

RK611
RK06, RK07

RK6 FCTNL CTRL
CZR6KF0

AH-9130F-MC
FICHE 1 OF 2

NOV 1980
COPYRIGHT © 76-80
MADE IN USA



The main body of the document consists of a dense grid of approximately 15 columns and 25 rows of small, illegible text. Each cell in the grid appears to contain a small table or data entry, but the text is too faint to be transcribed accurately. The overall layout is that of a technical manual or a data reference sheet.

RK611
RK06, RK07

RK6 FCTNL CTRL
CZR6KF0

AH-9130F-MC
FICHE 2 OF 2

NOV 1980
COPYRIGHT © 76-80
MADE IN USA



[Faint, illegible text, likely a microfiche grid or data table]

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
46
47
48
49

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9128F-MC
PRODUCT NAME: CZR6KFO RK6 FCTNL CTRL
DATE: JUNE 1980
MAINTAINER: DIAGNOSTIC GROUP

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) 1976,1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

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
91
92
93
94

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 TIME
4.0	PROGRAM DESCRIPTION
5.0	ERROR REPORTING
6.0	SUBROUTINES
7.0	REVISION HISTORY

95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150

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

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
203
204
205
206

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

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
315
316
317
318

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.

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
371
372
373
374

**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 AVAILABLE 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 AVAILABLE 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.

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
483
484
485
486

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

487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542

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

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
595
596
597
598

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

599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654

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

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
707
708
709
710

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,

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
763
764
765
766

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.

767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822

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

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
875
876
877
878

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 100C00

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.

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
931
932
933
934

**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

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
987
988
989
990

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.

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
1043
1044
1045
1046

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 AVAILABLE DRIVES. ISSUE A SUBSYSTEM CLEAR. MAKE SURE ALL ATTENTIONS RESET.

TEST 107 SVAL AND ATTENTION FROM OTHER DRIVE

DO A RECALIBRATE ON ONE AVAILABLE DRIVE. DO A SELECT ON ANOTHER AVAILABLE 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 AVAILABLE.

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
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102

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 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

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
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158

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 RKMR1
T.CS1 T.CS2 T.DS T.ER T.ASF T.MR1

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.

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.

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
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214

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
\$STESTN \$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

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

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
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270

"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
ILLEGAL FUNCTION CODE

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

ERROR ITEM 10

(DESCRIPTION OF ERROR)
ERROR AT COMPLETION OF OPERATION
(DESCRIPITON OF OPERATION)

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
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326

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 RKMR1

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

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

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
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382

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

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

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,LRLOAD
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

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

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
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494

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

THIS ROUTINE GETS ALL THE DRIVE STATUS AND PLACES IT
IN \$REG10 THROUGH \$REG17. THESE REGISTERS 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
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

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
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550

CZR6
CZR6
21
21
21
21
21
21
21
21
21
21
21
21
21
21
22
22
22
22
22
22
22
22

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

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
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
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662

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.

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

DATA GENERATION AND COMPARE ROUTINE

CALLS: JSR R4,GENCOM
CONTROL WORD

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

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
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.

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
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774

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 GENCCM 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

1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791

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 'OPTTST' ROUTINE TO SUPPORT NEW PROCESSOR AND MEMORY OPTIONS

x

```
1792 ; *** REV 007 ***
1793
1794 .NLIST CND,MD,MC,TOC
1795 .LIST ME
1796 .ENABL ABS,AMA
1797 167400 $SWR= 167400
1798 000001 $TN= 1
1799 .TITLE CZR6KFO RK6 FCTNL CTRL DIAG
1800 ;*COPYRIGHT (C) 1976,1980
1801 ;*DIGITAL EQUIPMENT CORP.
1802 ;*MAYNARD, MASS. 01754
1803 ;*
1804 ;*
1805 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1806 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
1807 ;*
1808 .SBTTL OPERATIONAL SWITCH SETTINGS
1809 ;*
1810 ;* SWITCH USE
1811 ;* -----
1812 ;* 15 HALT ON ERROR
1813 ;* 14 LOOP ON TEST
1814 ;* 13 INHIBIT ERROR TYPEOUTS
1815 ;* 12 ABORT PROGRAM AFTER 20 ERRORS
1816 ;* 11 INHIBIT ITERATIONS
1817 ;* 10 BELL ON ERROR
1818 ;* 9 LOOP ON ERROR
1819 ;* 8 LOOP ON TEST IN SWR<7:0>
1820 .SBTTL BASIC DEFINITIONS
1821 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1822 001100 STACK= 1100
1823
1824 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
1825 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
1826
1827 ;*MISCELLANEOUS DEFINITIONS
1828 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
1829 000012 LF= 12 ;;CODE FOR LINE FEED
1830 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
1831 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
1832 177776 PS= 177776 ;;PROCESSOR STATUS WORD
1833 .EQUIV PS,PSW
1834 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
1835 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
1836 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
1837 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
1838
1839 ;*GENERAL PURPOSE REGISTER DEFINITIONS
1840 000000 R0= %0 ;;GENERAL REGISTER
1841 000001 R1= %1 ;;GENERAL REGISTER
1842 000002 R2= %2 ;;GENERAL REGISTER
1843 000003 R3= %3 ;;GENERAL REGISTER
1844 000004 R4= %4 ;;GENERAL REGISTER
1845 000005 R5= %5 ;;GENERAL REGISTER
1846 000006 R6= %6 ;;GENERAL REGISTER
1847 000007 R7= %7 ;;GENERAL REGISTER
```

1848	000006	SP=	%6	::STACK POINTER
1849	000007	PC=	%7	::PROGRAM COUNTER
1850				
1851		;*PRIORITY LEVEL DEFINITIONS		
1852	000000	PR0=	0	::PRIORITY LEVEL 0
1853	000040	PR1=	40	::PRIORITY LEVEL 1
1854	000100	PR2=	100	::PRIORITY LEVEL 2
1855	000140	PR3=	140	::PRIORITY LEVEL 3
1856	000200	PR4=	200	::PRIORITY LEVEL 4
1857	000240	PR5=	240	::PRIORITY LEVEL 5
1858	000300	PR6=	300	::PRIORITY LEVEL 6
1859	000340	PR7=	340	::PRIORITY LEVEL 7
1860				
1861		;*''SWITCH REGISTER'' SWITCH DEFINITIONS		
1862	100000	SW15=	100000	
1863	040000	SW14=	40000	
1864	020000	SW13=	20000	
1865	010000	SW12=	10000	
1866	004000	SW11=	4000	
1867	002000	SW10=	2000	
1868	001000	SW09=	1000	
1869	000400	SW08=	400	
1870	000200	SW07=	200	
1871	000100	SW06=	100	
1872	000040	SW05=	40	
1873	000020	SW04=	20	
1874	000010	SW03=	10	
1875	000004	SW02=	4	
1876	000002	SW01=	2	
1877	000001	SW00=	1	
1878		.EQUIV	SW09,SW9	
1879		.EQUIV	SW08,SW8	
1880		.EQUIV	SW07,SW7	
1881		.EQUIV	SW06,SW6	
1882		.EQUIV	SW05,SW5	
1883		.EQUIV	SW04,SW4	
1884		.EQUIV	SW03,SW3	
1885		.EQUIV	SW02,SW2	
1886		.EQUIV	SW01,SW1	
1887		.EQUIV	SW00,SW0	
1888				
1889		;*DATA BIT DEFINITIONS (BIT00 TO BIT15)		
1890	100000	BIT15=	100000	
1891	040000	BIT14=	40000	
1892	020000	BIT13=	20000	
1893	010000	BIT12=	10000	
1894	004000	BIT11=	4000	
1895	002000	BIT10=	2000	
1896	001000	BIT09=	1000	
1897	000400	BIT08=	400	
1898	000200	BIT07=	200	
1899	000100	BIT06=	100	
1900	000040	BIT05=	40	
1901	000020	BIT04=	20	
1902	000010	BIT03=	10	
1903	000004	BIT02=	4	

BASIC DEFINITIONS

1904	000002	BIT01=	2	
1905	000001	BIT00=	1	
1906		.EQUIV	BIT09,BIT9	
1907		.EQUIV	BIT08,BIT8	
1908		.EQUIV	BIT07,BIT7	
1909		.EQUIV	BIT06,BIT6	
1910		.EQUIV	BIT05,BIT5	
1911		.EQUIV	BIT04,BIT4	
1912		.EQUIV	BIT03,BIT3	
1913		.EQUIV	BIT02,BIT2	
1914		.EQUIV	BIT01,BIT1	
1915		.EQUIV	BIT00,BIT0	
1916				
1917		;*BASIC "CPU" TRAP VECTOR ADDRESSES		
1918	000004	ERRVEC=	4	::TIME OUT AND OTHER ERRORS
1919	000010	RESVEC=	10	::RESERVED AND ILLEGAL INSTRUCTIONS
1920	000014	TBITVEC=	14	::"T" BIT
1921	000014	TRTVEC=	14	::TRACE TRAP
1922	000014	BPTVEC=	14	::BREAKPOINT TRAP (BPT)
1923	000020	IOTVEC=	20	::INPUT/OUTPUT TRAP (IOT) **SCOPE**
1924	000024	PWRVEC=	24	::POWER FAIL
1925	000030	EMTVEC=	30	::EMULATOR TRAP (EMT) **ERROR**
1926	000034	TRAPVEC=	34	::"TRAP" TRAP
1927	000060	TKVEC=	60	::TTY KEYBOARD VECTOR
1928	000064	TPVEC=	64	::TTY PRINTER VECTOR
1929	000240	PIRQVEC=	240	::PROGRAM INTERRUPT REQUEST VECTOR
1930		.SBTTL MEMORY MANAGEMENT DEFINITIONS		
1931				
1932		;*KT11 VECTOR ADDRESS		
1933				
1934	000250	MMVEC=	250	
1935				
1936		;*KT11 STATUS REGISTER ADDRESSES		
1937				
1938	177572	SR0=	177572	
1939	177574	SR1=	177574	
1940	177576	SR2=	177576	
1941	172516	SR3=	172516	
1942				
1943		;*KERNEL "I" PAGE DESCRIPTOR REGISTERS		
1944				
1945	172300	KIPDR0=	172300	
1946	172302	KIPDR1=	172302	
1947	172304	KIPDR2=	172304	
1948	172306	KIPDR3=	172306	
1949	172310	KIPDR4=	172310	
1950	172312	KIPDR5=	172312	
1951	172314	KIPDR6=	172314	
1952	172316	KIPDR7=	172316	
1953				
1954		;*KERNEL "I" PAGE ADDRESS REGISTERS		
1955				
1956	172340	KIPAR0=	172340	
1957	172342	KIPAR1=	172342	
1958	172344	KIPAR2=	172344	
1959	172346	KIPAR3=	172346	

```

1960      172350      KIPAR4= 172350
1961      172352      KIPAR5= 172352
1962      172354      KIPAR6= 172354
1963      172356      KIPAR7= 172356
1964
1965      000210      AVECT1= 210          ;DEFINE RK611 VECTOR INTERRUPT
1966      000240      APRIOR= PR5         ;DEFINE RK611 PRIORITY
1967      177440      ABASE= 177440       ;DEFINE RK611 BASE BUS ADDRESS
1968
1969      .SBTTL      RK611 CONTROLLER REGISTER DEFINITION
1970
1971      000000      RKCS1= 0            ;CONTROL AND STATUS REGISTER 1
1972      000002      RKWC= 2             ;WORD COUNT REGISTER
1973      000004      RKBA= 4             ;BUS ADDRESS REGISTER
1974      000006      RKDA= 6             ;DESIRED TRACK SECTOR REGISTER
1975      000010      RKCS2= 10          ;CONTROL AND STATUS REGISTER 2
1976      000012      RKDS= 12          ;DRIVE STATUS REGISTER
1977      000014      RKER= 14          ;ERROR REGISTER
1978      000016      RKASOF= 16         ;ATTENTION SUMMARY AND OFFSET REGISTER
1979      000020      RKDCYL= 20        ;DESIRED CYLINDER REGISTER
1980      000024      RKDB= 24          ;DATA BUFFER
1981      000026      RKMR1= 26         ;MAINTENANCE REGISTER 1
1982      000034      RKMR2= 34         ;MAINTENANCE REGISTER 2
1983      000036      RKMR3= 36         ;MAINTENANCE REGISTER 3
1984      000030      RRECPS= 30        ;ECC POSITION INFORMATION
1985      000032      RRECPT= 32        ;ECC PATTERN INFORMATION
1986      000022      RKSPAR= 22        ;SPARE REGISTER
1987
1988      .SBTTL      DRIVE COMMANDS
1989
1990      000101      SELDRV= 101        ;SELECT DRIVE
1991      000103      PACK= 103         ;PACK ACKNOWLEDGE
1992      000105      CLEAR= 105        ;DRIVE CLEAR
1993      000107      UNLOAD= 107       ;UNLOAD
1994      000111      SRTSPL= 111       ;START SPINDLE
1995      000113      RECAL= 113        ;RECALIBRATE
1996      000115      OFFSET= 115       ;OFFSET
1997      000117      SEEK= 117         ;SEEK
1998      000121      RDDATA= 121       ;READ DATA
1999      000123      WRDATA= 123       ;WRITE DATA
2000      000125      RDHEAD= 125      ;READ HEADER
2001      000127      WRHEAD= 127      ;WRITE HEADER AND DATA
2002      000131      WRTCHK= 131      ;WRITE CHECK
2003      000300      INTR= 300         ;GENERATE INTERRUPT TO CPU
2004
2005      .SBTTL      CONTROL AND STATUS REGISTER 1 BITS
2006
2007      000001      GO= BIT0           ;GO BIT
2008      000100      IE= BIT6          ;INTERRUPT ENABLE
2009      000200      RDY= BIT7         ;CONTROLLER READY
2010      000400      BA16= BIT8        ;BUS ADDRESS BIT 16
2011      001000      BA17= BIT9        ;BUS ADDRESS BIT 17
2012      002000      CDT= BIT10       ;CONTROLLER DRIVE TYPE (0=RK06)
2013      004000      CTO= BIT11       ;CONTROLLER TIMED OUT WAITING FOR
2014      ;           ;DRIVE RESPONSE
2015      010000      CFMT= BIT12       ;CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)

```

2016	020000	SPAR=	BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
2017	040000	DI=	BIT14	:DRIVE INTERRUPT
2018	100000	CERR=	BIT15	:CONTROLLER ERROR
2019	100000	CCLR=	BIT15	:CONTROLLER CLEAR

2020
2021 .SBTTL CONTROL AND STATUS REGISTER 2 BITS

2022		DRVMSK=	7	:MASK FOR DRIVE SELECTION CODE
2023	000007	RLS=	BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
2024	000010	BAI=	BIT4	:BUS ADDRESS INCREMENT INHIBIT
2025	000020	SCLR=	BIT5	:CLEAR CONTROLLER AND ALL DRIVES
2026	000040	IR=	BIT6	:INPUT READY
2027	000100	OR=	BIT7	:OUTPUT READY
2028	000200	UFE=	BIT8	:UNIT FIELD ERROR
2029	000400	MDS=	BIT9	:MULTIPLE DRIVE SELECT
2030	001000	PGE=	BIT10	:PROGRAMMING ERROR
2031	002000	NEM=	BIT11	:NON-EXISTENT MEMORY
2032	004000	NED=	BIT12	:NON-EXISTENT DRIVE
2033	010000	UPE=	BIT13	:UNIBUS PARITY ERROR
2034	020000	WCE=	BIT14	:WRITE CHECK ERROR
2035	040000	DLT=	BIT15	:DATA LATE ERROR
2036	100000			

2037
2038 .SBTTL ERROR REGISTER BIT DEFINITION

2039		ILF=	BIT0	:ILLEGAL FUNCTION CODE
2040	000001	SKI=	BIT1	:SEEK INCOMPLETE
2041	000002	NXF=	BIT2	:NON-EXECUTABLE DRIVE FUNCTION
2042	000004	DRPAR=	BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
2043	000010	FMTE=	BIT4	:FORMAT ERROR
2044	000020	DTYPE=	BIT5	:DRIVE TYPE ERROR
2045	000040	ECH=	BIT6	:ECC HARD
2046	000100	BSE=	BIT7	:BAD SECTOR ERROR
2047	000200	HVRC=	BIT8	:HEADER VRC ERROR
2048	000400	COE=	BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
2049	001000	IDAE=	BIT10	:INVALID DISK ADDRESS ERROR
2050	002000	WLE=	BIT11	:WRITE LOCK ERROR
2051	004000	DTE=	BIT12	:DRIVE TIMING ERROR
2052	010000	OPI=	BIT13	:OPERATION (SEARCH) INCOMPLETE
2053	020000	UNS=	BIT14	:DRIVE UNSAFE
2054	040000	DCK=	BIT15	:DATA CHECK
2055	100000			

2056
2057 .SBTTL STATUS REGISTER BIT DEFINITION

2058		DRA=	BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF : THIS BIT IS RESET)
2059	000001	OFST=	BIT2	:DRIVE OFFSET
2060		ACLO=	BIT3	:AC LOW
2061	000004	SPDLSS=	BIT4	:SPEED LOSS
2062	000010	DROT=	BIT5	:DRIVE OFF TRACK
2063	000020	VV=	BIT6	:VOLUME VALID
2064	000040	DRDY=	BIT7	:DRIVE READY
2065	000100	DDT=	BIT8	:DRIVE TYPE (0=RK06)
2066	000200	WRL=	BIT11	:WRITE LOCK
2067	000400	PIP=	BIT13	:POSITIONING IN PROGRESS
2068	004000	DSC=	BIT14	:DRIVE STATUS CHANGE
2069	020000	SVAL=	BIT15	:STATUS VALID
2070	040000			
2071	100000			

2072
2073 .SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION
2074
2075 000017 MESMSK= 17 ;MESSAGE MASK
2076
2077 000020 PAT= BIT4 ;FORCE EVEN PARITY ON SERCON MESSAGE LINES
2078 000040 DMD= BIT5 ;DIAGNOSTIC MODE
2079 000100 MSP= BIT6 ;MAINTENANCE SECTOR PULSE
2080 000200 MIND= BIT7 ;MAINTENANCE INDEX
2081 000400 MCLK= BIT8 ;MAINTENANCE CLOCK
2082 001000 MERD= BIT9 ;MAINTENANCE ENCODED READ DATA
2083 002000 MEWD= BIT10 ;MAINTENANCE ENCODED WRITE DATA
2084 004000 PCA= BIT11 ;PRECOMPENSATION ADVANCE
2085 010000 PCD= BIT12 ;PRECOMPENSATION DELAY
2086 020000 ECCW= BIT13 ;ECC WORD IS BEING READ OR WRITTEN
2087 040000 WRTGAT= BIT14 ;WRITE GATE
2088 100000 RDGATE= BIT15 ;READ GATE
2089

2090 .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
2091
2092 000040 S.DRA= BIT5 ;DRIVE AVAILIABLE
2093 000100 S.VV= BIT6 ;VOLUME VALID
2094 000200 S.DRY= BIT7 ;DRIVE READY
2095 000400 S.TYPE= BIT8 ;DRIVE TYPE
2096 001000 S.FORM= BIT9 ;DRIVE FORMAT
2097 002000 S.OFF= BIT10 ;OFFSET
2098 004000 S.WRL= BIT11 ;WRITE LOCK
2099 010000 S.SPIN= BIT12 ;SPINDLE ON
2100 020000 S.PIP= BIT13 ;POSITIONING IN PROGRESS
2101 040000 S.DSC= BIT14 ;DRIVE STATUS CHANGE
2102

2103 .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
2104
2105 000040 S.ICYL= BIT5 ;ILLEGAL CYLINDER ADDRESS
2106 000100 S.ACLO= BIT6 ;AC LOW
2107 000200 S.FLT= BIT7 ;DRIVE FAULT
2108 000400 S.ILF= BIT8 ;ILLEGAL FUNCTION
2109 001000 S.PAR= BIT9 ;DRIVE DETECTED SERCON PARITY ERROR
2110 002000 S.SKI= BIT10 ;SEEK INCOMPLETE
2111 004000 S.WLE= BIT11 ;WRITE LOCK ERROR
2112 010000 S.SPLS= BIT12 ;SPEED LOSS
2113 020000 S.DROT= BIT13 ;DRIVE OFF TRACK
2114 040000 S.UNS= BIT14 ;DRIVE UNSAFE
2115

2116 .SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
2117
2118 000020 S.XDOK= BIT4 ;TRANSDUCER OK
2119 000040 S.HDHM= BIT5 ;HEADS HOME
2120 000100 S.BRHM= BIT6 ;BRUSHES HOME
2121 000200 S.DOOR= BIT7 ;DOOR INTERLOCKED
2122 000400 S.CART= BIT8 ;CARTRAGE INTERLOCK
2123 001000 S.SPOK= BIT9 ;SPEED OK
2124 002000 S.FWD= BIT10 ;FORWARD
2125 004000 S.REV= BIT11 ;REVERSE
2126 010000 S.LOAD= BIT12 ;HEADS LOADING
2127 020000 S.RTZ= BIT13 ;RETURN TO ZERO

```
2128      040000      S.UNLD= BIT14      ;HEADS UNLOADING
2129
2130      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
2131
2132      000020      S.SECT= BIT4      ;SECTOR ERROR
2133      000040      S.WCLK= BIT5      ;WRITE CLOCK AND NO WRITE GATE
2134      000100      S.WGAT= BIT6      ;WRITE GATE AND NO TRANSISTIONS
2135      000200      S.HDFL= BIT7      ;HEAD FAULT
2136      000400      S.MHD= BIT8      ;MULTIPLE HEAD SELECT
2137      001000      S.XERR= BIT9      ;INDEX ERROR
2138      002000      S.DIB= BIT10     ;DIBIT ERROR
2139      004000      S.PLO= BIT11     ;PLO ERROR
2140      010000      S.NMOV= BIT12     ;SEEK AND NO MOTION
2141      020000      S.LIMD= BIT13     ;LIMIT DETECT ON SEEK
2142      040000      S.BRKE= BIT14     ;SERVO-BRAKE
2143
2144      .SBTTL  COMMON MASKS
2145
2146      000007      M.DRV= 7      ;DRIVE CODE
2147      100000      M.PAR= BIT15     ;PARITY
2148      000003      M.ID= 3      ;BYTE ID
2149      017760      M.CDIF= 17760   ;CYLINDER DIFFERENCE/OFFSET
2150      017760      M.CADD= 17760   ;CYLINDER ADDRESS
2151      077770      M.SER= 77770    ;DRIVE SERIAL NUMBER
2152      000760      M.SECT= 760     ;SECTOR COUNT
2153      007000      M.HEAD= 7000    ;HEAD DECODE
2154      .SBTTL  TRAP CATCHER
2155
2156      000000      .=0
2157      ;*ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
2158      ;*A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
2159      ;*TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
2160      ;*THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
2161      ;*TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
2162      ;*LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
2163      ;*THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
2164      ;*      PC=YYYYYY UNEXPECTED TRAP TO XXX
2165      ;*AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
2166      ;*WHERE XXX=LOCATION OF ILLEGAL TRAP
2167      ;*      YYYYYY=PC AT TIME OF TRAP
2168      ;*NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
2169      ;*      CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.
2170
2171      000000 000000  $40CAT: HALT      ;;HALT
2172      000002 000737  BR      .-100    ;;BRANCH TO 177700 & TIME OUT (NOT ON
2173      ;;11/05)
2174      000004 001772  .WORD  START    ;;VECTOR TO STARTING ADDRESS
2175      000006 000340  .WORD  340      ;;WITH PRIORITY LEVEL 7
2176      000174 000174  .=174
2177      000174 000000  DISPREG: .WORD  0      ;;SOFTWARE DISPLAY REGISTER
2178      000176 000000  SWREG:  .WORD  0      ;;SOFTWARE SWITCH REGISTER
2179      .SBTTL  STARTING ADDRESS(ES)
2180      000200 000137 001772  JMP      @#START ;;GO TO START OF PROGRAM
2181      000204 000137 001762  JMP      RESTRT  ;JUMP TO RESTART ROUTINE
2182      000214 000214
2183      000214 000137 001752  JMP      PARM    ;JUMP TO OPERATOR ASSIGNED PARMETERS
```

2184
2185 000220 000220 001742
2186
2187
2188
2189
2190
2191 000224
2192 000024
2193 000024 000200
2194 000044
2195 000044 000224
2196 000224
2197
2198
2199
2200
2201 000224
2202 000224 000000
2203 000226 001276
2204 000230 000024
2205 000232 000074
2206 000234 000740
2207 000236 000031

```
      .=220  
      JMP SETCLK ;JUMP TO SET CLOCK ROUTINE  
.SBTTL APT PARAMETER BLOCK  
  
:*****  
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
:*****  
      .SX=      ;;SAVE CURRENT LOCATION  
      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM  
      200      ;;FOR APT START UP  
      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.  
      $APTHDR ;;POINT TO APT HEADER BLOCK  
      .=$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)  
$STMT: .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)
```

