	7339	7386	7417	7462	7513	7620	7642	7681	7702	7757	7778	7793	7805	7849	7868
	7883	7907	7996	8032	8073	8088	8132	8164	8220	8236	8252	8275	8312	8349	8371
	8386	8401	8446	8469	8505	8523	8574	8628	8697	8783	8796	8939	8948	8961	8968
	8983	11346	11381												
SCOPE	1821#	2934	2965	3030	3275	3315	3365	3408	3452	3630	3678	3730	3796	3838	3899
	3964	4038	4217	4344	4420	4499	4551	4605	4665	4724	4821	4896	4977	5045	5126
	5196	5276	5345	5420	5489	5542	5605	5748	5832	5918	6003	6088	6175	6234	6291
	6348	6404	6458	6487	6522	6602	6674	6753	6907	6983	7039	7175	7305	7367	7572
	7743	7835	7941	7979	8059	8151	8207	8335	8542	8582	8609	8679	8753	8919	9099
SETPRI	1#	1926#	11833												
SETTRA	12077#	12086	12087	12088	12089	12091	12093	12094	12095	12096	12097	12098	12099	12100	12101
	12102	12103	12104	12105	12106	12107	12108	12109	12110	12111	12112	12113	12114	12115	12116
	12117	12118													
SETUP	1#	1926#	2760												
SKIP	1#	1926#													
SLASH	1#	1926#													
SPACE	1926#														
STARS	1#	1926#	2184	2186	2193	2206	2281	2284	2927	2933	2958	2964	2988	3029	3268
	3274	3305	3314	3357	3364	3401	3407	3434	3451	3620	3629	3667	3677	3717	3729
	3788	3795	3831	3837	3889	3898	3955	3963	4014	4037	4208	4216	4328	4343	4410
	4419	4490	4498	4543	4550	4595	4604	4656	4664	4715	4723	4812	4820	4886	4895
	4967	4976	5035	5044	5116	5125	5186	5195	5266	5275	5337	5344	5410	5419	5482
	5488	5535	5541	5589	5604	5740	5747	5823	5831	5910	5917	5994	6002	6079	6087

CZR6KGO RK6 FCTNL CTRL DIAG
CZR6KG.P11 28-AUG-81 15:16

MACY11 30(1046) 28-AUG-81 15:17 PAGE 277
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0274

.\$SAVE	1#	1793#	11966
.\$SB2D	1#		
.\$SB20	1#		
.\$SCOP	1#	1793#	9228
.\$SIZE	1#	1793#	9145
.\$SUPR	1#		
.\$TRAP	1#	1793#	12054
.\$TYPB	1#		
.\$TYPD	1#	1793#	11507
.\$TYPE	1#	1793#	11411
.\$TYPO	1#	1793#	11574
.\$4OCA	1#	1793#	2150
.1170	1#		

. ABS. 074414 000

ERRORS DETECTED: 0

CZR6KG,CZR6KG.LST/SOL/CRF/NL:TOC=CZR6KG.SML,CZR6KG.P11
RUN-TIME: 35 44 3 SECONDS
RUN-TIME RATIO: 130/82=1.5
CORE USED: 38K (76 PAGES)