2208
2209
2210
2211
2212
2213
2214 001100
2215 001100
2216 001100 000000
2217 001102 000
2218 001103 000
2219 001104 000000
2220 001106 000000
2221 001110 000000
2222 001112 000000
2223 001114 000
2224 001115 001
2225 001116 000000
2226 001120 000000
2227 001122 000000
2228 001124 000000
2229 001126 000000
2230 001130 000000
2231 001132 000000
2232 001134 000
2233 001135 000
2234 001136 000000
2235 001140 177570
2236 001142 177570
2237 001144 177560
2238 001146 177562
2239 001150 177564
2240 001152 177566
2241 001154 000
2242 001155 002
2243 001156 012
2244 001157 000
2245 001160 000000
2246
2247 001162 000000
2248 001164 000000
2249 001166 000000
2250 001170 000000
2251 001172 000000
2252 001174 000000
2253 001176 000000
2254 001200 000000
2255 001202 000000
2256 001204 000000
2257 001206 000000
2258 001210 000000
2259 001212 000000
2260 001214 000000
2261 001216 000000
2262 001220 000000
2263 001222 000000

.SBTTL COMMON TAGS

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

SCMTAG: .=1100

\$TSTNM: .WORD 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
\$TMP0: .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

```
2264 001224 000000 $TMP1: .WORD 0 ;;USER DEFINED
2265 001226 000000 $TMP2: .WORD 0 ;;USER DEFINED
2266 001230 000000 $TMP3: .WORD 0 ;;USER DEFINED
2267 001232 000000 $TMP4: .WORD 0 ;;USER DEFINED
2268 001234 000000 $TMP5: .WORD 0 ;;USER DEFINED
2269 001236 000000 $TMP6: .WORD 0 ;;USER DEFINED
2270 001240 000000 $TMP7: .WORD 0 ;;USER DEFINED
2271 001242 000000 $TMP10: .WORD 0 ;;USER DEFINED
2272 001244 000000 $TMP11: .WORD 0 ;;USER DEFINED
2273 001246 000000 $TMP12: .WORD 0 ;;USER DEFINED
2274 001250 000000 $TMP13: .WORD 0 ;;USER DEFINED
2275 001252 000000 $TMP14: .WORD 0 ;;USER DEFINED
2276 001254 000000 $TMP15: .WORD 0 ;;USER DEFINED
2277 001256 000000 $TMP16: .WORD 0 ;;USER DEFINED
2278 001260 000000 $TMP17: .WORD 0 ;;USER DEFINED
2279 001262 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
2280 001264 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
2281 001266 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
2282 001272 077 $QUES: .ASCII !/? ;;QUESTION MARK
2283 001273 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
2284 001274 000012 $LF: .ASCIZ <12> ;;LINE FEED
2285 *****
2286 .SBTTL APT MAILBOX-ETABLE
2287 *****
2288 *****
2289 .EVEN
2290 001276 $MAIL: ;;APT MAILBOX
2291 001276 000000 $MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
2292 001300 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
2293 001302 000000 $TESTN: .WORD ATESTN ;;TEST NUMBER
2294 001304 000000 $PASS: .WORD APASS ;;PASS COUNT
2295 001306 000000 $DEVCT: .WORD ADEVCT ;;DEVICE COUNT
2296 001310 000000 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
2297 001312 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
2298 001314 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
2299 001316 $ETABLE: ;;APT ENVIRONMENT TABLE
2300 001316 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
2301 001317 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
2302 001320 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
2303 001322 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
2304 001324 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
2305 *
2306 * BITS 15-11=CPU TYPE
2307 * 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2308 * 11/70=06,PDQ=07,Q=10
2309 * BIT 10=REAL TIME CLOCK
2310 * BIT 9=FLOATING POINT PROCESSOR
2311 * BIT 8=MEMORY MANAGEMENT
2311 001326 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
2312 001327 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
2313 * MEM. TYPE BYTE -- (HIGH BYTE)
2314 * 900 NSEC CORE=001
2315 * 300 NSEC BIPOLAR=002
2316 * 500 NSEC MOS=003
2317 001330 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
2318 * MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2319 001332 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
```

2320	001333	000	\$MTYP2:	.BYTE	AMTYP2	::MEM.TYPE,BLK#2
2321	001334	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
2322	001336	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
2323	001337	000	\$MTYP3:	.BYTE	AMTYP3	::MEM.TYPE,BLK#3
2324	001340	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
2325	001342	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
2326	001343	000	\$MTYP4:	.BYTE	AMTYP4	::MEM.TYPE,BLK#4
2327	001344	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
2328	001346	000210	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
2329	001350	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
2330	001352	177440	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
2331	001354	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
2332	001356	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
2333	001360		\$ETEND:			
2334			.MEXIT			

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
2387
2388
2389
2390

001360

001360 000000
001362 057146
001364 060520
001366 060612

.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
;* SIGHTLY 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.DCYL	T.DA
2391							
2392	RKBA	RKWC					
2393	T.BA	T.WC					
2394							
2395	001370	000000					
2396	001372	000000					
2397	001374	060526					
2398	001376	060616					
2399							
2400							
2401							
2402							
2403							
2404							
2405							
2406	001400	000000					
2407	001402	057174					
2408	001404	060476					
2409	001406	060636					
2410							
2411							
2412							
2413							
2414							
2415							
2416							
2417							
2418							
2419							
2420							
2421							
2422							
2423							
2424							
2425							
2426							
2427							
2428							
2429							
2430							
2431							
2432							
2433							
2434							
2435							
2436							
2437							
2438							
2439							
2440							
2441							
2442							
2443							
2444							
2445							
2446							

```
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 RKMR1
; * T.CS1 T.CS2 T.DS T.ER T.AST T.MR1

EM3N: .WORD 0
DH002A DT003
DF003
```

* ERROR ITEMS 4,5,6,8,7 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 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.

* 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.

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
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502

```
: * 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  
: * ILLEGAL DRIVE ADDRESS ERROR  
: * CYLINDER OVERFLOW  
: * DRIVE TYPE ERROR  
: * FORMAT ERROR  
: * WRITE LOCK ERROR  
: *  
EM4N: .WORD 0  
DH4N: .WORD 0  
DT004  
DF004  
: *  
: * 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.  
: *  
EM5N: .WORD 0  
DH5N: .WORD 0
```

```
001410 000000  
001412 000000  
001414 060526  
001416 060646  
  
001420 000000  
001422 000000
```

2503 001424 060526
2504 001426 060676

DT005
DF005

2505
2506 :* ERROR ITEM 6
2507 :* (DESCRIPTION OF ERROR)
2508 :* ERROR IN OPERATION
2509 :* (DESCRIPTION OF OPERATION)
2510 :* TST NUM ERR PC DRIVE
2511 :* \$TESTN \$ERRPC DRVNUM
2512 :* RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA
2513 :* T.CS1 T.CS2 T.DS T.ER T.ASOF T.DCYL T.DA
2514 :* RKBA RKWC
2515 :* T.BA T.WC

2516 :*
2517 :* THE ERRORS REPORTED BY THIS FORMAT ARE:
2518 :* DATA CHECK
2519 :* WRITE CHECK
2520 :* ECC HARD
2521 :* DATA LATE
2522 :* OPERATION INCOMPLETE
2523 :* HEADER VRC ERROR
2524 :* BAD SECTOR ERROR

2526 001430 000000
2527 001432 000000
2528 001434 060526
2529 001436 060732

EM6N: .WORD 0
DH6N: .WORD 0
DT006
DF006

2530
2531 :* ERROR ITEM 7
2532 :* (DESCRIPTION OF ERROR)
2533 :* ERROR IN OPERATION
2534 :* (DESCRIPTION OF OPERATION)
2535 :* TST NUM ERR PC DRIVE
2536 :* \$TESTN \$ERRPC DRVNUM
2537 :* RKCS1 RKCS2 RKDS RKER RKASOF
2538 :* T.CS1 T.CS2 T.DS T.ER T.ASOF

2539 :*
2540 :* THE ERRORS REPORTED BY THIS FORMAT ARE:
2541 :* NON-EXISTANT DRIVE
2542 :* NON-EXISTANT MEMORY
2543 :* CONTROLLER TIME OUT
2544 :* UNIT FIELD ERROR
2545 :* MULTIPLE DRIVE SELECT
2546 :* PROGRAMMING ERROR
2547 :* UNIBUS PARITY ERROR
2548 :* ILLEGAL FUNCTION CODE

2549 :*
2550 :* DESCRIPTON OF OPERATION CAN BE ANY COMMAND, EITHER LEGAL OR ILLEGAL

2551
2552 001440 000000
2553 001442 000000
2554 001444 060526
2555 001446 060756

EM7N: .WORD 0
DH7N: .WORD 0
DT007
DF007

2556 :*
2557 :* ERROR ITEM 10
2558 :* (DESCRIPTION OF ERROR)

```
2559      :*      ERROR AT COMPLETION OF OPERATION
2560      :*      (DESCRIPITON OF OPERATION)
2561      :*      TST NUM ERR PC  DRIVE
2562      :*      $TESTN $ERRPC  DRVNUM
2563      :*      EXPT  1S
2564      :*      $REG10 $REG11
2565      :*
2566      :*      THE ERRORS REPORTED BY THIS FORMAT ARE SOFTWARE DETECTED BY
2567      :*      COMPARING EXPECTED RESULTS WITH ACTUAL RESULTS.  THE SPECIFIC
2568      :*      ERRORS ARE:
2569      :*      WORD COUNT INCORRECT
2570      :*      BUS ADDRESS INCORRECT
2571      :*      CYLINDER ADDRESS INCORRECT
2572      :*      TRACK ADDRESS INCORRECT
2573      :*      SECTOR ADDRESS INCORRECT
2574
2575      001450  000000      EM10N:  .WORD  0
2576      001452  060007      DH010
2577      001454  060576      DT015
2578      001456  060776      DF010
2579
2580      :*      ERROR ITEM 11
2581      :*      (ERROR INDICATOR OR STATUS BIT)
2582      :*      NOT SET AS A RESULT OF
2583      :*      (ANOTHER ERROR INDICATOR, STATUS BIT, OR OPERATION
2584      :*      TST NUM ERR PC  DRIVE
2585      :*      $TESTN $ERRPC  DRVNUM
2586      :*      RKCS1  RKCS2  RKDS  RKER  RKASOF  RKMR1
2587      :*      T.CS1  T.CS2  T.DS  T.ER  T.ASOF  T.MR1
2588
2589      001460  000000      EM11N:  .WORD  0
2590      001462  060133      DH011
2591      001464  060526      DT010
2592      001466  061016      DF011
2593
2594      :*      ERROR ITEM 12
2595      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2596      :*      'NOT RESET AS A RESULT OF'
2597
2598      001470  000000      EM12N:  .WORD  0
2599      001472  060162      DH012
2600      001474  060526      DT010
2601      001476  061016      DF011
2602
2603      :*      ERROR ITEM 13
2604      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2605      :*      'SET AS A RESULT OF'
2606
2607      001500  000000      EM13N:  .WORD  0
2608      001502  060213      DH013
2609      001504  060526      DT010
2610      001506  061016      DF011
2611
2612      :*      ERROR ITEM 14
2613      :*      THIS ERROR IS IDENTICAL TO ITEM 11 EXCEPT THE SECOND LINE IS:
2614      :*      'RESET AS A RESULT OF'
```

2615 001510 000000
2616 001512 060236
2617 001514 060526
2618 001516 061016
2619
2620
2621
2622
2623
2624
2625
2626
2627 001520 000000
2628 001522 057174
2629 001524 060576
2630 001526 061040
2631
2632
2633
2634
2635 001530 000000
2636 001532 000000
2637 001534 060604
2638 001536 061050
2639
2640
2641
2642 001540 000000
2643 001542 000000
2644 001544 000000
2645 001546 000000
2646 001550 000000
2647 001552 000000
2648 001554 000000
2649 001556 000000
2650 001560 000000
2651 001562 000000
2652 001564 000000
2653 001566 000000
2654 001570 000000
2655 001572 000000
2656 001574 000000
2657 001576 000000
2658
2659
2660 001600 000100
2661 001602 000000
2662 001604 000000
2663 001606
2664 001606 000
2665 001607 000
2666 001610 000000
2667 001612 000000
2668 001614 000000
2669 001616 000000
2670

EM14N: .WORD 0
DH014
DT010
DF011

:* ERROR ITEM 15
:* (HEADER WORD MISCOMPARE) OR (DATA MISCOMPARE)
:* TST NUM ERR PC DRIVE
:* \$TESTN \$ERRPC DRVNUM
:* GOOD BAD WORD NUM
:* \$REG10 \$REG11 \$REG12

EM15N: .WORD 0
DH002A
DT015
DF015

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

0
0
DT015A
DF016

.SBTTL REGISTER STORAGE FOR TEST

T.CS1: .WORD 0
T.WC: .WORD 0
T.BA: .WORD 0
T.DA: .WORD 0
T.CS2: .WORD 0
T.DS: .WORD 0
T.ER: .WORD 0
T.ASOF: .WORD 0
T.DCYL: .WORD 0
T.SPAR: .WORD 0
T.DB: .WORD 0
T.MR1: .WORD 0
T.ECPS: .WORD 0
T.ECPT: .WORD 0
T.MR2: .WORD 0
T.MR3: .WORD 0

.SBTTL REGISTER SETUP STORAGE ;PRESET WITH INTERRUPT ENABLE
L.CS1: .WORD 100
L.WC: .WORD 0
L.BA: .WORD 0
L.DA: .WORD 0
L.DS: .BYTE 0
L.DT: .BYTE 0
L.CS2: .WORD 0
L.ASOF: .WORD 0
L.DCYL: .WORD 0
L.MR1: .WORD 0
.SBTTL PROGRAM DEFINED VARIABLES

```
2671
2672 001620 000000      RKVEC: .WORD 0      ;RK VECTOR
2673 001622 000000      RKPRI: .WORD 0      ;RK PRIORITY
2674 001624 000000      SRTFLG: .WORD 0     ;START FLAG
2675                                     ; 0 = 200
2676                                     ; 1 = 214
2677                                     ; -1 = 204
2678 001626 000000      DRVNUM: .WORD 0     ;DRIVE UNDER TEST
2679 001630 000000      DRVBIT: .WORD 0     ;WORD TO STORE BIT TO INDICATE DRIVE UNDER TEST
2680 001632 000024      ERRCNT: .WORD ^D20 ;ERROR COUNTER TO LIMIT ERROR
2681                                     ;ERRORS REPORTED IN PROGRAM
2682 001634 000024      EQRLMT: .WORD ^D20 ;DATA COMPARE ERROR LIMIT
2683 001636 061200      BSF24P: .WORD BS24 ;POINTER TO BAD SECTORS 24 SECTOR MODE
2684                                     ; (FACTORY)
2685 001640 061054      BSF26P: .WORD BS26 ;POINTER TO BAD SECTORS 26 SECTOR MODE
2686                                     ; (FACTORY)
2687 001642 000000      BSS24P: .WORD 0     ;POINTER TO BAD SECTORS 24 SECT MODE
2688                                     ; (SOFTWARE)
2689 001644 000000      BSS26P: .WORD 0     ;POINTER TO BAD SECTORS 26 SECTOR MODE
2690                                     ; (SOFTWARE)
2691 001646 000000      BS26CT: .WORD 0     ;COUNT OF BAD SECTORS 26 SECTOR MODE
2692 001650 000000      BS24CT: .WORD 0     ;COUNT OF BAD SECTORS 24 SECTOR MODE
2693 001652 000764      MILCNT: .WORD ^D500 ;COUNT TO APPROXIMATE 1 MILL SEC
2694 001654 000017      TIMCNT: .WORD ^D15 ;COUNTER FOR MYTIME ROUTINE
2695                                     ;=1660
2696 001660 000000
2697 001662 000000      INTSET: .WORD 0     ;NON-ZERO IF RK06 INTERRUPT SINCE
2698                                     ;LAST CLEARED
2699 001664 000000      OPTFLG: .WORD 0     ;OPTION FLAGS
2700
2701                                     DOTST= BIT0      ;DRIVE 0 TO BE TESTED FLAG
2702                                     MEMSZB= BIT1     ;MEMORY SIZE REPORT FLAG
2703                                     MEMPYB= BIT2     ;MEMORY PARITY REPORT FLAG
2704                                     SRTINS= BIT3     ;START UP INSTRUCTIONS REPORTED FLAG
2705                                     PARPRE= BIT6     ;PARITY PRESENT FLAG
2706                                     BSERPT= BIT7     ;BSE HAS BEEN REPORTED
2707                                     FPFMT= BIT8     ;FIRST PASS FORMAT SWITCH
2708                                     CP1170= BIT10    ;CP IS 11/70 FLAG
2709                                     DRVRPT= BIT11    ;DRIVE NUMBERS REPORTED FLAG
2710                                     LCLKPR= BIT15    ;LINE CLOCK PRESENT
2711
2712 001666 000000      DRVDRP: .WORD 0     ;LIST OF DRIVES DROPPED
2713 001670 000000      MEMPAR: .WORD 0     ;WORD OF PARITY FLAGS
2714 001672 000000      CSRPTR: .WORD 0     ;POINTER TO CSR TO USE FOR SETTING BAD
2715                                     ; PARITY
2716 001674 000000      LCLKTK: .WORD 0     ;LINE CLOCK TICK COUNTER
2717 001676 000000      REFMT: .WORD 0     ;REFORMAT SWITCH FOR HALT
2718
2719 ; THE FOLLOWING 4 VARIABLES ARE USED TO STORE PARAMETERS FOR
2720 ; HEADER OR DATA COMPARE CONTINUATION PROCESS.
2721 001700 000000      DESHLD: .WORD 0     ;DESTINATION HOLD
2722 001702 000000      SRCHLD: .WORD 0     ;SOURCE HOLD
2723 001704 000000      WRDNUM: .WORD 0     ;WORD NUMBER IN ERROR HOLD
2724 001706 000000      WRDCNT: .WORD 0     ;WORDS LEFT IN COMPARE HOLD
2725 001710 177546      KWLADD: .WORD 177546 ;KW11-L ADDRESS
2726 001712 000100      KWLVEC: .WORD 100  ;KW11-L VECTOR
```

2727	001714	172100			MEMBAS: .WORD	172100		:MM11 ADDRESS
2728	001716	000114			MMVECA: .WORD	114		:MM11 VECTOR
2729								
2730	001720	000000			DTYPE: 0			:DRV TYP 0=RK06, 2000=RK07
2731	001722	000000			TYPTBL: 0			:DRV 0 0=RK06, 2000=RK07
2732	001724	000000						:DRV 1
2733	001726	000000						:DRV 2
2734	001730	000000						:DRV 3
2735	001732	000000						:DRV 4
2736	001734	000000						:DRV 5
2737	001736	000000						:DRV 6
2738	001740	000000						:DRV 7
2739								
2740								
2741								
2742								
2743								
2744					.SBTTL	PROGRAM SETUP		
2745								
2746	001742	012737	000002	001624	SETCLK: MOV	#2,SRTFLG		:SET START FLAG FOR CLOCK ADJUST
2747	001750	000412			BR	START1		
2748								
2749	001752	012737	000001	001624	PARM: MOV	#1,SRTFLG		:SET START FLAG FOR PARMETER START
2750	001760	000406			BR	START1		
2751								
2752	001762	012737	177777	001624	RESTRT: MOV	#-1,SRTFLG		:LOAD START FLAG FOR PARMETER START
2753	001770	000402			BR	START1		
2754								
2755	001772	005037	001624		START: CLR	SRTFLG		:CLEAR START FLAG
2756	001776	000005			START1: RESET			:RESET THE WHOLE SYSTEM
2757	002000	012706	001100		MOV	#STACK,SP		:INITIALIZE STACK POINTER
2758	002004	012746	000340		MOV	#PR7,-(SP)		:LOAD STACK TO LOCK OUT ALL INTERRUPTS
2759	002010	012746	002016		MOV	#1\$,-(SP)		:LOAD START OF PROGRAM
2760	002014	000002			RTI			:LOAD PSW
2761								
2762	002016	004737	045146		1\$: JSR	PC,\$TKINT		:INITIALIZE KEYBOARD
2763	002022	005037	001676		CLR	REFMT		:CLEAR REFORMAT SWITCH
2764					.SBTTL	INITIALIZE THE COMMON TAGS		
2765					::CLEAR	THE COMMON TAGS (\$CMTAG) AREA		
2766	002026	012706	001100		MOV	#SCMTAG,R6		::FIRST LOCATION TO BE CLEARED
2767	002032	005026			CLR	(R6)+		::CLEAR MEMORY LOCATION
2768	002034	022706	001140		CMP	#SWR,R6	::DONE?	
2769	002040	001374			BNE	.-6		::LOOP BACK IF NO
2770	002042	012706	001100		MOV	#STACK,SP		::SETUP THE STACK POINTER
2771					::INITIALIZE	A FEW VECTORS		
2772	002046	012737	032576	000020	MOV	#\$SCOPE,@#IOTVEC		::IOT VECTOR FOR SCOPE ROUTINE
2773	002054	012737	000340	000022	MOV	#340,@#IOTVEC+2		::LEVEL 7
2774	002062	012737	033614	000030	MOV	#\$ERROR,@#EMTVEC		::EMT VECTOR FOR ERROR ROUTINE
2775	002070	012737	000340	000032	MOV	#340,@#EMTVEC+2		::LEVEL 7
2776	002076	012737	047006	000034	MOV	#\$STRAP,@#TRAPVEC		::TRAP VECTOR FOR TRAP CALLS
2777	002104	012737	000340	000036	MOV	#340,@#TRAPVEC+2		::LEVEL 7
2778	002112	012737	046630	000024	MOV	#\$SPWRDN,@#PWRVEC		::POWER FAILURE VECTOR
2779	002120	012737	000340	000026	MOV	#340,@#PWRVEC+2		::LEVEL 7
2780	002126	013737	032026	032020	MOV	SENDCT,\$EOPCT		::SETUP END-OF-PROGRAM COUNTER
2781	002134	005037	001262		CLR	\$TIMES		::INITIALIZE NUMBER OF ITERATIONS
2782	002140	005037	001264		CLR	\$ESCAPE		::CLEAR THE ESCAPE ON ERROR ADDRESS

```
2783 002144 112737 000001 001115      MOV#B #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
2784 002152 012737 002152 001106      MOV #,$SLPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2785 002160 012737 002160 001110      MOV #,$SLPERR      ;;SETUP THE ERROR LOOP ADDRESS
2786                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2787                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2788 002166 013746 000004      MOV @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2789 002172 012737 002226 000004      MOV #64$,@#ERRVEC  ;;SET UP ERROR VECTOR
2790 002200 012737 177570 001140      MOV #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
2791 002206 012737 177570 001142      MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2792 002214 022777 177777 176716      CMP #-1,@SWR      ;;TRY TO REFERENCE HARDWARE SWR
2793 002222 001012      BNE 66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2794                                     ;;AND THE HARDWARE SWR IS NOT = -1
2795 002224 000403      BR 65$           ;;BRANCH IF NO TIMEOUT
2796 002226 012716 002234      64$: MOV #65$, (SP)   ;;SET UP FOR TRAP RETURN
2797 002232 000002      RTI
2798 002234 012737 000176 001140      65$: MOV #SWREG,SWR  ;;POINT TO SOFTWARE SWR
2799 002242 012737 000174 001142      MOV #DISPREG,DISPLAY
2800 002250 012637 000004      66$: MOV (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2801
2802 002254 005037 001304      CLR $PASS         ;;CLEAR PASS COUNT
2803 002260 132737 000200 001317      BITB #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
2804 002266 001403      BEQ 67$          ;;YES,USE NON-APT SWITCH
2805 002270 012737 001320 001140      MOV #$$SWREG,SWR  ;;NO,USE APT SWITCH REGISTER
2806 002276
2807      67$:
2808      .SBTTL TYPE PROGRAM NAME
2809      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2810      INC #-1        ;;FIRST TIME?
2811      BNE 68$        ;;BRANCH IF NO
2812      CMP #SENDAD,@#42 ;;ACT-11?
2813      BEQ 68$        ;;BRANCH IF YES
2814      TYPE ,69$     ;;TYPE ASCIZ STRING
2815      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2816      TST @#42      ;;ARE WE RUNNING UNDER XXDP/ACT?
2817      BNE 70$        ;;BRANCH IF YES
2818      CMPB $ENV,#1  ;;ARE WE RUNNING UNDER APT?
2819      BEQ 70$        ;;BRANCH IF YES
2820      CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
2821      BNE 71$        ;;BRANCH IF NO
2822      GTSWR        ;;GET SOFT-SWR SETTINGS
2823      BR 71$
2824      70$: MOV#B #1,$AUTOB ;;SET AUTO-MODE INDICATOR
2825      71$:
2826      BR 68$        ;;GET OVER THE ASCIZ
2827      ;;69$: .ASCIZ <CRLF>*CZR6KFO RK6 FCTNL CTRL DIAG*<CRLF>
2828      68$:
2829      BITB #BIT7,$ENVM ;;TEST IF DO NOT SIZE
2830      BNE 3$         ;;YES - SKIP
2831      JSR PC,$SIZE
2832      CMP $LSTBK,#740 ;;MAKE SURE MEMORY IS SUFFICIENT
2833      BHIS 2$        ;;YES - SKIP
2834      TYPE ,OPRO05  ;;MESSAGE (NOT ENOUGH MEMORY)
2835      MOV #1,$EOPCT ;;FORCE END OF PROGRAM
2836      JMP $EOP
2837      2$:
2838      MOV #IBUFF,RO  ;;SET UP TO CLEAR 2000
      MOV #2000,R1    ;;WORDS OF BUFFER SPACE
```

2839	002472	005020			30\$:	CLR	(R0)+	:LOOP -
2840	002474	005301				DEC	R1	:UNTIL -
2841	002476	001375				BNE	30\$:DONE
2842	002500	122737	000013	000041		CMPB	#13,@#41	:LOAD FROM XXDP
2843	002506	001003				BNE	99\$:BRANCH IF NOT
2844	002510	104401	051146			TYPE	.OPR007	
2845	002514	000000				HALT		
2846	002516				99\$:			
2847	002516	013700	032574			MOV	\$LSTBK,R0	:GET LAST BANK
2848	002522	012701	000006			MOV	#6,R1	:SET SHIFT COUNT
2849	002526	013703	032572			MOV	\$LSTAD,R3	:GET LAST ADDRESS
2850	002532	005004				CLR	R4	:CLEAR R4 FOR OVERFLOW
2851	002534	005737	032316			TST	\$KT11	:MEM MANAGE PRESENT?
2852	002540	100005				BPL	23\$:NO - SKIP
2853	002542	006100			22\$:	ROL	R0	:SHIFT BANK LEFT
2854	002544	006104				ROL	R4	:ADD IN CARRY
2855	002546	005301				DEC	R1	:DECREMENT COUNT
2856	002550	001374				BNE	22\$:LOOP IF NOT ZERO
2857	002552	050003				BIS	R0,R3	:SET BANK BITS IN LAST ADDRESS
2858	002554	112737	000001	001327	23\$:	MOVB	#1,\$MTYP1	:FORCE MEMORY TYPE TO 1
2859	002562	110437	001326			MOVB	R4,\$MAMS1	:STORE UPPER MEMORY ADDRESS
2860	002566	010337	001330			MOV	R3,\$MADR1	:STORE LOWER ADDRESS
2861	002572	032737	000010	001664	3\$:	BIT	#SRTINS,OPTFLG	:TEST IF ALREDY REPORTED
2862	002600	001005				BNE	24\$:YES - SKIP
2863	002602	104401	051676			TYPE	.OPR016	:TYPE STARTUP INSTRUCTIONS
2864	002606	052737	000010	001664		BIS	#SRTINS,OPTFLG	:SET REPORTED FLAG
2865	002614				24\$:			
2866	002614	022737	000001	001624		CMP	#1,SRTFLG	:CHECK IF PARAMETER START
2867	002622	001122				BNE	15\$:NO, START TESTING
2868	002624	104401	050766		5\$:	TYPE	.OPR001	:TYPE 'RK611 BUS ADDRESS () ='
2869	002630	013746	001352			MOV	\$BASE,-(SP)	:SAVE \$BASE FOR TYPEOUT
2870	002634	104402				TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
2871	002636	104401	051015			TYPE	.OPR002	
2872	002642	104412				RDOCT		:GET VALUE
2873	002644	012637	001222			MOV	(SP)+,\$TMPO	
2874	002650	001407				BEQ	7\$:CHECK IF <CR>
2875	002652	022737	160000	001222		CMP	#160000,\$TMPO	:CHECK IF IN I/O PAGE
2876	002660	101361				BHI	5\$	
2877	002662	013737	001222	001352		MOV	\$TMPO,\$BASE	:LOAD NEW BUS ADDRESS
2878	002670	104401	051023		7\$:	TYPE	.OPR003	:TYPE 'RK611 VECTOR ADDRESS () ='
2879	002674	013746	001346			MOV	\$VECT1,-(SP)	:GET \$VECT1 FOR TYPOUT
2880	002700	042716	160000			BIC	#160000,(SP)	:CLEAR PRIORITY BITS
2881	002704	104402				TYPOC		
2882	002706	104401	051015			TYPE	.OPR002	
2883	002712	104412				RDOCT		:GET VALUE
2884	002714	012637	001222			MOV	(SP)+,\$TMPO	
2885	002720	001412				BEQ	10\$:CHECK IF <CR>
2886	002722	022737	001000	001222		CMP	#1000,\$TMPO	
2887	002730	101757				BLOS	7\$:CHECK IF LEGAL
2888	002732	042737	017777	001346		BIC	#17777,\$VECT1	:CLEAR OLD VECTOR
2889	002740	053737	001222	001346		BIS	\$TMPO,\$VECT1	:LOAD NEW VECTOR ADDRESS
2890	002746	104401	051053		10\$:	TYPE	.OPR004	:TYPE 'RK611 PRIORITY () ='
2891	002752	005046				CLR	-(SP)	
2892	002754	113716	001347			MOVB	\$VECT1+1,(SP)	
2893	002760	006216				ASR	(SP)	:SHIFT 5 BITS RIGHT
2894	002762	006216				ASR	(SP)	


```
2895 002764 006216 ASR (SP)
2896 002766 006216 ASR (SP)
2897 002770 006216 ASR (SP)
2898 002772 104402 TYPOC
2899 002774 104401 051015 TYPE .OPR002
2900 003000 104412 RDOCT ;GET VALUE
2901 003002 012637 001222 MOV (SP)+,$TMPO
2902 003006 001430 BEQ 15$ ;CHECK IF <CR>
2903 003010 022737 000007 001222 CMP #7,$TMPO ;CHECK IF LEGAL
2904 003016 103753 BLO 10$
2905 003020 022737 000004 001222 CMP #4,$TMPO
2906 003026 101347 BHI 10$
2907 003030 006337 001222 ASL $TMPO ;SHIFT 5 BITS LEFT
2908 003034 006337 001222 ASL $TMPO
2909 003040 006337 001222 ASL $TMPO
2910 003044 006337 001222 ASL $TMPO
2911 003050 006337 001222 ASL $TMPO
2912 003054 042737 160000 001347 BIC #160000,$VECT1+1 ;CLEAR OLD PRIORITY
2913 003062 053737 001222 001347 BIS $TMPO,$VECT1+1 ;LOAD RK611 PRIORITY
2914 003070 005037 001304 15$: CLR $PASS ;SET PASS COUNT TO ZERO
2915 003074 005037 001666 CLR DRVDRP ;CLEAR DROPPED DRIVES LIST
2916 003100 042737 004000 001664 BIC #DRVDRPT,OPTFLG ;CLEAR DRIVE #'S REPORTED FLAG
2917 003106 004737 034522 JSR PC,OPTTST ;SETUP PARITY CHECK & CLOCK
2918 003112 013700 001346 MOV $VECT1,R0 ;STORE VECTOR FOR USE
2919 003116 042700 160000 BIC #160000,R0 ;CLEAR PRIORITY BITS
2920 003122 010037 001620 MOV R0,RKVEC
2921 003126 012710 034442 MOV #INTHLR,(R0) ;SETUP INTERRUPT ADDRESS
2922 003132 113737 001347 001622 MOV $VECT1+1,RKPRI ;STORE PRIORITY FOR USE
2923 003140 013702 001352 MOV $BASE,R2 ;SET BASE ADDRESS
2924 003144 005037 001264 CLR $ESCAPE ;CLEAR ESCAPE
2925 003150 012746 000000 NEWPAS: MOV #PRO,-(SP) ;SET PRIORITY
2926 003154 012746 003162 MOV #16$,-(SP)
2927 003160 000002 RTI
2928 003162 16$:
2929
2930 .SBTTL **BASIC INTERFACE AND OPTION TESTS
2931 :*****
2932 :*TEST 1 RK611 BASE ADDRESS TEST
2933 :* CHECK THAT READING THE RK611 BASE ADDRESS (RKCS1) DOES NOT
2934 :* CAUSE A NON-EXISTANT MEMORY TRAP. IF A TRAP OCCURS
2935 :* THE PROGRAM IS HALTED.
2936 :*****
2937 :*****
2938 003162 000004 TST1: SCOPE
2939 003164 012737 000100 001262 MOV #100,$TIMES ;DO 100 ITERATIONS
2940 003172 012706 001100 MOV #STACK,SP ;CLEAN OFF STACK
2941 003176 012701 000004 MOV #4,R1 ;SET POINTER TO VECTOR
2942 003202 012146 MOV (R1)+,-(SP) ;STORE OLD VECTOR CONTENTS
2943 003204 011146 MOV (R1),-(SP)
2944 003206 012701 000004 MOV #4,R1 ;RESET POINTER
2945 003212 012721 034434 MOV #NEXINT,(R1)+ ;SET VECTOR TO NEM TEST HANDLER
2946 003216 012711 000340 MOV #PR7,(R1) ;SET PRIORITY
2947 003222 013702 001352 MOV $BASE,R2 ;SET POINTER TO RK611 BASE ADDRESS
2948 003226 005037 001662 CLR INTSET ;CLEAR INTERRUPT COUNTER
2949 003232 012762 000000 000000 MOV #0,RKCS1(R2) ;WRITE CS1 TO SEE IN NEM WILL SET
2950 003240 000240 NOP
```

```
2951 003242 000240 NOP
2952 003244 000240 NOP
2953 003246 005737 001662 TST INTSET ;TEST IF COUNTER IS 0
2954 003252 001406 BEQ 1$ ;YES - SKIP ERROR REPORT
2955 003254 012737 053315 001360 MOV #EM1,EM1N ;MESSAGE (NON-EXISTANT MEMORY TRAP ERR)
2956 003262 104001 ERROR 1
2957 003264 000137 043724 JMP CTRHLT ;GO TO CONTROLLED HALT
2958 003270 012701 000006 1$: MOV #6,R1 ;RESTORE VECTOR
2959 003274 012611 MOV (SP)+,(R1)
2960 003276 012641 MOV (SP)+,-(R1)
```

```
*****
:*TEST 2 INTERRUPT VECTOR ADDRESS TEST
:* CHECK THAT THE INTERRUPT VECTOR FOR THE RK611 IS SET TO THE
:* EXPECTED ADDRESS. IF INTERRUPT VECTOR IS IN ERROR,
:* THE PROGRAM IS HALTED.
```

```
*****
2969 003300 000004 TST2: SCOPE
2970 003302 012737 000100 001262 MOV #100,$TIMES ;;DO 100 ITERATIONS
2971 003310 012762 005000 000010 MOV #CLR,RKCS2(R2) ;CLEAR SUBSYSTEM, SPECIFICALLY TO
2972 ;CLEAR ANY OLD INTERRUPTS
2973 003316 005037 001662 CLR INTSET ;CLEAR INTERRUPT COUNTER
2974 003322 012762 000300 000000 MOV #RDY!IE,RKCS1(R2) ;WRITE CS1 TO FORCE INTERRUPT
2975 003330 000240 NOP
2976 003332 000240 NOP
2977 003334 000240 NOP
2978 003336 005737 001662 TST INTSET ;TEST IF INTERRUPT OCCURRED
2979 003342 001011 BNE 2$ ;YES - SKIP ERROR REPORT
2980 003344 105737 001103 TSTB $ERFLG ;TEST IF ERFLG ALREADY SET. IF SO THE
2981 ;INTERRUPT WENT TO THE WRONG VECTOR
2982 ;AND MESSAGE HAS BEEN REPORTED.
2983 003350 001004 BNE 2$ ;THEREFORE - EXIT
2984 003352 012737 053315 001360 MOV #EM1,EM1N ;MESSAGE (NO INTERRUPT)
2985 003360 104001 ERROR 1
2986 003362 000137 043724 2$: JMP CTRHLT ;GO TO CONTROLLED HALT
2987 003366 3$:
```

.SBTTL **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
```

2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006

```

3007      :*      THE RK06, DRIVE 0 WILL ONLY BE TESTED IF THE PARAM
3008      :*      START (214) WAS USED. IF THE AUTOMATIC START (200)
3009      :*      IS USED, DRIVE 0 IS NOT TESTED. THE RESTART (204)
3010      :*      WILL RETAIN THE TEST STATUS OF DRIVE 0.
3011      :*
3012      :*      IF THE PARAM START IS USED, THE OPERATOR MUST
3013      :*      EITHER PLACE DRIVE 0 OFF LINE IF IT IS NOT TO BE TESTED
3014      :*      OR UNLOADED AND A SCRATCH MEDIA MOUNTED IF IT IS TO
3015      :*      BE TESTED. THE PROGRAM WILL MONITOR OFF LINE AND VOLUME
3016      :*      VALID TO DETERMINE THE TEST STATUS OF DRIVE 0.
3017      :*
3018      :*      ALL DRIVES TO BE TESTED MUST BE ON-LINE, CYCLED UP, AND
3019      :*      WRITE LOCK RESET. ADDRESSES OF DRIVES THAT ARE NON-
3020      :*      EXISTANT EITHER BECAUSE THE DRIVE DOES NOT EXIST OR IS OFF-
3021      :*      LINE ARE USED TO VERIFY NON-EXISTANT DRIVE ERROR DETECTION.
3022      :*      DRIVES THAT ARE ON-LINE BUT NOT CYCLED UP OR ARE WRITE
3023      :*      LOCKED ARE NOT TESTED.
3024      :*
3025      :*      AT COMPLETION OF THE TEST A MESSAGE WILL BE GIVEN TO
3026      :*      IDENTIFY THE DRIVES TO BE USED IN TESTING.
3027      :*
3028      :*      NOTE: THIS TEST MUST BE RUN AT LEAST ONCE BEFORE
3029      :*      ANY OTHER TEST THAT FOLLOWS.
3030      :*      THE DETERMINATION OF WHETHER EACH DRIVE IS AN
3031      :*      RK06 OR AN RK07 IS MADE IN THIS TEST, AND A
3032      :*      TABLE IS MADE ACCORDINGLY.
3033      :*      *****
3034      003366 000004          TST3:  SCOPE
3035      003370 012737 000062 001262  MOV    #50, $TIMES      ;;DO 50. ITERATIONS
3036      003376 005037 001720          CLR    DTYPE          ;:ASSUME RK06
3037      003402 104416          TSSINIT
3038      003404 104003          ERROR   3             ;:CALL SUBSYSTEM CLEAR AND TEST
3039
3040      003406 012746 000000          MOV    #PRO,-(SP)      ;:SET PROCESSOR PRIORITY TO ALLOW
3041      003412 012746 003420          MOV    #1$,-(SP)     ;:RK611 INTERRUPTS
3042      003416 000002          RTI
3043
3044      003420 013701 001620          1$:  MOV    RKVEC,R1      ;:GET VECTOR
3045      003424 012721 034442          MOV    #INTHLR,(R1)+ ;:LOAD INTERRUPT VECTOR
3046      003430 012711 000340          MOV    #PR7,(R1)
3047      003434 012703 001354          MOV    #$DEVN,R3     ;:GET ADDRESS OF DEVICE MAP
3048      003440 132737 000200 001317  BITB   #BIT7,$ENVN   ;:TEST IF SHOULD SIZE
3049      003446 001007          BNE    50$           ;:NO - SKIP DRIVE SIZING.
3050      003450 005737 001304          TST    $PASS        ;:TEST IF FIRST PASS
3051      003454 001004          BNE    50$           ;:NO - SKIP TO DRIVE SELECT TEST
3052      003456 032737 004000 001664  BIT    #DRVRPT,OPTFLG ;:TEST IF DRIVE #'S REPORTED
3053      003464 001402          BEQ    92$           ;:NO - GOTO SIZING
3054      003466 000137 004142          50$:  JMP    11$
3055      003472 005013          92$:  CLR    (R3)         ;:CLEAR DEVICE MAP
3056      003474 123727 000041 000013  CMPB   @#41,#13     ;:TEST IF RK06 IS LOAD DEVICE
3057      003502 001111          BNE    77$           ;:NO - SKIP
3058      003504 022737 000001 001624  CMP    #1,SRTFLG    ;:WAS START AT PARAM?
3059      003512 001411          BEQ    2$            ;:YES - SKIP
3060      003514 104401 051146          TYPE   ,OPR007      ;:NO TEST OF DRIVE 0
3061      003520 052737 000001 001666  BIS    #BIT0,DRVDRP ;:SET DRIVE 0 DROPPED
3062      003526 042737 000001 001664  BIC    #DOTST,OPTFLG ;:DR FLAG - NO TEST DRIVE 0

```

3063	003534	000477				BR	7\$		
3064	003536	104401	051146		2\$:	TYPE	,OPR006		;MESSAGE - SWAP PACK ON DRIVE OFF LINE.
3065	003542	005037	001610			CLR	L.CS2		;SET TO DRIVE 0
3066	003546	005037	001232			CLR	\$TMP4		;CLEAR FOR USE AS A SWITCH
3067	003552	012737	000101	001600	3\$:	MOV	#SELDRV,L.CS1		;LOAD FOR SELECT
3068	003560	012737	003646	001264		MOV	#4\$,\$ESCAPE		;SET UP ESCAPE IN CASE OF ERR
3069	003566	104417				TLOADRK			;LOAD RK & DO SELECT
3070									
3071	003570	104423				TWAT16			;WAIT 16MS FOR COMPLETION
3072	003572	104002				ERROR	2		;NOT DONE ON TIME
3073									
3074	003574	104420				TGETRK			;GET RK REGISTER
3075	003576	032737	100000	001540		BIT	#CERR,T.CS1		;TEST IF CERR
3076	003604	001431				BEQ	5\$;NO - SKIP
3077	003606	032737	010000	001550		BIT	#NED,T.CS2		;TEST IF NED
3078	003614	001014				BNE	4\$;YES - SKIP
3079	003616	032737	000040	001554		BIT	#DTYE,T.ER		;ELSE SEE IF DRIVE TYPE ERROR
3080	003624	001406				BEQ	26\$;BR IF NO
3081									
3082	003626	012737	002000	001720		MOV	#CDT,DTYPE		;ELSE SETUP FOR RK07
3083									
3084	003634	104416				TSSINIT			;INIT SUBSYS
3085	003636	104003				ERROR	3		;NOT SUCCESSFUL
3086	003640	000744				BR	3\$;AND TRY AGAIN
3087									
3088	003642	104421			26\$:	TCHKOP			;CHECK THE OPERATION AND REPORT THE ERROR
3089	003644	104004				ERROR	4 ;OR5,6,7		;AFTER THE ERROR IS REPORTED THE TEST
3090									;IS ABORTED
3091	003646	104401	051146		4\$:	TYPE	,OPR007		;TYPE NO TEST OF DRIVE 0
3092	003652	042737	000001	001664		BIC	#DOTST,OPTFLG		;DR FLAG - NO TEST OF DRIVE 0
3093	003660	052737	000001	001666		BIS	#BIT0,DRVDRP		;SET DRV 0 AS DROPPED
3094	003666	000422				BR	7\$;SKIP OVER WAIT FOR PACK MOUNT
3095	003670	005737	001232		5\$:	TST	\$TMP4		;TEST FLAG DRIVE READY HAS RESET
3096	003674	001010				BNE	6\$;YES - SKIP TO CHECK IF IT IS SET AGAIN
3097	003676	032737	000200	001552		BIT	#DRDY,T.DS		;ELSE CHECK READY
3098	003704	001322				BNE	3\$;STILL SET - GET STATUS AGAIN
3099	003706	012737	177777	001232		MOV	#-1,\$TMP4		;ELSE SET FLAG TO INDICATE READY WENT LOW
3100	003714	000716				BR	3\$;GO GET STATUS AGAIN
3101									
3102	003716	032737	000200	001552	6\$:	BIT	#DRDY,T.DS		;TEST IF READY SET AGAIN
3103	003724	001712				BEQ	3\$;NO - GO GET STATUS AGAIN
3104	003726	052737	000001	001664	77\$:	BIS	#DOTST,OPTFLG		;ELSE SET DRV 0 TEST FLAG.
3105									
3106	003734	012737	004042	001264	7\$:	MOV	#35\$,\$ESCAPE		;SET UP ESCAPE IN CASE OF ERR
3107	003742	005000				CLR	RO		;CLEAR FOR DRIVE NUMBER COUNTER
3108	003744	012701	000001			MOV	#1,R1		;SET BIT 0 AS DRIVE SELECTOR
3109	003750	005037	001720			CLR	DTYPE		;INIT TO RK06
3110									
3111	003754	032737	000001	001664		BIT	#DOTST,OPTFLG		;TEST DRIVE 0?
3112	003762	001440				BEQ	9\$;NO - SKIP
3113									
3114	003764	104416			8\$:	TSSINIT			;INITIALZE SUBSYSTEM
3115	003766	104003				ERROR	3		;ERROR IF NOT SUCCESSFUL
3116									
3117	003770	010037	001610			MOV	RO,L.CS2		;LOAD DRIVE NUMBER
3118	003774	012737	000101	001600		MOV	#SELDRV,L.CS1		;LOAD DRIVE SELECT

```

3119 004002 104417          TLOADRK          ;LOAD RK REGS
3120
3121 004004 104423          TWAT16          ;WAIT FOR INTERRUPT
3122 004006 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
3123
3124 004010 104420          TGETRK          ;GET RK REGS
3125 004012 032737 100000 001540  BIT #CERR,T.CS1 ;ERROR?
3126 004020 001031          BNE 10$        ;YES - SKIP
3127 004022 032737 000200 001552  BIT #DRDY,T.DS  ;ELSE TEST IF DRIVE READY
3128 004030 001404          BEQ 35$        ;NO - SKIP
3129 004032 032737 004000 001552  BIT #WRL,T.DS  ;ELSE TEST IF WRITE LOCKED
3130 004040 001403          BEQ 36$        ;NO - SKIP
3131 004042 050137 001666          35$: BIS R1,DRVDRP ;SET THIS BIT AS DROPPED DRIVE
3132 004046 000406          BR 9$
3133 004050          36$:
3134 004050 050113          BIS R1,(R3)    ;SET BIT - DRIVE PRESET IN MAP
3135 004052 006300          ASL R0
3136 004054 013760 001720 001722  MOV DTYP,TYPTBL(R0) ;SET TABLE WITH DRIVE TYPE INFO
3137 004062 006200          ASL R0
3138
3139 004064 005200          9$: INC R0        ;BUMPS TO NEXT DRIVE
3140 004066 006301          ASL R1        ;SHIFT DRIVE SELECTOR TO NEXT DRIVE.
3141 004070 032701 000400          BIT #BIT8,R1  ;WAS LAST DRIVE DONE?
3142 004074 001022          BNE 11$       ;BR IF YES
3143 004076 005037 001720          CLR DTYP     ;ELSE INIT FOR RK06 TO TEST NEXT DRV
3144 004102 000730          BR 8$
3145
3146 004104 032737 010000 001550 10$: BIT #NED,T.CS2 ;WAS CERR DUE TO NED?
3147 004112 001364          BNE 9$        ;YES - BUMP TO NEXT DRIVE
3148 004114 032737 000040 001554  BIT #DTYE,T.ER ;ELSE SEE IF DRIVE TYPE ERR
3149 004122 001404          BEQ 24$       ;BR IF NO
3150
3151 004124 012737 002000 001720  MOV #CDT,DTYP ;ELSE SET UP FOR RK07
3152 004132 000714          BR 8$        ;AND TRY AGAIN
3153
3154 004134 104421          24$: TCHKOP          ;ELSE REPORT THE ERRORS
3155 004136 104004          ERROR 4 ;GR5,6,7
3156 004140 000000          .WORD 0
3157 004142 032737 004000 001664 11$: BIT #DRVPT,OPTFLG ;TEST IF SHOULD REPORT
3158 004150 001037          BNE 16$       ;NO - SKIP
3159
3160 004152 005713          TST (R3)      ;ANY DRIVE AVAILABLE?
3161 004154 001004          BNE 12$       ;BR IF NOT ZERO
3162 004156 104401 051227          TYPE ,OPR008 ;ELSE REPORT NO DRIVES AVAILABLE
3163 004162 000137 043724          JMP CTRHLT    ;GO TO CONTROLLED HALT
3164
3165 004166 012701 000200          12$: MOV #BIT7,R1  ;SET DRIVE SELECTOR FOR DRIVE 7
3166 004172 012700 000007          MOV #7,R0     ;SET DRIVE NUMBER TO 7
3167 004176 104401 051312          TYPE ,OPR009 ;TYPE PREFIX TO DRIVE TEST MESSAGE
3168
3169 004202 030113          13$: BIT R1,(R3)  ;TEST IF THIS DRIVE TO BE TESTED
3170 004204 001007          BNE 15$       ;YES - SKIP
3171
3172 004206 005300          14$: DEC R0     ;ELSE DECREMENT DRIVE NUMBER
3173 004210 006201          ASL R1        ;SHIFT BIT SELECTOR TO NEXT DRIVE DOWN
3174 004212 001373          BNE 13$       ;IF NOT SHIFTED OUT - LOOP

```

I 5
PAGE 61

CZR6KFO RK6 FCTNL CTRL DIAG MACY11 30A(1052) 19-JUN-80 14:14
 CZR6KF.P11 19-JUN-80 14:04 T3 SELECT ALL DRIVES SEQ 0060

```

3175 004214 052737 004000 001664      BIS      #DRV RPT,OPTFLG  ;SET DRIVE #'S REPORTED FLAG
3176 004222 000412                BR      16$        ;ELSE GO TO STATUS VALID TEST
3177
3178 004224 010037 004140      15$:  MOV      R0,101$    ;PUT DRIVE NUMBER IN TYPE LOCATION
3179 004230 052737 000060 004140      BIS      #BIT4!BIT5,101$ ;MAKE IT ASCIZ
3180 004236 104401                TYPE    101$        ;TYPE IT
3181 004240 004140                TYPE    ,SPACE2    ;TYPE SOME SPACES
3182 004242 104401 050763      BR      14$        ;LOOP
3183 004246 000757
3184
3185 004250 005000      16$:  CLR      R0        ;CLEAR DRIVE NUMBER COUNTER
3186 004252 012701 000001      MOV      #1,R1      ;SET DRIVE SELECTOR TO DRIVE 0
3187 004256 012737 177777 001240      MOV      #-1,$TMP7  ;SET $TMP7 NEGATIVE
3188 004264 012737 177777 001630      MOV      #-1,DRVBIT ;SET DRIVE SELECT BIT NEGATIVE
3189 004272 012737 004400 001264      MOV      #18$,$ESCAPE ;SET ESCAPE IN CASE OF ERR
3190 004300 030137 001666      BIT      R1,DRVDRP  ;HAS DRIVE 0 BEEN DROPPED?
3191 004304 001035      BNE     18$        ;YES - SKIP TO DRIVE INC.
3192
3193 004306 104416      17$:  TSSINIT ;CLEAR SUBSYSTEM
3194 004310 104003      ERROR  3          ;ERROR FOR BAD CLEAR
3195
3196 004312 010037 001610      MOV      R0,L.CS2   ;SET DRIVE SELECT
3197 004316 010037 001626      MOV      R0,DRVNUM  ;SET DRIVE NUMBER
3198 004322 004737 036134      JSR      PC,GTYP0   ;GET DRV TYPE
3199 004326 012737 000101 001600      MOV      #SELDRV,L.CS1 ;SET FOR DRIVE SELECT
3200
3201 004334 104417      TLOADRK ;LOAD RK REGS
3202 004336 104423      TWAT16  ;WAIT FOR INTERRUPT
3203 004340 104002      ERROR  2          ;ERROR TO SLOW/NOT COMPLETE
3204 004342 030113      BIT      R1,(R3)    ;WAS THAT DRIVE AVAILABLE
3205 004344 001026      BNE     19$        ;YES - SKIP
3206
3207 004346 104422      TCHKWE  ;CHECK THAT ERROR OCCURRED
3208 004350 000000      .WORD   0          ;NONE OF GROUP 1
3209 004352 000000      .WORD   0          ;NONE OF GROUP 2
3210 004354 000001      .WORD   1          ;NED IN GROUP 3
3211 004356 104004      ERROR  4 ;OR5,6,7 ;ERROR IF NO ERROR OR WRONG ERROR
3212
3213 004360 032737 100000 001552      BIT      #SVAL,T.DS ;DID STATUS VALID RESET?
3214 004366 001404      BEQ     18$        ;YES - SKIP
3215 004370 012737 055437 001400      MOV      #EM47,EM3N ;ELSE MESSAGE (SVAL NOT RESET WITH NED)
3216 004376 104003      ERROR  3
3217 004400 005200      18$:  INC      R0        ;BUMP TO NEXT DRIVE
3218 004402 006301      ASL     R1        ;SHIFT DRIVE SELECT BIT
3219 004404 032701 000400      BIT      #BIT8,R1   ;ALL DRIVES CHECKED
3220 004410 001063      BNE     21$        ;YES - EXIT
3221 004412 030137 001666      BIT      R1,DRVDRP  ;TEST IF DRIVE HAS BEEN DROPPED
3222 004416 001370      BNE     18$        ;YES - GET NEXT DRIVE
3223 004420 000732      BR      17$        ;ELSE CHECK THIS DRIVE
3224 004422 104421      19$:  TCHKOP  ;CHECK NO ERRORS SET
3225 004424 104004      ERROR  4 ;OR5,6,7 ;REPORT ALL ERRORS
3226
3227 004426 032737 100000 001552      BIT      #SVAL,T.DS ;CHECK SVAL SET
3228 004434 001004      BNE     20$        ;YES - SKIP
3229 004436 012737 055520 001400      MOV      #EM48,EM3N ;MESSAGE (NO SVAL FROM EXISTANT DR)
3230 004444 104003      ERROR  3

```

```
3231
3232 004446 012737 000103 001600 20$: MOV #PACK,L.CS1 ;ELSE SET TO DO PACK ACK
3233 004454 104417 TLOADRK ;LOAD RK
3234
3235 004456 104423 TWAT16 ;WAIT FOR INTERRUPT
3236 004460 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3237
3238 004462 104421 TCHKOP ;CHECK FOR ANY ERRORS
3239 004464 104004 ERROR 4 ;OR5,6,7 ;YES - REPORT & ABORT TEST
3240
3241 004466 032737 000100 001552 BIT #VV,T.DS ;DID VV SET
3242 004474 001005 BNE 22$ ;YES - SKIP
3243 004476 012737 055402 001400 MOV #EM46,EM3N ;MESSAGE (VV DID NOT SET/
3244 004504 104003 ERROR 3
3245 004506 000734 BR 18$
3246
3247 004510 032737 040000 001552 22$: BIT #DSC,T.DS ;TEST IF DSC RESET
3248 004516 001410 BEQ 23$ ;YES - SKIP
3249 004520 012737 056514 001470 MOV #EMDSC,EM12N
3250 004526 012737 056754 061022 MOV #EMSCLR,DF011A
3251 004534 104003 ERROR 3 ;'DSC NOT RESET RESULT OF SUBSYS CLEAR''
3252 004536 000720 BR 18$
3253
3254 004540 005737 001630 23$: TST DRVBIT ;TEST IF DRVBIT IS NEGATIVE
3255 004544 100315 BPL 18$ ;NO - SKIP
3256 004546 010137 001630 MOV R1,DRVBIT ;STORE DRIVE SELECT BIT
3257 004552 010037 001240 MOV R0,$TMP7 ;STORE DRIVE NUMBER TO BE TESTED
3258 004556 000710 BR 18$
3259
3260 004560 21$: CLR $ESCAPE ;REV 006
3261 004560 005037 001264 MOV $TMP7,DRVNUM ;LOAD LOWEST # DRIVE PRESENT INTO DRVNUM
3262 004564 013737 001240 001626 MOV DRVNUM,R1
3263 004572 013701 001626 MOV DRVNUM,R1
3264 004576 004737 036150 JSR PC,GTYP1 ;GET DRV TYP
3265
3266 004602 023727 001624 000002 CMP SRTFLG,#2 ;TEST IF CLOCK ADJUST START
3267 004610 001002 BNE 25$ ;NO - SKIP
3268 004612 000137 043614 JMP ADJCLK ;GO TO ADJUST CLOCK ROUTINE
3269
3270 004616 25$:
3271
3272 *****
3273 ;*TEST 4 RELEASE ALL DRIVES
3274 ;*
3275 ;* RELEASE ALL DRIVES. MAKE SURE NO ERROR
3276 ;* SETS AND STATUS VALID IS RESET.
3277 ;*
3278 *****
3279 TST4: SCOPE
3280 004616 000004 MOV #50, $TIMES ;:DO 50. ITERATIONS
3281 004620 012737 000062 001262 TSSINIT ;:INITIALIZE SUBSYSTEM
3282 004626 104416 ERROR 3 ;:BAD INIT
3283 004630 104003
3284 004632 013737 001626 001610 MOV DRVNUM,L.CS2 ;:SET DRIVE NUMBER
3285 004640 012737 000101 001600 MOV #SELDRV,L.CS1 ;:SET DRIVE SELECT
3286
```

```
3287 004646 104417 TLOADRK ;LOAD RK REGS
3288 004650 104423 TWAT16 ;WAIT FOR INTERRUPT
3289 004652 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3290
3291 004654 104421 TCHKOP ;CHECK FOR ANY ERRORS
3292 004656 104004 ERROR 4 ;OR5,6,7 ;REPORT ANY ERRORS
3293
3294 004660 012737 000010 001610 MOV #RLS,L.CS2 ;SET DRIVE RELEASE,STILL SET FOR SELECT
3295
3296 004666 104417 TLOADRK ;LOAD RK REGS
3297 004670 104423 TWAT16 ;WAIT FOR INTERRUPT
3298 004672 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3299
3300 004674 104421 TCHKOP ;CHECK FOR ANY ERRORS
3301 004676 104004 ERROR 4 ;OR 5, 6, OR 7 ;REPORT ALL ERRORS
3302 004700 032737 100000 001552 BIT #SVAL,T.DS ;DID SVAL RESET?
3303 004706 001404 BEQ 1$ ;YES - SKIP
3304 004710 012737 055577 001400 MOV #EM49,EM3N ;MESSAGE (SVAL NOT RESET W/RELEASE)
3305 004716 104003 ERROR 3
3306 004720
3307
3308 004720 004737 036034 TSTLUP: JSR PC,LDCON ;LOAD RK06-RK07 CONSTANTS
3309
3310 ;*****
3311 ;*TEST 5 NON-STANDARD MESSAGES AND SVAL
3312 ;*
3313 ;* PICK ONE OF THE AVAILABLE DRIVES AND GET
3314 ;* NON-STANDARD MESSAGES. MAKE SURE NO
3315 ;* ERROR OCCURS AND STATUS VALID DOES NOT SET
3316 ;* AND THAT NON-STANDARD MESSAGES CAUSE STATUS
3317 ;* VALID TO RESET.
3318 ;*****
3319 004724 000004 TST5: SCOPE
3320 004726 012737 000062 001262 MOV #50.,$TIMES ;;DO 50. ITERATIONS
3321 004734 104416 TSSINIT ;CLEAR SUBSYSTEM
3322 004736 104003 ERROR 3 ;BAD CLEAR MESSAGE
3323 004740 012701 000001 MOV #1,R1 ;PRESET R1 FOR MESSAGE PAIR 1
3324 004744 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRV NUMBER
3325 004752 012737 000101 001600 MOV #SELDRV,L.CS1 ;LOAD SELECT COMMAND
3326 004760 005037 001616 1$: CLR L.MR1 ;LOAD FOR STANDARD STATUS
3327 004764 104417 TLOADRK ;LOAD RK
3328 004766 104423 TWAT16 ;WAIT FOR INTERRUPT
3329 004770 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
3330 004772 104421 TCHKOP ;CHECK OPERATION
3331 004774 104004 ERROR 4 ;5,6 OR 7 ;REPORT ALL ERRORS
3332
3333 004776 032737 100000 001552 BIT #SVAL,T.DS ;TEST STATUS VALID SET
3334 005004 001007 BNE 2$ ;YES-SKIP
3335
3336 005006 012737 056354 001460 MOV #EMSVAL,EM11N
3337 005014 012737 050160 061022 MOV #EMSELD,DF011A
3338 005022 104011 ERROR 11 ;"SVAL NOT SET RESULT OF DRIVE SELECT"
3339
3340 005024 010137 001616 2$: MOV R1,L.MR1 ;LOAD MESSAGE PAIR SELECT
3341
3342 005030 104417 TLOADRK ;LOAD RK
```



```
3343 005032 104423          TWAT16          ;WAIT FOR INTERRUPT
3344 005034 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
3345
3346 005036 104421          TCHKOP          ;CHECK OPERATION
3347 005040 104004          ERROR 4 ;5,6, OR 7 ;REPORT ALL ERRORS
3348
3349 005042 032737 100000 001552 BIT #SVAL,T.DS ;TEST STATUS VALID RESET
3350 005050 001407          BEQ 3$          ;YES-SKIP
3351
3352 005052 012737 056354 001470 MOV #EMSVAL,EM12N
3353 005060 012737 056371 061022 MOV #EMNZPR,DF011A
3354 005066 104012          ERROR 12        ;'SVAL NOT RESET RESULT OF SEL W' NON-O PAIR''
3355
3356 005070 022701 000003 3$: CMP #3,R1      ;WAS PAIR 3 SELECTED?
3357 005074 001402          BEQ 4$          ;YES-SKIP
3358 005076 005201          INC R1          ;BUMP TO NEXT PAIR
3359 005100 000727          BR 1$          ;SKIP TO DO IT.
3360 005102
3361
3362
3363
3364
3365
3366
3367
3368
3369 005102 000004          TST6: SCOPE
3370 005104 012737 000062 001262 MOV #50.,$TIMES ;:DO 50. ITERATIONS
3371 005112 104416          TSSINIT        ;CLEAR SUBSYSTEM
3372 005114 104003          ERROR 3        ;BAD INIT ERROR
3373
3374 005116 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3375 005124 012737 000101 001600 MOV #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3376
3377 005132 104417          TLOADRK        ;LOAD RK
3378 005134 104423          TWAT16          ;WAIT FOR INTERRUPT
3379 005136 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
3380
3381 005140 104421          TCHKOP          ;CHECK OPERATION
3382 005142 104004          ERROR 4 ;5,6, OR 7 ;REPORT ALL ERRORS
3383
3384 005144 032737 100000 001552 BIT #SVAL,T.DS ;TEST STATUS VALID SET
3385 005152 001007          BNE 1$          ;YES-SKIP
3386
3387 005154 012737 056354 001460 MOV #EMSVAL,EM11N
3388 005162 012737 050160 061022 MOV #EMSELD,DF011A
3389 005170 104011          ERROR 11        ;'SVAL NOT SET RESULT OF DRV SELECT''
3390
3391 005172 013762 001626 000010 1$: MOV DRVNUM,RKCS2(R2) ;WRITE CS2 TO RESET SVAL
3392
3393 005200 104420          TGETRK         ;GET RK REGS.
3394
3395 005202 032737 100000 001552 BIT #SVAL,T.DS ;TEST SVAL RESET
3396 005210 001407          BEQ 2$          ;YES-SKIP
3397
3398 005212 012737 056354 001470 MOV #EMSVAL,EM12N
```

```
3399 005220 012737 056432 061022      MOV    #EMWCS2,DF011A
3400 005226 104012                      ERROR  12      ;'SVAL NOT RESET BY WRITING CS2''
3401 005230                               2$:
3402
3403      .SBTTL  **CONTROLLER ERROR TESTS
3404
3405      ;*****
3406      ;*TEST 7      DRIVE TYPE ERROR
3407      ;*
3408      ;*      CREATE A DRIVE TYPE ERROR.  MAKE SURE DRIVE
3409      ;*      TYPE ERROR SETS AND STATUS VALID SETS.
3410      ;*
3411      ;*****
3412 005230 000004      TST7:  SCOPE
3413 005232 012737 000062 001262      MOV    #50.,$TIMES      ;;DO 50. ITERATIONS
3414 005240 104416      TSSINIT                      ;CLEAR SUBSYSTEM
3415 005242 104003      ERROR  3      ;BAD INIT ERROR
3416
3417 005244 013737 001626 001610      MOV    DRVNUM,L.CS2      ;LOAD DRIVE NUMBER
3418 005252 012737 000101 001600      MOV    #SELDLV,L.CS1     ;LOAD DRIVE SELECT
3419 005260 053737 036130 001600      BIS    HOLD2,L.CS1      ;LOAD DRIVE TYPE
3420
3421 005266 104417      TLOADRK                      ;LOAD RK
3422 005270 104423      TWAT16                      ;WAIT FOR INTERRUPT
3423 005272 104002      ERROR  2      ;TO SLOW/NOT COMPLETE ERROR
3424
3425 005274 104422      TCHKWE                      ;CHECK OPERATION WITH EXPECTED ERROR
3426 005276 000040      .WORD 000040           ;DRIVE TYPE ERROR
3427 005300 000000      .WORD 0
3428 005302 000000      .WORD 0
3429 005304 104004      ERROR  4 ; OR 5,6,7    ;REPORT ANY DIFFERENCES (NO ERRORS,
3430                                     ;ADDITIONAL ERRORS, DIFFERENT ERRORS)
3431 005306 032737 100000 001552      BIT    #SVAL, T.DS      ;TEST IF SVAL SET
3432 005314 001007      BNE    1$              ;YES-SKIP
3433
3434 005316 012737 056354 001460      MOV    #EMSVAL,EM11N
3435 005324 012737 055050 061022      MOV    #EMDTPE,DF011A
3436 005332 104011      ERROR  11      ;'SVAL NOT SET RESULT OF DRV TYPE ERR''
3437 005334                               1$:
3438      ;*****
3439      ;*TEST 10     STATUS VALID AND PARITY ERROR
3440      ;*
3441      ;*      ISSUE A SELECT TO AN AVAILIABLE DRIVE WITH BAD PARITY.
3442      ;*      MAKE SURE SPAR, CONTROLLER ERROR, ATTENTION,
3443      ;*      DRIVE STATUS CHANGES, DRPAR, DRIVE INTERRUPT,
3444      ;*      AND STATUS VALID SET, ISSUE A CONTROLLER
3445      ;*      CLEAR.  MAKE SURE DRIVE INTERRUPT AND ATTENTION
3446      ;*      ARE STILL SET.  SELECT DRIVE AGAIN WITH GOOD
3447      ;*      PARITY.  MAKE SURE ATTENTION, DRIVE STATUS
3448      ;*      CHANGE, DRPAR, CONTROLLER ERROR, DRIVE INTERRUPT,
3449      ;*      AND STATUS VALID ARE SET AND SPAR IS RESET.
3450      ;*      ISSUE A CONTROLLER CLEAR.  GET NON-STANDARD MESSAGES
3451      ;*      AND MAKE SURE ONLY DRIVE INTERRUPT AND ATTENTION
3452      ;*      ARE SET.  CLEAR ATTENTION WITH DRIVE CLEAR.  REPEAT
3453      ;*      FOR ALL AVAILIABLE DRIVES.
3454      ;*
```

```
*****  
3455  
3456 005334 000004  
3457 005336 012737 000062 001262  
3458 005344 104416  
3459 005346 104003  
3460  
3461 005350 013737 001626 001610  
3462 005356 012737 000101 001600  
3463 005364 012737 000020 001616  
3464  
3465 005372 104417  
3466 005374 104423  
3467 005376 104002  
3468  
3469 005400 104422  
3470 005402 000011  
3471 005404 000000  
3472 005406 000000  
3473 005410 104004  
3474  
3475 005412 012700 000400  
3476  
3477 005416 013701 001626  
3478 005422 001403  
3479 005424 006300  
3480 005426 005301  
3481 005430 001375  
3482  
3483 005432 030037 001556  
3484 005436 001007  
3485 005440 012737 056540 001460  
3486 005446 012737 054765 061022  
3487 005454 104011  
3488 005456 032737 040000 001540  
3489 005464 001007  
3490 005466 012737 056474 001460  
3491 005474 012737 054765 061022  
3492 005502 104011  
3493  
3494 005504 032737 040000 001552  
3495 005512 001007  
3496 005514 012737 056514 001460  
3497 005522 012737 054765 061022  
3498 005530 104011  
3499  
3500 005532 032737 100000 001552  
3501 005540 001007  
3502 005542 012737 056354 001460  
3503 005550 012737 054765 061022  
3504  
3505 005556 104011  
3506  
3507 005560 005037 001616  
3508  
3509 005564 052737 100000 001600  
3510 005572 104417
```

IST10: SCOPE
MOV #50.,\$TIMES ;DO 50. ITERATIONS
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
MOV #SELDIV,L.CS1 ;LOAD DRIVE SELECT
MOV #PAT,L.MR1 ;LOAD EVEN PARITY BIT
TLOADRK ;LOAD RK REGS-SELECT W/EVEN PARITY
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
TCHKWE ;CHECK OPERATION FOR EXPECTED ERROR
DRPARERR!SPARERR ;DRIVE SELECTED DRIVE BUS PARITY ERROR
.WORD 0 ;CONTROLLER DETECTED DRIVE BUS PARITY ERROR
.WORD 0
ERROR 4 ; OR 5,6,7 ;REPORT ANY DIFFERENCES
MOV #BIT8,R0 ;ROUTINE TO DETERMINE WHICH BIT
MOV DRVNUM,R1 ;SHOULD BE SET IN ASOF TO INDICATE
BEQ 3\$;DRIVE ATTENTION. R0 WILL HAVE THE
2\$: ASL R0 ;BIT THAT SHOULD BE SET FOR THE DRIVE
DEC R1 ;IN USE
BNE 2\$
3\$: BIT R0,T.ASOF ;TEST ATTENTION SET
BNE 4\$;YES-SKIP
MOV #EMDA,EM11N
MOV #EMDPAR,DF011A
ERROR 11 ;'DRV ATT NOT SET RESULT OF DRV PARITY ERR'
4\$: BIT #DI,T.CS1 ;TEST DRIVE INTERRUPT SET
BNE 5\$;YES-SKIP
MOV #EMDI,EM11N
MOV #EMDPAR,DF011A
ERROR 11 ;'DRV INT NOT SET RESULT OF DRV PARITY ERR'
5\$: BIT #DSC,T.DS ;TEST DRIVE STATUS CHANGE SET
BNE 6\$;YES-SKIP
MOV #EMDSC,EM11N
MOV #EMDPAR,DF011A
ERROR 11 ;'DSC NOT SET RESULT OF DRV PARITY ERR'
6\$: BIT #SVAL,T.DS ;TEST STATUS VALID SET
BNE 7\$;YES-SKIP
MOV #EMSVL,EM11N
MOV #EMDPAR,DF011A
ERROR 11 ;'SVAL NOT SET RESULT OF DRV PAR ERR'
7\$: CLR L.MR1 ;CLEAR PAT IN MR1
BIS #CCLR,L.CS1 ;CLEAR CONTROLLER
TLOADRK ;LOAD RK REGS TO DO CLEAR


```

3567
3568 006036 012701 000001          MOV    #1,R1          ;SET COUNTER TO SELECT STATUS PAIR
3569 006042 012737 000101 001600  MOV    #SELDRV,L.CS1 ;LOAD DRIVE SELECT
3570
3571 006050 010137 001616          14$:  MOV    R1,L.MR1      ;LOAD STATUS PAIR SELECTION
3572 006054 104417                   TLOADRK ;LOAD RK REGS
3573 006056 104423                   TWAT16  ;WAIT FOR INTERRUPT
3574 006060 104002                   ERROR   2           ;TO SLOW/NOT COMPLETE ERROR
3575
3576 006062 104421                   TCHKOP ;CHECK IF ANY ERRORS SET
3577 006064 104004                   ERROR   4 ; OR 5,6,7 ;REPORT ALL ERRORS SET.
3578
3579 006066 030037 001556          BIT    R0,T.ASOF     ;TEST ATTENTION STILL SET
3580 006072 001007                   BNE    15$          ;YES-SKIP
3581 006074 012737 056540 001410  MOV    #EMDA,EM4N
3582 006102 012737 056371 061022  MOV    #EMNZPR,DF011A
3583 006110 104014                   ERROR   14          ;"ATTENTION RESET RESULT OF NON-0 PAIR SEL"
3584
3585 006112 032737 040000 001540  15$:  BIT    #DI,T.CS1
3586 006120 001007                   BNE    16$
3587 006122 012737 056474 001510  MOV    #EMDI,EM14N
3588 006130 012737 056371 061022  MOV    #EMNZPR,DF011A
3589 006136 104014                   ERROR   14          ;"DRV INT RESET RESULT OF NON-0 PAIR SELECT"
3590
3591 006140 005201          16$:  INC    R1           ;BUMP PAIR SELECT
3592 006142 022701 000004          CMP    #4,R1        ;ALL PAIRS DONE?
3593 006146 001340          BNE    14$          ;NO-LOOP
3594
3595 006150 005037 001616          CLR    L.MR1        ;CLEAR MR1
3596
3597 006154 012737 000105 001600  MOV    #CLEAR,L.CS1 ;LOAD DRIVE CLEAR
3598
3599 006162 104417                   TLOADRK ;DO DRIVE CLEAR
3600 006164 104423                   TWAT16  ;WAIT FOR INTERRUPT
3601 006166 104002                   ERROR   2           ;TO SLOW/NOT COMPLETE ERROR
3602
3603 006170 104421                   TCHKOP ;CHECK FOR ANY ERRORS
3604 006172 104004                   ERROR   4 ; OR 5,6,7 ;REPORT ALL ERRORS
3605
3606 006174 012701 000020          17$:  MOV    #20,R1       ;SET COUNT FOR SHORT WAIT
3607 006200 005301                   DEC    R1           ;TO ALLOW CONTROLLER TIME TO POLL
3608 006202 001376                   BNE    17$          ;DRIVES
3609
3610 006204 104420                   TGETRK ;GET RK REGS
3611 006206 030037 001556          BIT    R0,T.ASOF     ;TEST ATTENTION RESET
3612 006212 001407                   BEQ    18$          ;YES-SKIP
3613 006214 012737 056540 001470  MOV    #EMDA,EM12N
3614 006222 012737 050206 061022  MOV    #EMDCLR,DF011A
3615 006230 104012                   ERROR   12          ;"ATTENTION NOT RESET RESULT OF DRV CLEAR
3616
3617 006232 032737 040000 001540  18$:  BIT    #DI,T.CS1
3618 006240 001407                   BEQ    19$          ;TEST DRIVE INTERRUPT RESET
3619 006242 012737 056474 001470  MOV    #EMDI,EM12N ;YES-SKIP
3620 006250 012737 050206 061022  MOV    #EMDCLR,DF011A
3621 006256 104012                   ERROR   12          ;"DRV INT NOT RESET RESULT OF DRIVE CLR"
3622

```

3623 006260

19\$:

*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.

3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634 006260 000004
3635 006262 012737 000062 001262
3636 006270 104416
3637 006272 104002
3638
3639 006274 013737 001626 001610
3640 006302 012737 000101 001600
3641
3642 006310 104417
3643 006312 104423
3644 006314 104002
3645
3646 006316 104421
3647 006320 104004
3648
3649 006322 052737 000010 001610
3650 006330 012737 000040 001616
3651
3652 006336 104417
3653
3654 006340 004437 036316
3655 006344 023
3656 006345 002
3657
3658 006346 042762 000040 000026
3659
3660 006354 104423
3661 006356 104002
3662
3663 006360 104422
3664 006362 000000
3665 006364 000000
3666 006366 000004
3667 006370 104004
3668
3669 006372 104416
3670 006374 104002

TST11: SCOPE
MOV #50.,\$TIMES ;;DO 50. ITERATIONS
TSSINIT ;CLEAR SUBSYSTEM
ERROR 2 ;BAD INIT ERROR

MOV DRVNUM,L.CS2 ;SELECT A DRIVE
MOV #SELDRV,L.CS1 ;DO DRIVE SELECT

TLOADRK ;LOAD RK
TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

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

BIS #RLS,L.CS2 ;LOAD RELEASE
MOV #DMD,L.MR1 ;SET DIAGNOSTIC MODE

TLOADRK ;LOAD RK

JSR R4,MCLOCK ;CALL MAINT CLOCK
.BYTE ^D19 ;NUMBER OF PHASES
.BYTE 2 ;NUMBER OF CLOCK XISTIONS

BIC #DMD,RKMR1(R2) ;CLEAR DIAG MODE

TWAT16 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETED

TCHKWE ;CHECK OPERATION WITH ERROR
.WORD 0
.WORD 0
.WORD UFERR ;UNIT FIELD ERROR
ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES

TSSINIT ;CLEAR SUBSYSTEM TO INSURE UFE RESETS
ERROR 2

3671
3672
3673
3674
3675
3676
3677
3678

*TEST 12 UNIT FIELD ERROR ON SELECT
*
* ISSUE A SUBSYSTEM CLEAR. SELECT AN AVAILABLE
* DRIVE. 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

```
3679
3680
3681
3682 006376 000004
3683 006400 012737 000062 001262
3684 006406 104416
3685 006410 104003
3686
3687 006412 013737 001626 001610
3688 006420 012737 000101 001600
3689
3690 006426 104417
3691 006430 104423
3692 006432 104002
3693
3694 006434 104421
3695 006436 104004
3696
3697 006440 012737 000043 001616
3698 006446 005037 001610
3699
3700 006452 104417
3701
3702 006454 004437 036316
3703 006460 026
3704 006461 002
3705
3706 006462 042762 000040 000026
3707
3708 006470 104423
3709 006472 104002
3710
3711 006474 013737 036132 006504
3712
3713 006502 104422
3714 006504 000000
3715 006506 000000
3716 006510 000004
3717 006512 104004
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734 006514 000004
```

TST12: SCOPE ;DO 50. ITERATIONS
MOV #50.,\$TIMES ;CLEAR SUBSYSTEM
TSSINIT ;BAD INIT ERROR
ERROR 3 ;LOAD DRIVE NUMBER
MOV DRVNUM,L.CS2 ;LOAD DRIVE SELECT
MOV #SELDRV,L.CS1 ;LOAD RK
TLOADRK ;WAIT FOR INTERRUPT
TWAT16 ;TO SLOW/NOT COMPLETE
ERROR 2 ;CHECK FOR ANY ERROR
TCHKOP ;REPORT ALL ERRORS
ERROR 4 ; OR 5,6,7 ;LOAD DIAG MODE & MSG PAIR 3
MOV #DMD!BIT1!BIT0,L.MR1 ;LOAD FOR DRIVE 0
CLR L.CS2 ;LOAD RK
TLOADRK ;CALL MAINTENANCE CLOCK
JSR R4,MCLOCK ;THROUGH PHASE 6
.BYTE ^D22 ;PLUS 2 TRANSITIONS
.BYTE 2 ;CLEAR DIAG MODE
BIC #DMD,RKMR1(R2) ;WAIT FOR INTERRUPT
TWAT16 ;TO SLOW/NOT COMPLETED ERROR
ERROR 2 ;EXPECTED ERROR
MOV HOLD3,1\$;CHECK OPERATION WITH ERROR
TCHKWE ;UNIT FIELD ERROR SHOULD SET
1\$: .WORD 0 ;REPORT ANY DISCREPENCIES
.WORD 0
.WORD UFERR
ERROR 4 ; OR 5,6,7

.SBTTL **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)

TST13: SCOPE

3735	006516	012737	000062	001262	MOV	#50.,\$TIMES	::DO 50. ITERATIONS
3736	006524	104416			TSSINIT		:CLEAR SUBSYSTEM
3737	006526	104003			ERROR	3	:BAD INIT ERROR
3738							
3739	006530	013737	001626	001610	MOV	DRVNUM,L.CS2	:LOAD DRIVE NUMBER
3740	006536	012737	000113	001600	MOV	#RECAL,L.CS1	:LOAD RECAL
3741							
3742	006544	104417			TLOADRK		:LOAD RK
3743	006546	104423			TWAT16		:WAIT FOR 1ST INTERRUPT
3744	006550	104002			ERROR	2	:TO SLOW/NOT COMPLETE
3745	006552	005037	001662		CLR	INTSET	:CLEAR INTERRUPT FLAG
3746	006556	104420			TGETRK		:GET RK REGS
3747	006560	032737	100000	001552	BIT	#SVAL,T.DS	:TEST SVAL SET
3748	006566	001010			BNE	1\$:YES-SKIP
3749	006570	012737	056354	001460	MOV	#EMSVAL,EM11N	
3750	006576	012737	050247	061022	MOV	#EMRCAL,DF011A	
3751	006604	104011			ERROR	11	: 'SVAL NOT SET RESULT OF RECAL'
3752	006606	000463			BR	50\$:ABORT TEST
3753							
3754	006610	104437			1\$: TWAT8S		:WAIT FOR INTERRUPT
3755	006612	000401			BR	2\$:NO INTERRUPT RETURN
3756	006614	000404			BR	3\$:INTERRUPT RETURN
3757							
3758	006616	012737	055647	001370	2\$: MOV	#EM50,EM2N	:ALTER MESSAGE 'NO 2ND INTERRUPT OR IT WAS LATE'
3759	006624	104002			ERROR	2	
3760							
3761	006626	104420			3\$: TGETRK		:GET RK REGS
3762	006630	032737	100000	001552	BIT	#SVAL,T.DS	:TEST SVAC SET NOW
3763	006636	001410			BEQ	4\$:NO-SKIP
3764	006640	012737	056354	001470	MOV	#EMSVAL,EM12N	
3765	006646	012737	056601	061022	MOV	#EM2INT,DF011A	
3766	006654	104012			ERROR	12	: 'SVAL NOT RESET RESULT OF SECOND TEST'
3767	006656	000437			BR	50\$	
3768							
3769	006660	032737	040000	001552	4\$: BIT	#DSC,T.DS	:TEST DSC SET BY ATTENTION
3770	006666	001010			BNE	5\$:YES-SKIP
3771	006670	012737	056514	001460	MOV	#EMDSC,EM11N	
3772	006676	012737	056601	061022	MOV	#EM2INT,DF011A	
3773	006704	104011			ERROR	11	: 'DSC NOT SET RESULT OF SECOND INTERRUPT'
3774	006706	000423			BR	50\$	
3775							
3776	006710	012737	000101	001600	5\$: MOV	#SELDRV,L.CS1	:LOAD DRIVE SELECT
3777							
3778	006716	104417			TLOADRK		:LOAD RK REGS
3779	006720	104423			TWAT16		:WAIT FOR INTERRUPT
3780	006722	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
3781							
3782	006724	104421			TCHKOP		:CHECK FOR ANY ERRORS
3783							
3784	006726	104004			ERROR	4 ; OR 5,6,7	:REPORT ALL ERRORS
3785							
3786	006730	032737	100000	001552	BIT	#SVAL,T.DS	:TEST SVAC SET
3787	006736	001007			BNE	50\$:YES-SKIP
3788	006740	012737	056354	001460	MOV	#EMSVAL,EM11N	
3789	006746	012737	050160	061022	MOV	#EMSELD,DF011A	
3790	006754	104011			ERROR	11	: 'SVAL NOT SET RESULT OF DRV SEL.

3791 006756

3792

3793

3794

3795

3796

3797

3798

3799

3800 006756 000004

3801 006760 012737 000062 001262

3802 006766 104416

3803 006770 104003

3804

3805 006772 013737 001626 001610

3806 007000 012737 000117 001600

3807

3808 007006 104417

3809 007010 104423

3810 007012 104002

3811

3812

3813 007014 104420

3814

3815 007016 032737 040000 001540

3816 007024 001010

3817 007026 012737 056474 001460

3818 007034 012737 056622 061022

3819 007042 104011

3820 007044 000417

3821

3822

3823 007046 012700 000031

3824 007052 005300

3825 007054 001376

3826

3827 007056 022737 000001 001662

3828 007064 001407

3829 007066 012737 056774 001500

3830 007074 012737 056622 061022

3831 007102 104013

3832

3833 007104 104421

3834 007106 104004

3835

3836

3837

3838

3839

3840

3841

3842 007110 000004

3843 007112 012737 000005 001262

3844 007120 104416

3845 007122 104003

3846

50\$:

*TEST 14 SINGLE INTERRUPT FROM ATTENTION

*

*

*

*

*

*

TST14: SCOPE ;DO 50. ITERATIONS

MOV #50.,\$TIMES ;CLEAR SUBSYSTEM

TSSINIT ;BAD INIT ERROR

ERROR 3

MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER

MOV #SEEK,L.CS1 ;LOAD SEEK DCYL LEFT AT 0.

TLOADRK ;LOAD RK REGS

TWAT16 ;WAIT FOR INTERRUPT

ERROR 2 ;TO SLOW/NOT COMPLETED ERROR

TGETRK ;GET RK REGS

BIT #DI,T.CS1 ;TEST DI SET

BNE 2\$;YES-SKIP

MOV #EMDI,EM11N

MOV #EMSKSF,DF011A

ERROR 11 ;'DI NOT SET RESULT OF SEEK TO SELF'

BR 50\$

2\$: MOV #25.,R0 ;LOAD AND DECREMENT A COUNT TO

3\$: DEC R0 ;ZERO. GIVE CONTROLLER A CHANCE TO

BNE 3\$;INTERRUPT AGAIN. ERROR IF IT DOES.

CMP #1,INTSET ;CHECK ONLY ONE INTERRUPT OCCURRED

SEQ 50\$;YES-SKIP

MOV #EMMI,EM13N

MOV #EMSKSF,DF011A

ERROR 13 ;'MULTIPLE INTERRUPTS RESULT OF SEEK TO SELF'

50\$: TCHKOP ;CHECK FOR ANY ERRORS

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

*TEST 15 RESET ATTENTIONS WITH UNIBUS INIT

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

```

3847 007124 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3848 007132 012737 000117 001600 MOV #SEEK,L.CS1 ;LOAD SEEK (TO SELF-0)
3849
3850 007140 104417 TLOADRK ;LOAD RK REGS
3851 007142 104423 TWAT16 ;WAIT FOR INTERRUPT
3852 007144 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
3853
3854 007146 104420 TGETRK ;GET RK REGS
3855
3856 007150 032737 040000 001540 BIT #DI,T.CS1 ;TEST DI SET
3857 007156 001010 BNE 1$ ;YES-EXIT
3858 007160 012737 056474 001460 MOV #EMDI,EM11N
3859 007166 012737 056622 061022 MOV #EMSKSF,DF011A
3860 007174 104011 ERROR 11 ;'DI NOT SET RESULT OF SEEK TO SELF
3861 007176 000450 BR 50$
3862
3863 007200 005037 001662 1$: CLR INTSET ;CLEAR INTERRUPT COUNTER
3864 007204 000005 RESET ;DO UNIBUS RESET
3865 007206 004737 045146 JSR PC,$TKINT ;RESET KEYBOARD INTERRUPT
3866
3867 007212 005037 001674 CLR LCLKTK ;CLEAR TICK COUNTER
3868 007216 004737 035314 JSR PC,MYTIME ;CALL TIMER
3869 007222 022737 000012 001674 CMP #10.,LCLKTK ;COUNT 10 TICKS (MILLISECONDS)?
3870 007230 001372 BNE 5$ ;NO - LOOP
3871
3872 007232 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET IE FOR ANY STRAY INTERRUPTS
3873 007240 004737 034522 JSR PC,OPTTST ;SET UP OPTIONS AGAIN
3874
3875 007244 104423 TWAT16 ;WAIT 16 MS FOR AN INTERRUPT
3876 007246 000410 BR 2$ ;NONE IS EXPECTED SO RETURN SHOULD BE
3877 ;HERE-BR TO CONTINUE TEST.
3878 007250 012737 056637 001500 MOV #EMUXIT,EM13N ;INT OCCURRED ON RESET
3879 007256 012737 056701 061022 MOV #EMRSET,DF011A
3880 007264 104013 ERROR 13 ;'UNEXECUTED INTERRUPT RESULT OF RESET'
3881 007266 000414 BR 50$
3882 007270 104420 2$: TGETRK ;GET RK REGS
3883 007272 032737 040000 001540 BIT #DI,T.CS1 ;TEST DI RESET
3884 007300 001407 BEQ 50$ ;YES-SKIP
3885 007302 012737 056474 001470 MOV #EMDI,EM12N
3886 007310 012737 056701 061022 MOV #EMRSET,DF011A
3887 007316 104012 ERROR 12 ;'DI NOT RESET RESULT OF RESET'
3888
3889 007320 50$:
3890
3891 .SBTTL **ILLEGAL DISK ADDRESS ERROR TESTS
3892
3893 ;*****
3894 ;*TEST 16 ILLEGAL DISK ADDRESS (PART 1)
3895 ;*
3896 ;* ISSUE A SEEK TO CYLINDER 0, HEAD 3. MAKE SURE
3897 ;* ILLEGAL ADDRESS ERROR AND SEEK INCOMPLETE SETS.
3898 ;* CLEAR CONTROLLER AND CLEAR DRIVE. REPEAT FOR HEADS 4-7,
3899 ;* CHECKING THAT BOTH IDAE AND SEEK INCOMPLETE SET FOR
3900 ;* HEAD 7 AND IDAE ALONE SETS FOR HEADS 4, 5, AND 6.
3901 ;*
3902 ;*****
  
```

```

3903 007320 000004
3904 007322 012737 000012 001262 TST16: SCOPE
3905 007330 012701 000003          MOV #10,$TIMES      ;;DO 10. ITERATIONS
3906                                     MOV #3,R1           ;;PRESET FOR SELECTING TRACK 3
3907 007334 104416          TSSINIT           ;CLEAR SUBSYSTEM
3908 007336 104003          ERROR 3
3909
3910 007340 012737 000113 001600 MOV #RECÄL,L.CS1  ;SET UP TO RECAL
3911 007346 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE
3912
3913 007354 104417          TLOADRK          ;LOAD RK REGS
3914
3915 007356 104423          TWAT16           ;WAIT FOR 1ST INTERRUPT
3916 007360 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
3917
3918 007362 005037 001662          CLR INTSET       ;CLEAR INTERRUPT FLAG
3919
3920 007366 104437          TWAT8S           ;WAIT FOR INTERRUPT
3921 007370 104002          ERROR 2
3922
3923 007372 012737 007400 001110 MOV #1$,$LPERR   ;SET LOCAL LOOP ON ERROR
3924
3925 007400 104416          1$: TSSINIT        ;CLEAR SUBSYSTEM
3926 007402 104003          ERROR 3          ;BAD INIT ERROR
3927
3928 007404 013737 001626 001610 MOV DRVNUM,L.CS2 ;LOAD DRIVE NUMBER
3929 007412 012737 000117 001600 MOV #SEEK,L.CS1  ;LOAD SEEK
3930 007420 110137 001607          MOV#B R1,L.DT    ;LOAD TRACK
3931
3932 007424 104417          TLOADRK          ;LOAD RK REGS
3933 007426 104423          TWAT16           ;WAIT FOR INTERRUPT
3934 007430 104002          ERROR 2          ;TO SLOW/NOT COMPLETE
3935
3936 007432 032701 000001          BIT #BIT0,R1     ;TEST IF HEAD ADDRESS HAS BIT 0
3937 007436 001403          BEQ 2$           ;NO - SKIP
3938 007440 032701 000002          BIT #BIT1,R1     ;TEST IF HEAD ADDRESS HAS BOTH 0 AND 1
3939 007444 001007          BNE 3$           ;YES-GO CHECK BOTH IDAE AND SKI SET
3940
3941 007446 104422          2$: TCHKWE        ;CHECK OPERATION WITH ERROR
3942 007450 002000          IDAERR          ;ILLEGAL DISK ADDRESS ERROR
3943 007452 000000          0
3944 007454 000000          0
3945 007456 104004          ERROR 4 ; OR 5,6,7 ;REPORT ALL DISCREPANCIES
3946 007460 104415          SCOP1          ;LOCAL LOOP ON ERROR
3947 007462 000406          BR 4$
3948
3949 007464 104422          3$: TCHKWE        ;CHECK OPERATION WITH ERROR
3950 007466 002002          IDAERR!SKIERR  ;ILLEGAL DISK ADDRESS ERROR
3951 007470 000000          0              ;SEEK INCOMPLETE
3952 007472 000000          0
3953 007474 104004          ERROR 4 ;OR 5,6,7 ;REPORT ANY DISCREPANCIES
3954 007476 104415          SCOP1          ;LOCAL LOOP ON ERROR TO 1$
3955
3956 007500 005201          4$: INC R1        ;ELSE BUMP TO NEXT ILLEGAL TRACK
3957 007502 022701 000010          CMP #8.,R1      ;ALL ILLEGAL TRACKS SELECTED?
3958 007506 001334          BNE 1$          ;NO-LOOP

```

```
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968 007510 000004
3969 007512 012737 000012 001262
3970 007520 104416
3971 007522 104003
3972
3973 007524 012737 000113 001600
3974 007532 013737 001626 001610
3975
3976 007540 104417
3977
3978 007542 104423
3979 007544 104002
3980
3981 007546 005037 001662
3982
3983 007552 104437
3984 007554 104002
3985
3986 007556 012737 007564 001110
3987
3988 007564 104416
3989 007566 104003
3990
3991 007570 013737 001626 001610
3992 007576 012737 000117 001600
3993 007604 005737 001720
3994 007610 001402
3995
3996
3997
3998 007612 000137 007644
3999
4000
4001
4002
4003 007616 012737 001000 001614
4004
4005
4006 007624 104417
4007 007626 104423
4008 007630 104002
4009
4010 007632 104422
4011 007634 002000
4012 007636 000000
4013 007640 000000
4014 007642 104004

*****
*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 FOR THE RK07 BECAUSE THE RK07
* CONTROLLER DOES NOT RECOGNIZE THE ILLEGAL DISK ADDRESS
*****
TST17: SCOPE
MOV #10.,$TIMES ;:DO 10. ITERATIONS
TSSINIT ;:CLEAR SUBSYSTEM
ERROR 3 ;:BAD INIT ERROR

MOV #RECAL,L.CS1 ;:LOAD RECALIBRATE
MOV DRVNUM,L.CS2 ;:LOAD DRIVE

TLOADRK ;:LOAD RK REGS

TWTAT16 ;:WAIT FOR 1ST INTERRUPT
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR

CLR INTSET ;:CLEAR INTERRUPT FLAG

TWTAT8S ;:WAIT FOR INTERRUPT
ERROR 2

MOV #1$,$LPERR ;:SET LOOP TO BYPASS RECAL

1$: TSSINIT ;:CLEAR SUBSYSTEM
ERROR 3

MOV DRVNUM, L.CS2 ;:LOAD DRIVE NUMBER
MOV #SEEK,L.CS1 ;:LOAD SEEK
TST DTYPE ;:SEE IF RK06
BEQ 2$ ;:BR IF YES
MOV #1777,L.DCYL ;:ELSE LOAD RK07 ILL CYL
MOV #<IDAERR!SKIERR>,4$ ;:EXP RK07 ERR
BR 3$
JMP TST20 ;:RK07 CONTROLLER
;:DOES NOT RECOGNIZE
;:ILLEGAL DISK ADDRESS
;:REV 006 THE ABOVE THREE LINES

2$: MOV #1000,L.DCYL ;:LOAD DK06 ILL CYL
MOV #IDAERR,4$ ;:EXP RK06 ERR

3$: TLOADRK ;:LOAD RK REGS
TWTAT16 ;:WAIT FOR INTERRUPT
ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR

4$: TCHKWE ;:CHECK OPERATION WITH ERROR
.WORD IDAERR ;:DISK ADDRESS ERROR
.WORD 0
.WORD 0
ERROR 4 ; OR 5,6,7 ;:REPORT ANY DISCREPANCIES
```

4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070

.SBTTL **WRITE HEADER TESTS

: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.

007644 000004
007646 012737 000001 001262

ST20: SCOPE
MOV #1,\$TIMES ;DO 1 ITERATION

TSSINIT ;CLEAR SUBSYS FROM LAST IDAERR
ERROR 3 ;BAD INIT ERROR

MOV #RECAL,L.CS1 ;DO RECAL CMD
MOV DRVNUM,L.CS2

TLOADRK ;LOAD 'L' REGS INTO RK REGS
TWAT16
ERROR 2 ;NO INTR

CLR INTSET ;CLR INIT FLAG

TWAT8S ;WAIT FOR 2'N INTR
ERROR 2 ;NOT THERE

CLRB 2\$;CLEAR SECTOR POINTER
CLR R0
CLR R5 ;CLEAR R5 FOR BAD SECTOR COUNTING
MOV BSF26P,R3 ;SET POINT IN TO STORE BS 26 SECT FORMAT
MOV #1\$,\$LPERR ;SET ERROR RETURN ADDRESS FOR INTERNAL LOOP
1\$: TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR

MOV LSTCYL,15\$

4071	007710	004437	035674			JSR	R4,LRLOAD		;LOAD "L" REGS WITH
4072	007714	000121				RDDATA			; READ DATA
4073	007716	177400				-400			; WORD COUNT
4074	007720	070414				IBUFF			; BUFFER ADDRESS
4075	007722	000			2\$:	.BYTE	0		; SECTOR ADDRESS
4076	007723	002				.BYTE	2		; TRACK ADDRESS
4077	007724	000000			15\$:		0		; CYLINDER ADDRESS 632/1456
4078									
4079	007726	104417				TLOADRK			;LOAD "L" REGS INTO RK
4080	007730	104431				TWAT112			;WAIT FOR INTERRUPT
4081	007732	104002				ERROR	2		;TO SLOW/NOT COMPLETE ERROR
4082	007734	104421				TCHKOP			;CHECK FOR ANY ERRORS
4083	007736	104004				ERROR	4 ; OR 5,6,7		;REPORT ALL ERRORS
4084	007740	104415				SCOP1			;LOOP TO 1\$ IF SW 9 SET
4085									
4086	007742	105737	001103			TSTB	\$ERFLG		;TEST FOR ERROR IN OPERATION
4087	007746	001502				BEQ	7\$;NO-SKIP
4088	007750	005700				TST	R0		;GETTING A BS FACTORY SECTOR 26 SECT FORMAT?
4089	007752	001023				BNE	3\$;NO-SKIP
4090	007754	062737	000002	007722		ADD	#2,2\$;NEXT SECTOR ADDRESS
4091	007762	122737	000012	007722		CMPB	#10.,2\$;PAST APPLICABLE SECTORS?
4092	007770	001066				BNE	6\$;NO-SKIP
4093	007772	012737	055746	001360		MOV	#EM51,EM1N		
4094	010000	104001				ERROR	1		;"CANNOT READ BS FILES"
4095	010002	043737	001630	001354	25\$:	BIC	DRVBIT,\$DEVM		;CLEAR DRIVE FROM DRIVE MAP
4096	010010	053737	001630	001666		BIS	DRVBIT,DRVDRP		;SET DRIVE DROPPED
4097	010016	000137	027716			JMP	NEWDRV		;ABORT TEST PASS.
4098									
4099	010022	022700	000001		3\$:	CMP	#1,R0		;GETTING A BS SOFT SECTOR 26 SECT FORMAT?
4100	010026	001014				BNE	4\$;NO-SKIP
4101	010030	062737	000002	007722		ADD	#2,2\$;NEXT SECTOR ADDRESS
4102	010036	122737	000026	007722		CMPB	#22.,2\$;PAST APPLICABLE SECTORS?
4103	010044	001040				BNE	6\$;NO-SKIP
4104	010046	012737	055746	001360		MOV	#EM51,EM1N		
4105	010054	104001				ERROR	1		;"CANNOT READ BS FILES"
4106	010056	000751				BR	25\$		
4107									
4108	010060	022700	000002		4\$:	CMP	#2,R0		;GETTING A BS FACT SECTOR 24 SECTOR FORMAT?
4109	010064	001014				BNE	5\$;NO-SKIP
4110	010066	062737	000002	007722		ADD	#2,2\$;NEXT SECTOR ADDRESS
4111	010074	122737	000013	007722		CMPB	#11.,2\$;PAST APPLICABLE SECTORS?
4112	010102	001021				BNE	6\$;NO-SKIP
4113									
4114	010104	012737	055746	001360		MOV	#EM51,EM1N		
4115	010112	104001				ERROR	1		;"CANNOT READ BS FILES"
4116	010114	000732				BR	25\$		
4117									
4118	010116	062737	000002	007722	5\$:	ADD	#2,2\$;NEXT SECTOR (BS SOFT 24 SECT MODE)
4119	010124	122737	000027	007722		CMPB	#23.,2\$;PAST APPLICABLE SECTORS?
4120	010132	001005				BNE	6\$;NO-SKIP
4121	010134	012737	055746	001360		MOV	#EM51,EM1N		
4122	010142	104001				ERROR	1		;"CANNOT READ BS FILES"
4123	010144	000716				BR	25\$		
4124									
4125	010146	105037	001103		6\$:	CLRB	\$ERFLG		;CLEAR ERROR FLAG
4126	010152	000651				BR	1\$;DO NEXT READ

```

4127
4128 010154 005737 070422      7$:  TST      Ibuff+6      ;CHECK FOR ALIGNMENT PACK
4129 010160 001405              BEQ      8$              ;NO-SKIP
4130 010162 012737 056041 001360  MOV      #EM52,EM1N
4131 010170 104001              ERROR    1              ;"ALIGNMENT PACK. DRIVE ABORTING"
4132 010172 000703              BR       25$
4133
4134 010174 012701 070424      8$:  MOV      #IBUFF+10,R1  ;SET TO START OF BAD SECTOR DATA
4135
4136 010200 022711 177777      9$:  CMP      #-1,(R1)      ;TEST IF WORD ALL ONES (END OF DATA)
4137 010204 001422              BEQ      11$            ;YES-SKIP
4138 010206 012123              MOV      (R1)+,(R3)+   ;STORE CYLINDER
4139 010210 012123              MOV      (R1)+,(R3)+   ;      TRACK AND SECTOR
4140 010212 005205              INC      R5            ;BUMP ERROR COUNTER
4141 010214 022705 000025      CMP      #21.,R5      ;DOES IT TOTAL 20 FOR THIS FORMAT?
4142 010220 001367              BNE      9$            ;NO-TEST AND MORE NEXT ADDRESS
4143 010222 012737 056137 001360  MOV      #EM53,EM1N
4144 010230 104001              ERROR    1              ;TO MANY BAD SECTORS
4145 010232 043737 001630 001354 10$:  BIC      DRVBIT,$DEV   ;CLEAR DRIVE FROM TESTING
4146 010240 053737 001630 001666  BIS      DRVBIT,DRVDRP ;SET DRIVE DROPPED
4147 010246 000137 027716      JMP      NEWDRV        ;ABORT PASS
4148
4149 010252 005200              11$:  INC      R0            ;BUMP TO NEXT
4150 010254 022700 000001      CMP      #1,R0        ;NOW TESTING BS SOFT 26 SECTOR MODE?
4151 010260 001011              BNE      12$            ;NO-SKIP
4152 010262 012723 177777      MOV      #-1,(R3)+   ;INSERT END OF FIELD FLAG
4153 010266 010337 001644      MOV      R3,BSS26P    ;SET POINTER TO BAD SECTOR SOFTWARE FIELD
4154 010272 112737 000012 007722  MOVVB   #12,2$        ;SET TO FIRST SECTOR THIS MODE
4155 010300 000137 007676      JMP      1$            ;GO READ IT.
4156 010304 022700 000002      12$:  CMP      #2,R0        ;NOW TESTING BS FACT 24 SECTOR MODE?
4157 010310 001014              BNE      13$            ;NO-SKIP
4158 010312 012723 177777      MOV      #-1,(R3)+   ;INSERT END OF FIELD FLAG
4159 010316 112737 000001 007722  MOVVB   #1,2$        ;SET TO FIRST SECTOR THIS MODE
4160 010324 010537 001646      MOV      R5,BS26CT    ;STORE TOTAL BS COUNT 26 SECTOR MODE
4161 010330 005005              CLR      R5            ;CLEAR COUNTER FOR COUNTING 24 SECT BS
4162 010332 013703 001636      MOV      BSF24P,R3    ;SET POINTER FOR STORING BS
4163 010336 000137 007676      JMP      1$            ;GO READ
4164
4165 010342 022700 000003      13$:  CMP      #3,R0        ;NOW TESTING BS SOFT 24 SECTOR MODE
4166 010346 001011              BNE      14$            ;NO-SKIP
4167 010350 012723 177777      MOV      #-1,(R3)+   ;INSERT END OF FIELD FLAG
4168 010354 010337 001642      MOV      R3,BSS24P    ;STORE POINTER TO BSS 24 SECTOR MODE
4169 010360 112737 000013 007722  MOVVB   #13,2$        ;GET START OF FIELDS BSS 24 SECT MODE
4170 010366 000137 007676      JMP      1$            ;GO READ IT
4171
4172 010372 012723 177777      14$:  MOV      #-1,(R3)+   ;INSERT END OF FIELD FLAG
4173 010376 010537 001650      MOV      R5,BS24CT    ;STORE COUNT BSS 24 SECTOR MODE
4174
4175
4176 010402 012700 061054      MOV      #BS26,R0     ;GET START OF BAD SECTOR BUFFER
4177      :      MOV      #BS24+52.,R3 ;GET END OF BUFFER
4178
4179      :27$:  CMP      #312,(R0)+   ;TEST IF ANY SECTORS BAD IN CYL 312
4180      :      BEQ      26$      ;YES - GET OUT
4181      :
4182      :      CMP      R0,R3    ;CHECK IF ALL OF BUFFER TESTED

```

```
4183      :      BNE      27$      :NO - LOOP
4184      :      BR       28$      :EXIT
4185 010406 005003      :      CLR      R3         :COUNT
4186 010410 022710 177777 27$:      CMP      #-1,(R0)     :AT THE END OF TABLE ?
4187 010414 001404      :      BEQ      16$         :BRANCH IF SO
4188 010416 022720 000312      :      CMP      #312,(R0)+  :CYL 312 BAD ?
4189 010422 001372      :      BNE      27$         :BRANCH IF NOT TRY AGAIN
4190 010424 000424      :      BR       26$         :ELSE ERROR
4191 010426 005203      :      INC      R3         :
4192 010430 020327 000001 16$:      CMP      R3,#1         :SEE IF 1 ST HALF OF BS 26 TABLE
4193 010434 001003      :      BNE      17$         :BRANCH IF NOT
4194 010436 013700 001644      :      MOV      BSS26P,R0   :MOVE UP THE TABLE
4195 010442 000762      :      BR       27$         :
4196 010444 020327 000002 17$:      CMP      R3,#2         :SEE IF DID 2 ND HALF OF BS26 TABLE
4197 010450 001003      :      BNE      18$         :BRANCH IF NOT
4198 010452 012700 061200      :      MOV      #BS24,R0   :ELSE MOVE UP 1 ST HALF
4199 010456 000754      :      BR       27$         :
4200 010460 020327 000003 18$:      CMP      R3,#3         :SEE IF DID 1 ST HALF OF BS24 TABLE
4201 010464 001003      :      BNE      19$         :BRANCH IF NOT
4202 010466 013700 001642      :      MOV      BSS24P,R0   :ELSE MOVE UP 2ND HALF OF BS24
4203 010472 000746      :      BR       27$         :
4204 010474 000406      :      BR       28$         :ALL DONE
4205
4206 010476 012737 057020 001360 26$:      MOV      #DRVABT,EM1N  :'BAD SECTOR IN AREA FOR TEST'
4207 010504 104001      :      ERROR    1
4208 010506 000137 010002      :      JMP      25$
4209
4210 010512      :      28$:
4211
4212      :*****
4213      :*TEST 21          FORMAT IN 26 SECTOR FORMAT
4214      :*
4215      :*          FORMAT CYLINDER 312, TRACK 0 AND TRACK 1 FOR 26 SECTOR
4216      :*          FORMAT. VERIFY FORMAT AND THAT DATA LATE DID NOT
4217      :*          OCCUR WITH WRITE HEADER OR IN READING DATA
4218      :*          BUFFER AFTER READ HEADER.
4219      :*
4220      :*****
4221 010512 000004      :TST21: SCOPE
4222 010514 012737 000012 001262      :      MOV      #10,$TIMES  ;;DO 10. ITERATIONS
4223 010522 012737 000312 001676      :      MOV      #312,REFMT  :SET REFORMAT SWITCH
4224 010530 105037 010653      :      CLRB    10$         :CLEAR SECTOR POINTER
4225 010534 105037 010557      :      CLRB    11$         :CLEAR TRACK POINTER
4226 010540 104416      :      TSSINIT  :CLEAR SUBSYSTEM
4227 010542 104003      :      ERROR    3         :BAD INIT ERROR
4228
4229 010544 004437 035674      :      JSR      R4,LRLOAD  :LOAD 'L' REGS
4230 010550 000127      :      WRHEAD  :WRITE HEADER
4231 010552 177676      :      -102     :WORD CNT FOR 26 SECTOR MODE
4232 010554 072414      :      OBUFF   :BUFFER
4233 010556 000      :      .BYTE    0         :SECTOR 0
4234 010557 000      :      .BYTE    0         :TRACK 0
4235 010560 000312      :      312     :CYL 0
4236
4237 010562 004437 042260      :      JSR      R4,GENCOM  :GENERATE DATA
4238 010566 000600      :      000600 :BUILD HEADERS-NO BAD SECTORS
```



```
4295 011002          117$:
4296 011002 104416
4297 011004 104003
4298 011006 004437 036362
4299 011012 104421
4300 011014 104004
4301 011016 104013
4302 011020 104002
4303
4304
4305 011022 004437 042260
4306 011026 100200
4307 011030 000414
4308 011032 104015
4309
4310 011034 013700 001634
4311 011040 005300
4312 011042 001407
4313 011044 004437 042260
4314
4315 011050 040000
4316 011052 000403
4317 011054 104016
4318 011056 000770
4319 011060 104415
4320
4321 011062 105737 001607
4322 011066 001010
4323
4324 011070 112737 000001 010557
4325 011076 112737 000001 010653
4326 011104 000137 010544
4327
4328 011110
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348 011110 000004
4349 011112 012737 000012 001262
4350 011120 012737 000312 001676

TSSINIT          ;CLEAR SUBSYSTEM
ERROR 3          ;BAD INIT ERROR
JSR R4,RDSTHD    ;GO READ & SEQUENCE HEADERS
TCHKOP           ;CONTROLLER ERROR RETURN
ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
ERROR 13         ;'DATA LATE SET RESULT OF DATA BUFFER READ'
ERROR 2          ;'OPERATION TO SLOW' MESSAGE
                ;OR 'HEADER 0 NOT FOUND' MESSAGE

JSR R4,GENCOM
100200           ;COMPARE IBUF & OBUF (HEADERS)
BR 6$           ;GOOD RETURN-NO MISCOMPARES
ERROR 15        ;REPORT 1ST MISCOMPARES

MOV ERRLMT,RO   ;GET ERROR LIMIT
DEC RO          ;DECREMENT IT
BEQ 6$         ;EXIT IF ZERO
JSR R4,GENCOM

040000          ;RESUME COMPARE
BR 6$          ;GOOD RETURN-NO MORE ERRORS
ERROR 16        ;REPORT NEXT ERROR LINE
BR 12$         ;LOOP
SCOPE1         ;LOCAL ERROR LOOP TO 117$

6$: TSTB L,DT   ;WAS TRACK 1 JUST DONE?
BNE 8$         ;YES-SKIP

MOVB #1,11$    ;CHANGE PARAM TO LOAD 'L' WITH
MOVB #1,10$    ;TRACK 2
JMP 9$        ;JUMP TO DO ENTIRE TEST ON TRK 1

8$:

.SBTTL **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.
*****
TST22: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
MOV #312, REFMT ;SET REFORMAT SWITCH
```

```
4351 011126 104416 TSSINIT ;CLEAR SUBSYSTEM
4352 011130 104003 ERROR 3 ;BAD INIT ERROR
4353
4354 011132 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4355 011136 000127 WRHEAD ;WRITE HEADER
4356 011140 177676 -102 ;WORD COUNT FOR 26 SECTOR MODE
4357 011142 072414 OBUFF ;BUFFER ADDRESS
4358 011144 000 .BYTE 0 ;SECTOR
4359 011145 000 .BYTE 0 ;TRACK
4360 011146 000312 312 ;CYLINDER
4361
4362
4363 011150 004437 042260 JSR R4,GENCOM ;GENERATE HEADERS
4364 011154 000600 600 ;WITH NO BS BITS
4365
4366 011156 012700 072416 MOV #OBUFF+2,R0 ;RESET BIT 15 IN WORD 2 OF
4367 011162 042720 100000 BIC #BIT15,(R0)+ ;SECTOR 0 HEADER AND BIT 14
4368 011166 042720 100000 BIC #BIT15,(R0)+ ;IN WORD 2 OF SECTOR 1 HEADER.
4369 011172 005720 TST (R0)+ ;ALSO CORRECT THE VRC
4370 011174 042720 040000 BIC #BIT14,(R0)+
4371 011200 042710 040000 BIC #BIT14,(R0)
4372
4373 011204 104417 TLOADRK ;LOAD RK REGS
4374 011206 104431 TWAT112 ;WAIT FOR INTERRUPT
4375 011210 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4376
4377 011212 104421 TCHKOP ;CHECK IF ANY ERRORS
4378 011214 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4379 011216 012737 011224 001110 MOV #4$, $LPERR ;SET LOCAL LOOP ON ERROR
4380 011224 104416 4$: TSSINIT
4381 011226 104003 ERROR 3
4382 011230 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4383 011234 000123 WRDATA ;WRITE DATA
4384 011236 177400 -400 ;WORD COUNT
4385 011240 072414 OBUFF ;BUS ADDRESS
4386 011242 000 5$: .BYTE 0 ;SECT 0
4387 011243 000 .BYTE 0 ;TRACK 0
4388 011244 000312 312 ;CYL 312
4389
4390 011246 104417 1$: TLOADRK ;LOAD RK REGS
4391 011250 104424 TWAT32 ;WAIT FOR INTERRUPT
4392 011252 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4393
4394 011254 022737 000002 011242 CMP #2,5$ ;JUST READ SECTOR 2?
4395 011262 001415 BEQ 6$ ;YES - SKIP
4396
4397 011264 104422 TCHKWE ;CHECK OPERATION WITH ERROR
4398 011266 000000 0 ;
4399 011270 000100 100 ;EXPECTED BSE
4400 011272 000000 0 ;
4401 011274 104004 ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
4402
4403 011276 104415 SCOP1 ;LOCAL ERROR LOOP TO 4$
4404
4405 011300 122737 000002 011242 CMPB #2,5$ ;WAS SECTOR SET TO 2
4406 011306 001405 BEQ 7$ ;YES-SKIP
```

```
4407 011310 105237 011242          INCB  5$          ;BUMP TO NEXT SECTOR
4408 011314 000743          BR    4$          ;LOOP
4409
4410 011316 104421          6$:  TCHKOP          ;CHECK FOR GOOD OPERATION
4411 011320 104004          ERROR  4 ; OR 5,6,7 ;REPORT ALL ERRORS
4412
4413 011322          7$:
4414          ;*****
4415          ;*TEST 23          HEADER VRC ERROR
4416          ;*
4417          ;*          FORMAT CYLINDER 312, TRACK 0, ON SCRATCH PACK TO HAVE
4418          ;*          16 SECTORS WITH BAD HEADER VRC.  ISSUE A WRITE DATA
4419          ;*          OF EACH OF THE SECTORS WITH A BAD HEADER VRC.  MAKE
4420          ;*          SURE HEADER VRC ERROR SETS.  ISSUE A WRITE DATA TO
4421          ;*          A GOOD HEADER AND MAKE SURE NO ERROR OCCURS.
4422          ;*****
4423          ;*****
4424 011322 000004          TST23: SCOPE
4425 011324 012737 000012 001262          MOV   #10, $TIMES          ;;DO 10. ITERATIONS
4426 011332 012737 000312 001676          MOV   #312, REFMT          ;SET REFORMAT SWITCH
4427 011340 104416          TSSINIT          ;CLEAR SUBSYSTEM
4428 011342 104003          ERROR  3          ;BAD INIT ERROR
4429
4430 011344 004437 035674          JSR   R4, LRLOAD          ;LOAD 'L' REGS
4431 011350 000127          WRHEAD          ;WRITE HEADER
4432 011352 177676          -102          ;WORD COUNT
4433 011354 072414          OBUFF          ;BUFF ADD
4434 011356 000          .BYTE  0          ;SECT
4435 011357 000          .BYTE  0          ;TRACK
4436 011360 000312          312          ;CYL
4437
4438 011362 004437 042260          JSR   R4, GENCOM
4439 011366 000600          600          ;BUILD HEADERS NO BSE
4440
4441 011370 012700 072420          MOV   #OBUFF+4, R0          ;GET ADDRESS OF VRC HDRO
4442 011374 012703 000001          MOV   #BIT0, R3          ;SET FOR BIT CHANGE SELECT
4443 011400 030310          1$:  BIT   R3, (R0)          ;CHECK A VRC BIT
4444 011402 001402          BEQ   2$          ;SKIP IF ZERO
4445 011404 040310          BIC   R3, (R0)          ;ELSE CLEAR IT
4446 011406 000401          BR    3$          ;SKIP
4447 011410 050310          2$:  BIS   R3, (R0)          ;IF ZERO SET IT
4448 011412 062700 000006          3$:  ADD   #6, R0          ;BUMP TO NEXT VRC WORD
4449
4450 011416 006303          ASL   R3          ;SHIFT THE SELECT
4451 011420 001367          BNE   1$          ;IF BIT NOT SHIFTED OUT-LOOP
4452
4453 011422 104417          TLOADRK          ;LOAD RK REGS
4454 011424 104431          TWAT112          ;WAIT FOR INTERRUPT
4455 011426 104002          ERROR  2          ;TO SLOW/NOT COMPLETE ERROR
4456 011430 104421          TCHKOP          ;CHECK OPERATION COMPLETE
4457 011432 104004          ERROR  4 ; OR 5,6,7 ;REPORT ALL ERRORS
4458
4459 011434 012737 011442 001110          4$:  MOV   #4$, $LPERR          ;SET LOCAL LOOP
4460 011442 104416          TSSINIT          ;CLEAR SUBSYSTEM
4461 011444 104003          ERROR  3          ;BAD INIT ERROR
4462
```

```
4463 011446 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4464 011452 000123 WRDATA ;WRITE DATA
4465 011454 177400 -400 ;WORD COUNT
4466 011456 072414 OBUFF ;BUFFER ADD
4467 011460 000 5$: .BYTE 0 ;SECT
4468 011461 000 .BYTE 0 ;TRACK
4469 011462 000312 312 ;CYL
4470
4471 011464 104417 TLOADRK ;LOAD RK REG
4472 011466 104424 TWAT32 ;WAIT FOR INTERRUPT
4473 011470 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4474
4475 011472 022737 000020 011460 CMP #16.,5$ ;WAS THIS WRITE SECTOR 16?
4476 011500 001415 BEQ 6$ ;YES-SKIP
4477 ;ELSE
4478 011502 104422 TCHKWE ;CHECK OPERATION WITH ERROR
4479 011504 000000 0
4480 011506 000040 40 ;HVRC EM EXPECTED
4481 011510 000000 0
4482 011512 104004 ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
4483
4484 011514 104415 SCOPI ;LOCAL LOOP TO 4$
4485
4486 011516 105237 011460 INCB 5$ ;BUMP SECTOR IN 'L' REG
4487 011522 022737 000016 011460 CMP #16,5$ ;IF SECTOR IS 16 OR LESS
4488 011530 003744 BLE 4$ ;LOOP
4489 011532 000402 BR 7$ ;ELSE EXIT
4490 011534 104421 6$: TCHKOP ;CHECK LAST OPERATION NO ERRORS
4491 011536 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4492
4493 011540 7$:
4494 ;*****
4495 ;*TEST 24 BAD SECTOR ERROR AND HVRC ERROR
4496 ;*
4497 ;* FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR ZERO HAS
4498 ;* BOTH A BAD SECTOR ERROR AND HEADER VRC. ISSUE A WRITE DATA
4499 ;* TO CYLINDER 0, TRACK 0, SECTOR 0. MAKE SURE ONLY HEADER VRC
4500 ;* ERROR SETS.
4501 ;*
4502 ;*****
4503 011540 000004 TST24: SCOPE
4504 011542 012737 000012 001262 MOV #10.,$TIMES ;:DO 10. ITERATIONS
4505 011550 012737 000312 001676 MOV #312,REFMT ;:SET REFORMAT SWITCH
4506 011556 104416 TSSINIT ;:CLEAR SUBSYSTEM
4507 011560 104003 ERROR 3 ;:BAD INIT ERROR
4508
4509 011562 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REG
4510 011566 000127 WRHEAD ;WRITE HEADER
4511 011570 177676 -102 ;WORD CNT FOR 26 SECTOR MODE
4512 011572 072414 OBUFF ;BUFF ADD
4513 011574 000 .BYTE 0 ;SECTOR
4514 011575 000 .BYTE 0 ;TRACK
4515 011576 000312 312 ;CYLINDER
4516
4517 011600 004437 042260 JSR R4,GENCOM
4518 011604 000600 600 ;BUILD HEADERS-NO BSE
```

```
4519
4520 011606 042737 100000 072416      BIC      #BIT15,OBUFF+2 ;CLEAR BIT TO SET BSE,LEAVE VRC BAD.
4521
4522 011614 104417      TLOADRK      ;LOAD RK REGS
4523 011616 104431      TWAT112     ;WAIT FOR INTERRUPT
4524 011620 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
4525
4526 011622 104421      TCHKOP      ;CHECK FOR ANY ERRORS
4527 011624 104004      ERROR      4 ; OR 5,6,7 ;REPORT ALL ERRORS
4528
4529 011626 004437 035674      JSR      R4,LRLOAD ;LOAD 'L' REGS
4530 011632 000123      WRDATA      ;WRITE DATA
4531 011634 177400      -400        ;WORD COUNT
4532 011636 072414      OBUFF      ;BUFF ADD
4533 011640 000        .BYTE      0      ;SECTOR
4534 011641 000        .BYTE      0      ;TRACK
4535 011642 000312      312        ;CYLINDER
4536
4537 011644 104417      TLOADRK      ;LOAD RK REGS
4538 011646 104424      TWAT32     ;WAIT FOR INTERRUPT
4539 011650 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
4540
4541 011652 104422      TCHKWE      ;CHECK OPERATION WITH EXPECTED ERR
4542 011654 000000      0
4543 011656 000040      40         ;HVRC ERR EXPECTED
4544 011660 000000      0
4545 011662 104004      ERROR      4 ; OR 5,6,7 ;REPORT ALL DISCREPENCIES
4546
4547
4548      :*****
4549      *TEST 25      OPERATION INCOMPLETE
4550
4551      *
4552      *      FORMAT CYLINDER 312, TRACK 0 SUCH THAT SECTOR 21 HAS THE
4553      *      WRONG FORMAT.  ISSUE A WRITE DATA OF 400 TO CYLINER 0,
4554      *      TRACK 0, SECTOR 21.  MAKE SURE OPI SET.
4555      :*****
4555 011664 000004      TST25:  SCOPE
4556 011666 012737 000012 001262      MOV      #10, $TIMES ;:DO 10. ITERATIONS
4557 011674 012737 000312 001676      MOV      #312,REFMT ;:SET REFORMAT SWITCH
4558 011702 104416      TSSINIT     ;:CLEAR SUBSYSTEM
4559 011704 104003      ERROR      3      ;:BAD INIT ERROR
4560
4561 011706 004437 035674      JSR      R4,LRLOAD ;LOAD 'L' REGS
4562 011712 000127      WRHEAD     ;WRITE HEADER
4563 011714 177676      -102      ;WORD COUNT FOR 26 SECT MODE
4564 011716 072414      OBUFF      ;BUFF ADD
4565 011720 000        .BYTE      0      ;SECTOR
4566 011721 000        .BYTE      0      ;TRACK
4567 011722 000312      312        ;CYLINDER
4568
4569 011724 004437 042260      JSR      R4,GENCOM ;BUILD HEADERS-NO BSE ERRORS
4570 011730 000600      600
4571
4572 011732 052737 001000 072614      BIS      #BIT9,OBUFF+200 ;CHANGE FORMAT IN SECTOR 25
4573 011740 052737 001000 072616      BIS      #BIT9,OBUFF+202 ;CORRECT THE VRC
4574
```

```
4575 011746 104417 TLOADRK ;LOAD RK REGS
4576 011750 104431 TWAT112 ;WAIT FOR INTERRUPT
4577 011752 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
4578
4579 011754 104421 TCHKOP ;CHECK FOR ANY ERRORS
4580 011756 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4581
4582 011760 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4583 011764 000123 WRDATA ;WRITE DATA
4584 011766 177400 -400 ;400 WORDS
4585 011770 072414 OBUFF ;BUFF ADD
4586 011772 025 .BYTE 25 ;SECTOR 25
4587 011773 000 .BYTE 0 ;TRACK 0
4588 011774 000312 312 ;CYL 312
4589
4590 011776 104417 TLOADRK ;LOAD RK REGS
4591 012000 104425 TWAT48 ;WAIT FOR INTERRUPT
4592 012002 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
4593
4594 012004 104422 TCHKWE ;CHECK OPERATION EXPECTED ERROR
4595 012006 000000 0 ;
4596 012010 000020 20 ;OPI EXPECTED
4597 012012 000000 0 ;
4598 012014 104004 ERROR 4 ; OR 5,6,7 ;REPORT ANY DISCREPENCIES
4599
4600 *****
4601 *TEST 26 OPI WITH HVRC ERROR
4602 *
4603 * FORMAT CYLINDER 312, TRACK 0 SUCH THAT A HEADER VRC
4604 * ERROR IS PRESENT AND SECTOR 17 HAS THE WRONG FORMAT.
4605 * ISSUE A WRITE DATA OF 400 WORDS TO CYLINDER 312,
4606 * TRACK 0, SECTOR 17. THAT BOTH OPERATION INCOMPLETE
4607 * AND HEADER VRC SET.
4608 *****
4609 012016 000004 TST26: SCOPE
4610 012020 012737 000012 001262 MOV #10, $TIMES ;DO 10. ITERATIONS
4611 012026 012737 000312 001676 MOV #312, REFMT ;SET REFORMAT SWITCH
4612 012034 104416 TSSINIT ;CLEAR SUBSYSTEM
4613 012036 104003 ERROR 3 ;BAD INIT ERROR
4614
4615 012040 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4616 012044 000127 WRHEAD ;WRITE HEADER
4617 012046 177676 -102 ;WORD COUNT FOR 26 SECT MODE
4618 012050 072414 OBUFF ;BUS ADDRESS
4619 012052 000 .BYTE 0 ;SECTOR
4620 012053 000 .BYTE 0 ;TRACK
4621 012054 000312 312 ;CYLINDER
4622
4623 012056 004437 042260 JSR R4,GENCOM
4624 012062 000600 600 ;BUILD HEADER- NO BSE ERRORS
4625
4626 012064 012700 072550 MOV #OBUFF+134, R0 ;GET ADDRESS 2ND WORD HDR 17(8)
4627 012070 052720 001000 BIS #BIT9, (R0)+ ;SET FORMAT 24 SECT PER TRACK
4628 012074 052720 001000 BIS #BIT9, (R0)+ ;SET VRC BIT
4629 012100 062700 000004 ADD #4, R0 ;BUMP TO HVRC WORD HDR 20(8)
4630 012104 032710 000001 BIT #BIT0, (R0) ;TEST BIT 0
```

```

4631 012110 001403          BEQ      1$          ;RESET-SKIP
4632 012112 042710 000001   BIC      #BIT0,(R0) ;CLEAR BIT
4633 012116 000402          BR       2$          ;
4634 012120 052710 000001   1$:     BIS      #BIT0,(R0) ;SET BIT
4635                                     ;FORCE OPI AND HVRC ERROR
4636 012124 104417          2$:     TLOADRK    ;LOAD RK REGS
4637 012126 104431          TWAT112 ;WAIT FOR INTERRUPT
4638 012130 104002          ERROR 2   ;TO SLOW/NOT COMPLETE ERROR
4639
4640 012132 104421          TCHKOP   ;CHECK FOR ANY ERRORS
4641 012134 104004          ERROR 4 ; OR 5,6,7 ;YES-REPORT ALL ERRORS
4642
4643 012136 004437 035674   JSR      R4,LRLOAD ;LOAD 'L' REGS
4644 012142 000123          WRDATA   ;WRITE DATA
4645 012144 177400          -400    ;400 WORDS
4646 012146 072414          OBUFF   ;BUFF ADDRESS
4647 012150 017           .BYTE 17 ;SECT 17
4648 012151 000           .BYTE 0  ;TRACK 0
4649 012152 000312          312    ;CYLINDER 312
4650
4651 012154 104417          TLOADRK    ;LOAD RK REGS
4652 012156 104425          TWAT48    ;WAIT FOR INTERRUPT
4653 012160 104002          ERROR 2   ;TO SLOW/NOT COMPLETE
4654
4655 012162 104422          TCHKWE   ;CHECK WITH EXPECTED ERROR
4656 012164 000000          0
4657 012166 000060          60     ;HVRC ERR & OPI EXPECTED
4658 012170 000000          0
4659 012172 104004          ERROR 4 ;OR 5,6,7

```

```

*****
*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
*****

```

```

4660
4661
4662
4663
4664
4665
4666
4667
4668
4669 012174 000004          TST27: SCOPE
4670 012176 012737 000012 001262   MOV      #10, $TIMES ;:DO 10. ITERATIONS
4671 012204 012737 000312 001676   MOV      #312,REFMT ;SET REFORMAT SWITCH
4672
4673 012212 104416          TSSINIT   ;CLEAR SUBSYSTEM
4674 012214 104003          ERROR 3   ;BAD INIT ERROR
4675
4676 012216 004437 035674   JSR      R4,LRLOAD ;LOAD 'L' REGISTERS
4677 012222 000127          WRHEAD   ;WRITE HEADER
4678 012224 177676          -102    ;WORD COUNT FOR 26 SECTOR MODE
4679 012226 072414          OBUFF   ;BUFF ADD
4680 012230 000           .BYTE 0  ;SECTOR
4681 012231 000           .BYTE 0  ;TRACK
4682 012232 000312          312    ;CYLINDER
4683
4684 012234 004437 042260   JSR      R4,GENCOM
4685 012240 000600          600    ;BUILD HEADERS-NO BSE ERRORS
4686

```



```

4687 012242 012700 072560      MOV    #OBUF+144,R0    ;ADDRESS OF HEAD 20 HVRC WORD
4688 012246 012701 000002      MOV    #BIT1,R1       ;BIT 1 CONSTANT
4689 012252 030110              BIT    R1,(R0)        ;TEST BIT 1 SET
4690 012254 001402              BEQ    1$             ;RESET-SKIP
4691 012256 040110              BIC    R1,(R0)        ;ELSE CLEAR BIT 1
4692 012260 000401              BR     2$             ;SKIP
4693 012262 050110      1$:   BIS    R1,(R0)    ;SET BIT 1
4694
4695 012264 104417      2$:   TLOADRK          ;LOAD RK REGS
4696 012266 104431      TWAT112          ;WAIT FOR INTERRUPT
4697 012270 104002      ERROR 2          ;TO SLOW/NOT COMPLETE
4698
4699 012272 104421      TCHKOP          ;CHECK FOR ANY ERROR
4700 012274 104004      ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
4701
4702 012276 004437 035674      JSR    R4,LRLOAD    ;LOAD 'L' REGISTER
4703 012302 000123      WRDATA          ;WRITE DATA
4704 012304 177400      -400            ;WORD COUNT
4705 012306 072414      OBUF           ;BUFF ADD
4706 012310 021          .BYTE 21         ;SECTOR
4707 012311 000          .BYTE 0         ;TRACK
4708 012312 000312      312            ;CYLINDER
4709 012314 104417      TLOADRK          ;LOAD RK REGS
4710 012316 104424      TWAT32          ;WAIT FOR INTERRUPT
4711 012320 104002      ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
4712
4713 012322 104421      TCHKOP          ;CHECK FOR ANY ERROR
4714 012324 104004      ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS.
4715
4716
4717
4718
4719

```

.SBTTL **DATA TRANSFER TESTS

```

*****
*TEST 30      WRITE AND READ ONE SECTOR
*
*      FORMAT CYLINDER 312, ALL TRACKS AND CYLINDER 313, TRACK 0
*      TO AGREED 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.
*****

```

```

4727
4728 012326 000004      TST30:  SCOPE
4729 012330 012737 000012 001262      MOV    #10, $TIMES    ;;DO 10. ITERATIONS
4730 012336 012737 000312 001676      MOV    #312,REFMT    ;SET REFORMAT SWITCH
4731 012344 104416      TSSINIT          ;CLEAR SUBSYSTEM
4732 012346 104003      ERROR 3          ;BAD INIT ERROR
4733
4734 012350 012737 000312 012376      MOV    #312,7$       ;PRESET CYL POINTER
4735 012356 105037 012375      CLRB  2$          ;CLEAR TRACK POINTER
4736
4737 012362 004437 035674      1$:   JSR    R4,LRLOAD    ;LOAD 'L' REG
4738 012366 000127      WRHEAD          ;WRITE HEADER
4739 012370 177676      -102           ;WORD COUNT FOR 26 SECTOR MODE
4740 012372 072414      OBUF           ;BUFF ADDRESS
4741 012374 000          .BYTE 0         ;SECTOR
4742 012375 000          .BYTE 0         ;TRACK

```

4743	012376	000312		7\$:	312		;CYLINDER
4744							
4745	012400	004437	042260		JSR	R4,GENCOM	
4746	012404	001200			1200		;BUILD HDRS-INCLUDE BAD SECTORS
4747							
4748	012406	104417			TLOADRK		;LOAD RK REGS
4749	012410	104431			TWAT112		;WAIT FOR INTERRUPT
4750	012412	104002			ERROR 2		;TO SLOW/NOT COMPLETE ERROR
4751							
4752	012414	104421			TCHKOP		;CHECK FOR ANY ERRORS
4753	012416	104004			ERROR 4 ; .OR 5,6,7		;REP'RT ALL ERRORS
4754							
4755	012420	022737	000313 012376		CMP	#313,7\$;TEST IF DONE 313 TK 0
4756	012426	001414			BEQ	3\$;YES - SKIP
4757	012430	123727	012375 000002		CMPB	2\$,#2	;DID WE JUST FORMAT TRACK 2
4758	012436	001403			BEQ	8\$;YES-SKIP
4759	012440	105237	012375		INCB	2\$;BUMP TO NEXT TRACK
4760	012444	000746			BR	1\$;GO FORMAT NEXT TRACK
4761							
4762	012446	105037	012375	8\$:	CLRB	2\$;CLEAR TRACK POINTER
4763	012452	005237	012376		INC	7\$;BUMP CYL TO 313
4764	012456	000741			BR	1\$;GO FORMAT 313 TK 0
4765							
4766	012460	004437	035674	3\$:	JSR	R4,LRLOAD	;LOAD "L" REGS
4767	012464	000123			WRDATA		;WRITE DATA
4768	012466	177400			-400		;ONE SECTOR WORD COUNT
4769	012470	072414			OBUFF		;BUFF ADDRESS
4770	012472	012			.BYTE	12	;SECTOR 12
4771	012473	000			.BYTE	0	;TRACK 0
4772	012474	000312			312		;CYLINDER 312
4773							
4774	012476	004437	042260		JSR	R4,GENCOM	
4775	012502	000001			1		;BUILD DATA PATTERN 1
4776	012504	000400			400		;400 WORDS LONG
4777	012506	012737	012514 001110		MOV	#4\$, \$LPERR	;SET FOR LOCAL LOOP
4778	012514	104417		4\$:	TLOADRK		;LOAD RK REGS
4779	012516	104431			TWAT112		;WAIT FOR INTERRUPT
4780	012520	104402			ERROR 2		;TO SLOW/NOT COMPLETE ERROR
4781							
4782	012522	104421			TCHKOP		;CHECK FOR ANY ERRORS
4783	012524	104004			ERROR 4 ; OR 5,6,7		;REPORT ALL ERRORS
4784							
4785	012526	004437	035674		JSR	R4,LRLOAD	;LOAD "L" REGS
4786	012532	000121			RDDATA		;READ DATA
4787	012534	177400			-400		;400 WORDS
4788	012536	070414			IBUFF		;BUFF ADD
4789	012540	012			.BYTE	12	;SECTOR 12
4790	012541	000			.BYTE	0	;TRACK 0
4791	012542	000312			312		;CYL 312
4792							
4793	012544	104417			TLOADRK		;LOAD RK
4794	012546	104424			TWAT32		;WAIT FOR INTERRUPT
4795	012550	104002			ERROR 2		;TO SLOW/NOT COMPLETE
4796							
4797	012552	104421			TCHKOP		;CHECK FOR ANY ERRORS
4798	012554	104004			ERROR 4 ; OR 5,6,7		;REPORT ALL ERRORS

```
4799
4800 012556 004437 042260 JSR R4,GENCOM
4801 012562 100001 100001 ;GO COMPARE DATA TO PATTERN 1
4802 012564 000400 400 ;400 WORDS LONG
4803 012566 000413 BR 6$ ;GOOD RETURN-NO DATA ERRORS
4804 012570 104015 ERROR 15 ;ERROR RETURN
4805
4806 012572 013700 001634 MOV ERRMT,RO ;GET ERROR LIMIT
4807 012576 005300 5$: DEC RO ;DEC LIMIT
4808 012600 001406 BEQ 6$ ;EXIT IF 0
4809 012602 004437 042260 JSR R4,GENCOM
4810 012606 040000 040000 ;CONTINUE COMPARE
4811 012610 000402 BR 6$ ;EXIT IF NO MORE ERRORS
4812 012612 104016 ERROR 16 ;ELSE REPORT MISCOMPARE
4813 012614 000770 BR 5$ ;LOOP
4814 012616 005037 001676 6$: CLR REFORMAT ;CLEAR REFORMAT SWITCH
4815
4816 *****
4817 *TEST 31 WRITE DATA WITH BUS ADDRESS INCREMENT INHIBIT
4818 *
4819 * ISSUE A WRITE DATA OF ONE SECTOR TO CYLINDER 312,
4820 * TRACK 2, SECTOR 12 WITH INHIBIT BUS
4821 * ADDRESS INCREMENT. READ DATA BACK TO MAKE SURE
4822 * EVERY WORD IS THE SAME AND CORRECT.
4823 *
4824 *****
4825 012622 000004 TST31: SCOPE
4826 012624 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
4827
4828 012632 104416 TSSINIT ;CLEAR SUBSYSTEM
4829 012634 104003 ERROR 3 ;BAD INIT ERROR
4830
4831 012636 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4832 012642 000123 WRDATA ;WRDATA
4833 012644 177400 -400 ;-400 WORDS
4834 012646 072414 OBUFF ;OBUFF IS BUFF ADDRESS
4835 012650 012 .BYTE 12 ;SECTOR 12
4836 012651 001 .BYTE 1 ;TRACK 1
4837 012652 000312 312 ;CYLINDER 312
4838
4839 012654 052737 000020 001610 BIS #BAI,L.CS2 ;SET INCREMENT INHIBIT
4840 012662 004437 042260 JSR R4,GENCOM ;BUILD PATTERN
4841 012666 000016 16 ;PATTERN 16
4842 012670 000400 400 ;400 WORDS
4843
4844 012672 104417 TLOADRK ;LOAD RK REGS
4845 012674 104430 TWAT96 ;WAIT FOR INTERRUPT
4846 012676 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4847
4848 012700 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
4849 012702 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
4850
4851 012704 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
4852 012710 000121 RDDATA ;RDDATA
4853 012712 177400 -400 ;-400 WORDS
4854 012714 070414 IBUFF ;IBUFF IS BUFF ADDRESS
```

```
4855 012716 012 .BYTE 12 ;SECTOR 12
4856 012717 001 .BYTE 1 ;TRACK 1
4857 012720 000312 312 ;CYLINDER 312
4858
4859 012722 012700 000377 MOV #377,R0 ;SET COUNT TO SET OBUFF TO BE
4860 012726 012701 072416 MOV #OBUFF+2,R1 ;ALL THE FIRST WORD OF PATTERN
4861 012732 012703 072414 MOV #OBUFF,R3
4862
4863 012736 011321 1$: MOV (R3),(R1)+ ;MOV THE WORD
4864 012740 005300 DEC R0
4865 012742 001375 BNE 1$ ;LOOP UNTIL ALL WORDS SET
4866
4867 012744 104417 TLOADRK ;LOAD RK REGS
4868 012746 104424 TWAT32 ;WAIT FOR INTERRUPT
4869 012750 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
4870
4871 012752 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
4872 012754 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
4873
4874 012756 004437 042260 JSR R4,GENCOM ;COMPARE THE DATA
4875 012762 100000 100000
4876 012764 000400 400
4877 012766 000413 BR 2$
4878 012770 104015 ERROR 15
4879 012772 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
4880 012776 005300 64$: DEC R0 ;DECREMENT COUNT
4881 013000 001406 BEQ 65$ ;IF ZERO - EXIT
4882 013002 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
4883 013006 040000 40000
4884 013010 000402 BR 65$ ;NO MORE ERRORS - EXIT
4885 013012 104016 ERROR 16 ;REPORT NEXT ERROR
4886 013014 000770 BR 64$ ;LOOP
4887 013016 65$:
4888
4889 013016 2$:
4890 *****
4891 *TEST 32 WRITE DATA ADDRESS GREATER THAN 32K
4892 *
4893 * ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 177770.
4894 * MAKE SURE CORRECT DATA IS ON DISK.
4895 *
4896 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
4897 * OF MEMORY IS PRESENT.
4898 *
4899 *****
4900 013016 000004 TST32: SCOPE
4901 013020 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
4902 013026 123727 001326 000001 CMPB $MAMS1,#1 ;TEST IF >32K MEM
4903 013034 002016 BGE 2$ ;YES-SKIP
4904
4905 013036 032737 000002 001664 BIT #MEMSZB,OPTFLG ;TEST IF REPORT ALREADY MADE
4906 013044 001011 BNE 1$ ;YES -SKIP
4907
4908 013046 104401 051503 TYPE ,OPR011 ;"INSUFFICIENT MEMORY DATA TRANSFER WITH
4909 013052 104401 051646 TYPE ,OPR012 ;ADDRESS >32K
4910 013056 052737 000002 001664 BIS #MEMSZB,OPTFLG ;SET FLAG
```

Address	OpCode	Operand 1	Operand 2	Label	Comments	Notes
4911	013064	104401	051662		TYPE	OPR015 ;BYPASSED''
4912	013070	000470			BR	4\$;EXIT
4913	013072	012737	013100	001110	MOV	#5\$, \$LPERR ;SET LOCAL LOOP ON ERROR ADDRESS
4914	013100					1\$: 2\$: 5\$:
4915	013100	104416			TSSINIT	;CLEAR SUBSYSTEM
4916	013102	104003			ERROR	3 ;BAD INIT ERROR
4917						
4918	013104	004437	035674		JSR	R4, LRLOAD ;LOAD 'L' REGS
4919	013110	000123			WRDATA	;WRITE DATA
4920	013112	177400			-400	;400 WORDS
4921	013114	177770			177770	;BUS ADDRESS IN 32K -10 BYTES
4922	013116	016			.BYTE	16 ;SECTOR 16
4923	013117	000			.BYTE	0 ;TRACK 0
4924	013120	000312			312	;CYLINDER 312
4925	013122	004437	042260		JSR	R4, GENCOM ;GENERATE DATA
4926	013126	010010			1000	;PATTERN 10, MEM. MANAGEMENT FOR DEST.
4927	013130	001777			1777	;RELOCATION ARGUMENT
4928	013132	000400			400	;400 WORDS
4929						
4930	013134	104417			TLOADRK	;LOAD RK REGS
4931	013136	104430			TWAT96	;WAIT FOR INTERRUPT
4932	013140	104002			ERROR	2 ;TO SLOW/NOT COMPLETE ERROR
4933						
4934	013142	104421			TCHKOP	;CHECK OPERATION FOR ANY ERRORS
4935	013144	104004			ERROR	4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
4936	013146	104415			SCOPI	;LOCAL LOOP ON ERROR TO 5\$
4937						
4938	013150	004437	042260		JSR	R4, GENCOM ;CLEAR Ibuff TO 1'S.
4939	013154	002007			2007	
4940	013156	001000			1000	
4941						
4942	013160	004437	035674		JSR	R4, LRLOAD ;LOAD 'L' REGS
4943	013164	000121			RDDATA	;RDDATA
4944	013166	177400			-400	; -400 WORDS
4945	013170	070414			IBUFF	;IBUFF IS BUFF ADDRESS
4946	013172	016			.BYTE	16 ;SECTOR 16
4947	013173	000			.BYTE	0 ;TRACK 0
4948	013174	000312			312	;CYLINDER 312
4949	013176	104417			TLOADRK	;LOAD RK REGS
4950	013200	104424			TWAT32	;WAIT FOR INTERRUPT
4951	013202	104002			ERROR	2 ;TO SLOW/NOT COMPLETE ERROR
4952	013204	104421			TCHKOP	;CHECK OPERATION FOR ANY ERRORS
4953	013206	104004			ERROR	4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
4954	013210	004437	042260		JSR	R4, GENCOM ;COMPARE
4955	013214	110000			110000	;MEMORY MANAGEMENT FOR DESTINATION
4956	013216	001777			1777	;RELOCATION ARGUMENT
4957	013220	000400			400	;400 WORDS
4958	013222	000413			BR	4\$;NO ERROR-SKIP
4959	013224	104015			ERROR	15 ;REPORT FIRST MISCOMPARE
4960	013226	013700	001634		MOV	ERRLMT, RO ;GET ERROR LIMIT
4961	013232	005300		64\$:	DEC	RO ;DECREMENT COUNT
4962	013234	001406			BEQ	65\$;IF ZERO - EXIT
4963	013236	004437	042260		JSR	R4, GENCOM ;CONTINUE DATA COMPARE
4964	013242	050000			50000	
4965	013244	000402			BR	65\$;NO MORE ERRORS - EXIT
4966	013246	104016			ERROR	16 ;REPORT NEXT ERROR

4967 013250 000770
4968 013252
4969
4970 013252
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981 013252 000004
4982 013254 012737 000012 001262
4983 013262 123727 001326 000001
4984 013270 002001
4985
4986 013272 000462
4987
4988 013274 012737 013302 001110
4989
4990 013302
4991 013302 104416
4992 013304 104003
4993 013306 004437 035674
4994 013312 000123
4995 013314 177400
4996 013316 072414
4997 013320 017
4998 013321 000
4999 013322 000312
5000 013324 004437 042260
5001 013330 000011
5002 013332 000400
5003
5004 013334 104417
5005 013336 104430
5006 013340 104002
5007
5008 013342 104421
5009 013344 104004 035674
5010 013346 004437
5011 013352 000121
5012 013354 177400
5013 013356 177770
5014 013360 017
5015 013361 000
5016 013362 000312
5017
5018 013364 104417
5019 013366 104424
5020 013370 104002
5021 013372 104421
5022 013374 104004

BR 64\$;LOOP
65\$:
4\$:

*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.

TST33: SCOPE ;CLEAR SUBSYSTEM
MOV #10, \$TIMES ;;DO 10. ITERATIONS
CMPB \$MAMS1, #1 ;CHECK IF >32K MEMORY
BGE 2\$;YES-SKIP
1\$: BR 5\$;EXIT
2\$: MOV #3\$, \$LPERR ;SET LOCAL ERROR LOOP
3\$:
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 17 ;SECTOR 17
.BYTE 0 ;TRACK 0
312 ;CYLINDER 312
JSR R4, GENCOM ;GENERATE DATA IN OBUFF
11 ;PATTERN 11
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' REG
RDDATA ;READ DATA
-400 ;400 WORDS
177770 ;ACROSS 32K BOUNDARY
.BYTE 17 ;SECTOR 17
.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

```
5023 013376 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5024 013402 120000 120000 ;MEMORY MANAGEMENT WITH SOURCE
5025 013404 001777 1777 ;RELOCATION ARGUMENT
5026 013406 000400 400 ;COMPARE 400 WORDS
5027 013410 000413 - BR 5$ ;NO MISCOMPARE-EXIT
5028 013412 104015 ERROR 15 ;REPORT FIRST MISCOMPARE
5029 013414 013700 001634 MOV ERRHLT,RO ;GET ERROR LIMIT
5030 013420 005300 64$: DEC RO ;DECREMENT,COUNT
5031 013422 001406 BEQ 65$ ;IF ZERO - EXIT
5032 013424 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5033 013430 060000 60000 BR 65$ ;NO MORE ERRORS - EXIT
5034 013432 000402 ERROR 16 ;REPORT NEXT ERROR
5035 013434 104016 BR 64$ ;LOOP
5036 013436 000770
5037 013440
5038 013440
5039
5040 *****
5041 *TEST 34 WRITE DATA ADDRESS GREATER THAN 64K
5042 *
5043 * ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 377770.
5044 * MAKE SURE CORRECT DATA IS ON DISK.
5045 *
5046 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
5047 * OF MEMORY IS PRESENT.
5048 *****
5049 TST34: SCOPE
5050 013440 000004 MOV #10, $TIMES ;:DO 10. ITERATIONS
5051 013442 012737 000012 001262 CMPB $MAMS1,#2 ;:CHECK IF >64K MEMORY
5052 013450 123727 001326 000002 BGE 2$ ;:YES-SKIP
5053 013456 002016 BIT #MEMSZB,OPTFLG ;:TEST IF REPORT FLAG SET
5054 013460 032737 000002 001664 6$: BNE 1$ ;:NO-SKIP
5055
5056 013470 104401 051503 TYPE ,OPR011 ;:'INSUFFICIENT MEMORY-DATA XFER WITH
5057 013474 104401 051652 TYPE ,OPR013 ;:ADDRESS >64K
5058 013500 104401 051662 TYPE ,OPR015 ;:BYPASSED''
5059 013504 052737 000002 001664 BIS #MEMSZB,OPTFLG ;:SET FLAG
5060 013512 000467 1$: BR 5$
5061
5062 013514 012737 013522 001110 2$: MOV #3$, $LPERR ;:SET LOCAL LOOP ON ERROR
5063
5064 013522 3$:
5065 013522 104416 TSSINIT ;:CLEAR SUBSYSTEM
5066 013524 104003 ERROR 3 ;:BAD INIT ERROR
5067 013526 004437 042260 JSR R4,GENCOM ;:GENERATE DATA, PATTERN 11
5068 013532 010011 10011 ;:MEM MANAGEMENT ON DESTINATION
5069 013534 003777 3777 ;:RELOCATION ARGUMENT
5070 013536 000400 400 ;:400 WORDS
5071
5072 013540 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
5073 013544 000523 WRDATA!BA16 ;:WRITE DATA AND SET BA16
5074 013546 177400 -400 ;:400 WORDS
5075 013550 177770 177770 ;:ACROSS 64K BOUNDARY
5076 013552 020 .BYTE 20 ;:SECTOR 20
5077 013553 000 .BYTE 0 ;:TRACK 0
5078 013554 000312 312 ;:CYLINDER 312
```

```
5079
5080 013556 104417 TLOADRK ;LOAD RK REGS
5081 013560 104430 TWAT96 ;WAIT FOR INTERRUPT
5082 013562 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5083
5084 013564 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5085 013566 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5086 013570 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO 1'S
5087 013574 002007 2007
5088 013576 001000 1000
5089
5090 013600 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5091 013604 000121 RDDATA ;RDDATA
5092 013606 177400 -400 ;-400 WORDS
5093 013610 070414 Ibuff ;IBUFF IS BUFF ADDRESS
5094 013612 020 .BYTE 20 ;SECTOR 20
5095 013613 000 .BYTE 0 ;TRACK 0
5096 013614 000312 312 ;CYLINDER 312
5097 013616 104417 TLOADRK ;LOAD RK REGS
5098 013620 104424 TWAT32 ;WAIT FOR INTERRUPT
5099 013622 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5100
5101 013624 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5102 013626 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5103 013630 004437 042260 JSR R4,GENCOM ;CHECK DATA
5104 013634 110000 110000 ;MEMORY MANAGEMENT WITH DESTINATION
5105 013636 003777 3777 ;RELOCATION ARGUMENT
5106 013640 000400 400 ;400 WORDS
5107 013642 000413 BR 5$ ;NO MISCOMPARES-SKIP
5108 013644 104015 ERROR 15 ;REPORT FIRST ERROR
5109
5110 013646 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5111 013652 005300 64$: DEC RO ;DECREMENT COUNT
5112 013654 001406 BEQ 65$ ;IF ZERO - EXIT
5113 013656 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5114 013662 050000 50000
5115 013664 000402 BR 65$ ;NO MORE ERRORS - EXIT
5116 013666 104016 ERROR 16 ;REPORT NEXT ERROR
5117 013670 000770 BR 64$ ;LOOP
5118 013672
5119 013672
5120
5121 *****
5122 *TEST 35 READ DATA ADDRESS GREATER THAN 64K
5123 *
5124 * ISSUE A READ DATA OF 400 WORDS WITH ADDRESS = 377770.
5125 * CHECK MEMORY FOR CORRECT TRANSFER.
5126 *
5127 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
5128 * OF MEMORY IS PRESENT.
5129 *****
5130 TST35: SCOPE
5131 013674 012737 000012 001262 MOV #10, $TIMES ;DO 10. ITERATIONS
5132 013702 123727 001326 000002 CMPB $MAMS1, #2 ;CHECK IF >64K MEMORY
5133 013710 002001 BGE 2$ ;YES-SKIP
5134 013712 000462 1$: BR 5$ ;EXIT
```



```

5135
5136 013714 012737 000032 001110 2$: MOV #32,$LPERR ;SET LOCAL LOOP ON ERROR
5137
5138 013722 3$:
5139 013722 104416 TSSINIT ;CLEAR SUBSYSTEM
5140 013724 104003 ERROR 3 ;BAD INIT ERROR
5141 013726 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5142 013732 000123 WRDATA ;WRDATA
5143 013734 177400 -400 ;-400 WORDS
5144 013736 072414 OBUFF ;OBUFF IS BUFF ADDRESS
5145 013740 021 .BYTE 21 ;SECTOR 21
5146 013741 000 .BYTE 0 ;TRACK 0
5147 013742 000312 312 ;CYLINDER 312
5148 013744 004437 042260 JSR R4,GENCOM ;GENERATE DATA
5149 013750 000012 12 ;PATTERN 12
5150 013752 000400 400 ;400 WORDS
5151
5152 013754 104417 TLOADRK ;LOAD RK REGS
5153 013756 104430 TWAT96 ;WAIT FOR INTERRUPT
5154 013760 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5155
5156 013762 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5157 013764 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5158 013766 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5159 013772 000521 RDDATA!BA16 ;READ DATA AND SET BA16
5160 013774 177400 -400 ;400 WORDS
5161 013776 177770 177770 ;ACROSS 64K BOUNDARY
5162 014000 021 .BYTE 21 ;FROM SECTOR 21
5163 014001 000 .BYTE 0 ;TRACK 0
5164 014002 000312 312 ;CYLINDER 312
5165
5166 014004 104417 TLOADRK ;LOAD RK REGS
5167 014006 104424 TWAT32 ;WAIT FOR INTERRUPT
5168 014010 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5169
5170 014012 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5171 014014 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5172 014016 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5173 014022 120000 120000 ;MEM MANAGEMENT WITH SOURCE
5174 014024 003777 3777 ;RELOCATION ARGUMENT
5175 014026 000400 400 ;400 WORDS
5176 014030 000413 BR 5$ ;NO MISCOMPARES-SKIP
5177 014032 104015 ERROR 15 ;REPORT FIRST ERROR
5178
5179 014034 013700 001634 64$: MOV ERRLMT,R0 ;GET ERROR LIMIT
5180 014040 005300 DEC R0 ;DECREMENT COUNT
5181 014042 001406 BEQ 65$ ;IF ZERO - EXIT
5182 014044 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5183 014050 060000 60000
5184 014052 000402 BR 65$ ;NO MORE ERRORS - EXIT
5185 014054 104016 ERROR 16 ;REPORT NEXT ERROR
5186 014056 000770 BR 64$ ;LOOP
5187 014060 65$:
5188
5189 014060 5$:
5190 ::*****
  
```

```
5191' : *TEST 36 WRITE DATA ADDRESS GREATER THAN 96K
5192 : *
5193 : * ISSUE A WRITE DATA OF 400 WORDS WITH ADDRESS = 577770.
5194 : * MAKE SURE CORRECT DATA IS ON DISK.
5195 : *
5196 : * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K
5197 : * OF MEMORY IS PRESENT.
5198 : *
5199 : *
*****
5200 014060 000004 TST36: SCOPE
5201 014062 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
5202 014070 123727 001326 000003 CMPB $MAMS1, #3 ;;CHECK IF >96K MEMORY
5203 014076 002016 BGE 3$ ;;YES-SKIP
5204 014100 032737 000002 001664 1$: BIT #MEMSZB, OPTFLG ;;TEST IF REPORT FLAG SET
5205 014106 001011 BNE 2$ ;;NO-SKIP
5206
5207 014110 104401 051503 TYPE ,OPR011 ;'INSUFFICIENT MEMORY-DATA TRANSFET WITH
5208 014114 104401 051656 TYPE ,OPR014 ;ADDRESS >96K BYPASSED''
5209 014120 104401 051662 TYPE ,OPR015
5210 014124 052737 000002 001664 BIS #MEMSZB, OPTFLG ;SET REPORT FLAG
5211 014132 000463 2$: BR 6$
5212
5213 014134 012737 014142 001110 3$: MOV #4$, $LPERR ;SET LOCAL LOOP ON ERROR
5214
5215 014142 4$:
5216 014142 104416 TSSINIT ;CLEAR SUBSYSTEM
5217 014144 104003 ERROR 3 ;BAD INIT ERROR
5218 014146 004437 035674 JSR R4, LRLOAD ;LOAD 'L' REG
5219 014152 001123 WRDATA!BA17 ;WRITE DATA AND BA17
5220 014154 177400 -400 ;400 WORDS FROM
5221 014156 177770 177770 ;ACROSS 96K BOUNDARY
5222 014160 022 .BYTE 22 ;TO SECTOR 22
5223 014161 000 .BYTE 0 ;TRACK 0
5224 014162 000312 312 ;CYL 312
5225 014164 004437 042260 JSR R4, GENCOM ;GENERATE DATA
5226 014170 010013 10013 ;PATTERN 13 MEM MAN WITH DEST.
5227 014172 005777 5777 ;RELOCATION ARGUMENT
5228 014174 000400 400 ;400 WORDS
5229
5230 014176 104417 TLOADRK ;LOAD RK REGS
5231 014200 104430 TWAT96 ;WAIT FOR INTERRUPT
5232 014202 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5233
5234 014204 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5235 014206 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5236
5237 014210 004437 035674 JSR R4, LRLOAD ;LOAD 'L' REGS
5238 014214 000121 RDDATA ;RDDATA
5239 014216 177400 -400 ;-400 WORDS
5240 014220 070414 IBUFF ;IBUFF IS BUFF ADDRESS
5241 014222 022 .BYTE 22 ;SECTOR 22
5242 014223 000 .BYTE 0 ;TRACK 0
5243 014224 000312 312 ;CYLINDER 312
5244
5245 014226 104417 TLOADRK ;LOAD RK REGS
5246 014230 104424 TWAT32 ;WAIT FOR INTERRUPT
```

```
5247 014232 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
5248
5249 014234 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
5250 014236 104004          ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5251
5252 014240 004437 042260    JSR R4,GENCOM   ;COMPARE DATA
5253 014244 110000          110000         ;MEM MANAGEMENT WITH DESTINATION
5254 014246 005777          5777          ;RELOCATION ARGUMENT
5255 014250 000400          400           ;400 WORDS
5256 014252 000413          BR 6$         ;NO MISCOMPARES-BRANCH
5257 014254 104015          ERROR 15      ;REPORT 1ST ERROR
5258
5259 014256 013700 001634    MOV ERRLMT,R0  ;GET ERROR LIMIT
5260 014262 005300          64$: DEC R0     ;DECREMENT COUNT
5261 014264 001406          BEQ 65$      ;IF ZERO - EXIT
5262 014266 004437 042260    JSR R4,GENCOM ;CONTINUE DATA COMPARE
5263 014272 050000          50000
5264 014274 000402          BR 65$      ;NO MORE ERRORS - EXIT
5265 014276 104016          ERROR 16    ;REPORT NEXT ERROR
5266 014300 000770          BR 64$     ;LOOP
5267 014302
5268
5269 014302
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280 014302 000004          TST37: SCOPE
5281 014304 012737 000012 001262 MOV #10, $TIMES ;:DO 10. ITERATIONS
5282 014312 123727 001326 000003 CMPB $MAMS1, #3 ;:CHECK IF >96K MEMORY
5283 014320 002001          BGE 3$      ;:YES-SKIP
5284 014322 000462          2$: BR 6$
5285
5286 014324 012737 014332 001110 3$: MOV #4$, $LPERR ;:SET LOCAL LOOP ON ERROR
5287
5288 014332
5289 014332 104416          4$: TSSINIT    ;:CLEAR SUBSYSTEM
5290 014334 104003          ERROR 3     ;:BAD INIT ERROR
5291
5292 014336 004437 035674    JSR R4,LRLOAD ;:LOAD 'L' REGS
5293 014342 000123          WRDATA     ;:WRDATA
5294 014344 177400          -400      ;:-400 WORDS
5295 014346 072414          OBUFF     ;:OBUFF IS BUFF ADDRESS
5296 014350 005           .BYTE 5    ;:SECTOR 5
5297 014351 000           .BYTE 0    ;:TRACK 0
5298 014352 000312          312      ;:CYLINDER 312
5299 014354 004437 042260    JSR R4,GENCOM ;:GENERATE DATA
5300 014360 000014          14       ;:PATTERN 14
5301 014362 000400          400      ;:400 WORDS
5302
```

```

5303 014364 104417 TLOADRK ;LOAD RK REGS
5304 014366 104430 TWAT96 ;WAIT FOR INTERRUPT
5305 014370 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5306
5307 014372 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5308 014374 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5309 014376 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5310 014402 001121 RDDATA!BA17 ;READ DATA WITH BA17 SET
5311 014404 177400 -400 ;400 WORDS
5312 014406 177770 177770 ;ACROSS 96K BOUNDARY
5313 014410 005 .BYTE 5 ;FROM SECTOR 5
5314 014411 000 .BYTE 0 ;TRACK 0
5315 014412 000312 312 ;CYL 312
5316
5317 014414 104417 TLOADRK ;LOAD RK REGS
5318 014416 104424 TWAT32 ;WAIT FOR INTERRUPT
5319 014420 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5320
5321 014422 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5322 014424 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5323 014426 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5324 014432 120000 120000 ;MEM MANAGEMENT WITH SOURCE
5325 014434 005777 5777 ;RELOCATION ARGUMENT
5326 014436 000400 400 ;400 WORDS
5327 014440 000413 BR 6$ ;NO MISCOMPARES-SKIP
5328 014442 104015 ERROR 15 ;REPORT FIRST ERROR
5329
5330 014444 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5331 014450 005300 64$: DEC R0 ;DECREMENT COUNT
5332 014452 001406 BEQ 65$ ;IF ZERO - EXIT
5333 014454 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5334 014460 060000 60000
5335 014462 000402 BR 65$ ;NO MORE ERRORS - EXIT
5336 014464 104016 ERROR 16 ;REPORT NEXT ERROR
5337 014466 000770 BR 64$ ;LOOP
5338 014470 65$:
5339
5340 014470 6$:
5341 *****
5342 ;*TEST 40 PARTIAL SECTOR WRITE DATA
5343 ;*
5344 ;* ISSUE A WRITE DATA OF 103 WORDS TO CYLINDER 312,
5345 ;* HEAD 0, SECTOR 0. MAKE SURE THE SECTOR WAS
5346 ;* ZERO FILLED CORRECTLY.
5347 ;*
5348 *****
5349 014470 000004 TST40: SCOPE
5350 014472 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
5351 014500 104416 TSSINIT ;CLEAR SUBSYSTEM
5352 014502 104003 ERROR 3 ;BAD INIT ERROR
5353
5354 014504 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REG
5355 014510 000123 WRDATA ;WRITE DATA
5356 014512 177675 -103 ;WORD COUNT PARTIAL SECTOR
5357 014514 072414 OBUFF ;BUFF ADDRESS
5358 014516 007 .BYTE 7 ;SECTOR 7
```

```

5359 014517 000 .BYTE 0 ;TRACK 0
5360 014520 000312 312 ;CYLINDER 312
5361
5362 014522 004437 042260 JSR R4,GENCOM ;GENERATE DATA
5363 014526 000003 3 ;PATTERN 3
5364 014530 000400 400 ;400 WORDS
5365
5366 014532 104417 TLOADRK ;LOAD RK REGS
5367 014534 104430 TWAT96 ;WAIT FOR INTERRUPT
5368 014536 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5369
5370 014540 104421 TCHKOP ;CHECK FOR ANY ERROR
5371 014542 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERROR
5372
5373 014544 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5374
5375 014550 000121 RDDATA ;READ DATA
5376 014552 177400 -400 ;ONE FULL SECTOR
5377 014554 070414 IBUFF ;BUFF ADDRESS
5378 014556 007 .BYTE 7 ;SECTOR 7
5379 014557 000 .BYTE 0 ;TRACK 0
5380 014560 000312 312 ;CYLINDER 312
5381
5382 014562 004437 042260 JSR R4,GENCOM
5383 014566 002007 2007 ;CLEAR IBUFF TO ALL ONES
5384 014570 000400 400
5385
5386 014572 104417 TLOADRK ;LOAD RK REGS
5387 014574 104424 TWAT32 ;WAIT FOR INTERRUPT
5388 014576 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
5389
5390 014600 104421 TCHKOP ;CHECK FOR ANY ERRORS
5391 014602 104004 ERROR 4 ; OR 5,6,7 ;REPORT ALL ERRORS
5392
5393 014604 012701 072622 MOV #OBUFF+206,R1 ;CLEAR THE LAST 205 WORDS
5394 014610 012700 000275 MOV #275,R0 ;OF THE OUTPUT BUFFER TO ZERO
5395 014614 005021 1$: CLR (R1)+ ;TO VERIFY THE PARTIAL SECTOR
5396 014616 005300 DEC R0 ;WRITE 0 FILLED THE SECTOR
5397 014620 001375 BNE 1$
5398 014622 004437 042260 JSR R4,GENCOM
5399 014626 100000 100000 ;COMPARE OBUFF & IBUFF.
5400 014630 000400 400 ;ALL 400 WORDS
5401 014632 000413 BR 3$ ;NO ERRORS-EXIT
5402 014634 104015 ERROR 15 ;REPORT FIRST COMPARE ERROR
5403
5404 014636 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5405 014642 005300 2$: DEC R0 ;DECREMENT IT
5406 014644 001406 BEQ 3$ ;IF ZERO-EXIT
5407 014646 004437 042260 JSR R4,GENCOM
5408 014652 040000 40000 ;CONTINUE COMPARE
5409 014654 000402 BR 3$ ;NO MORE ERRORS-EXIT
5410 014656 104016 ERROR 16 ;REPORT NEXT COMPARE ERROR
5411 014660 000770 BR 2$ ;LOOP
5412
5413 014662 3$:
5414 ;:*****
  
```

5415
5416
5417
5418
5419
5420
5421
5422
5423
5424 014662 000004
5425 014664 012737 000012 001262
5426 014672 104416
5427 014674 104003
5428
5429 014676 004437 035674
5430 014702 000123
5431 014704 177400
5432 014706 072414
5433 014710 017
5434 014711 000
5435 014712 000312
5436 014714 004437 042260
5437 014720 000004
5438 014722 000400
5439
5440 014724 104417
5441 014726 104430
5442 014730 104002
5443
5444 014732 104421
5445 014734 104004
5446
5447 014736 004437 035674
5448 014742 000121
5449 014744 177675
5450 014746 070414
5451 014750 017
5452 014751 000
5453 014752 000312
5454 014754 004437 042260
5455 014760 002007
5456 014762 000400
5457
5458 014764 104417
5459 014766 104424
5460 014770 104002
5461 014772 104421
5462 014774 104004
5463
5464 014776 012700 072622
5465 015002 012701 000275
5466 015006 012720 177777
5467 015012 005301
5468 015014 001374
5469 015016 004437 042260
5470 015022 100000

:*TEST 41 PARTIAL SECTOR READ DATA
: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.
:*****
TST41: SCOPE
MOV #10.,\$TIMES ;:DO 10. ITERATIONS
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 17 ;:SECTOR 17
.BYTE 0 ;:TRACK 0
312 ;:CYLINDER 312
JSR R4,GENCOM ;:GENERATE DATA
4 ;:PATTERN 4
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 ;:RDDATA
-103 ;:-103 WORDS
IBUFF ;:IBUFF IS BUFF ADDRESS
.BYTE 17 ;:SECTOR 17
.BYTE 0 ;:TRACK 0
312 ;:CYLINDER 312
JSR R4,GENCOM ;:CLEAR IBUFF
2007
400
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
1\$: MOV #OBUFF+206,R0 ;:AFTER THE LAST 205 WORDS OF
MOV #275,R1 ;:THE OUTPUT BUFFER TO ALL ONES.
MOV #-1,(R0)+ ;:THESE SHOULD ALL BE ONES IN
DEC R1 ;:IBUFF BECAUSE THE PARTIAL
BNE 1\$;:READ FILLED ONLY 103 WORDS.
JSR R4,GENCOM ;:GO COMPARE IBUFF & OBUFF
100000

```

5471 015024 000400          400          :ALL 400 WORDS
5472 015026 000413          BR          3$          :NO ERRORS-EXIT
5473 015030 104015          ERROR       15          :REPORT FIRST COMPARE ERROR
5474
5475 015032 013700 001634          MOV          ERRMT,RO    :GET ERROR LIMIT
5476 015036 005300          64$: DEC          RO      :DECREMENT COUNT
5477 015040 001406          BEQ          65$         :IF ZERO - EXIT
5478 015042 004437 042260          JSR          R4,GENCOM   :CONTINUE DATA COMPARE
5479 015046 040000          40000
5480 015050 000402          BR          65$         :NO MORE ERRORS - EXIT
5481 015052 104016          ERROR       16          :REPORT NEXT ERROR
5482 015054 000770          BR          64$         :LOOP
5483 01,056
5484
5485 015056          3$:
5486 :*****
5487 :*TEST 42          WRITE DATA WITH NON-EXISTENT MEMORY
5488 :*
5489 :*          ISSUE A WRITE DATA OF 1 WORD USING ADDRESS 776000.
5490 :*          MAKE SURE NON-EXISTENT MEMORY SETS.
5491 :*
5492 :*****
5493 015056 000004          TST42: SCOPE
5494 015060 012737 000012 001262          MOV          #10.,$TIMES  ;;DO 10. ITERATIONS
5495 015066 104416          TSSINIT
5496 015070 104003          ERROR       3          :CLEAR SUBSYSTEM
5497 :BAD INIT ERROR
5498 015072 004437 035674          JSR          R4,LRLOAD   :LOAD 'L' REG
5499 015076 001523          BA16!BA17!WRDATA       :BA16 & 17 SET WITH WRITE DATA
5500 015100 177777          -1          :WORD COUNT OF 1
5501 015102 176000          176000      :BUFF ADDRESS=10 PAGE BASE
5502 015104 013          .BYTE       13         :SECT 13
5503 015105 000          .BYTE       0          :TRACK 0
5504 015106 000312          312         :CYLINDER 312
5505
5506 015110 104417          TLOADRK
5507 015112 104430          TWAT96      :LOAD RK REGS
5508 015114 104002          ERROR       2          :WAIT FOR INTERRUPT
5509 015116 104422          TCHKWE
5510 015120 000000          0          :TO SLOW/NOT COMPLETE ERROR
5511 015122 000000          0          :CHECK OPERATION WITH ERROR
5512 015124 000040          NEMERR
5513 015126 104004          ERROR       4 ;OR5,6,7  :NON-EXISTENT MEMORY ERROR
5514 015130 012737 054061 001450          MOV          #EM11A,EM10N :REPORT ANY DISCREPENCIES
5515 015136 012737 050560 061002          MOV          #OPER42,DF010A :SET UP ERROR MESSAGE
5516 015144 113700 001541          MOV          T.CS1+1,RO   :WITH SUPPORT MESSAGE
5517 015150 042700 177774          MOV          #177774,RO   :GET UPPER CS1
5518 015154 022700 000003          BIC          #3,RO        :CLEAR UNUSED BITS
5519 015160 001406          CMP          #3,RO        :TEST IF BOTH UPPER BUS BITS SET
5520 015162 010037 001204          BEQ          1$          :YES - SKIP
5521 015166 012737 000003 001202          MOV          RO,$REG11    :SET UP FOR ERROR REPORT
5522 015174 104010          MOV          #3,$REG10
5523 015176 022737 176002 001544 1$: CMP          #176002,T.BA  :TEST IF BUSS ADDRESS LOW OKAY
5524 015204 001412          BEQ          2$          :YES - SKIP
5525 015206 012737 054033 001450          MOV          #EM11,EM10N  :SET UP MESSGAE
5526 015214 012737 176002 001202          MOV          #176002,$REG10 :STORE VALUE FOR REPORT
    
```

```

5527 015222 013737 001544 001204      MOV    T.BA,$REG11
5528 015230 104010                      ERROR  10
5529
5530 015232 005737 001542      2$:   TST    T.WC          ;TEST IF WORD COUNT CORRECT
5531 015236 001411                      BEQ    3$              ;YES - SKIP
5532 015240 012737 054006 001450      MOV    #EM10,EM10N    ;SET UP MESSAGE
5533 015246 005037 001202                      CLR    $REG10
5534 015252 013737 001542 001204      MOV    T.WC,$REG11
5535 015260 104010                      ERROR  10

```

```

5536
5537 015262      3$:
5538
5539

```

```

5540      ;*****
5541      ;*TEST 43      READ DATA WITH NON-EXISTENT MEMORY
5542      ;*
5543      ;*      ISSUE A READ DATA OF 1 WORD USING ADDRESS 776000.
5544      ;*      MAKE SURE NON-EXISTENT MEMORY SETS.
5545      ;*****

```

```

5546 015262 000004      TST43: SCOPE
5547 015264 012737 000012 001262      MOV    #10.,$TIMES    ;;DO 10. ITERATIONS
5548 015272 104416                      TSSINIT                ;CLEAR SUBSYSTEM
5549 015274 104003                      ERROR  3                ;BAD INIT ERROR
5550
5551 015276 004437 035674      JSR    R4,LRLOAD      ;LOAD 'L' REG
5552 015302 001521                      BA16!BA17!RDDATA      ;BA16 & 17 WITH READ DATA
5553 015304 177777                      -1                      ;WORD COUNT OF 1
5554 015306 176000                      176000                 ;BUFF ADDRESS=10 PAGE BASE
5555 015310 013                      .BYTE 13                ;SECTOR 13
5556 015311 000                      .BYTE 0                 ;TRACK 0
5557 015312 000312                      312                     ;CYL 312
5558

```

```

5559 015314 104417      TLOADRK                ;LOAD RK REGS
5560 015316 104430      TWAT96                 ;WAIT FOR INTERRUPT
5561 015320 104002      ERROR  2                ;TO SLOW/NOT COMPLETE ERROR
5562 015322 104422      TCHKWE                 ;CHECK OPERATION WITH ERRORS
5563 015324 000000      0
5564 015326 000000      0

```

```

5565 015330 000040      NEMERR                ;NON-EXISTENT MEMORY ERROR
5566 015332 104004      ERROR  4; OR 5,6,7    ;REPORT ALL DISCREPANCIES
5567 015334 012737 054061 001450      MOV    #EM11A,EM10N   ;SET MESSAGE
5568 015342 012737 050634 061002      MOV    #OPER43,DF010A ;SET SUPPORT MESSAGE
5569 015350 113700 001541                      MOV    T.CS1+1,R0      ;GET UPPER CS1
5570 015354 042700 177774                      BIC    #177774,R0      ;CLEAR UNWANTED BITS
5571 015360 022700 000003                      CMP    #3,R0           ;TEST BOTH BUS 16 & 17 SET
5572 015364 001406                      BEQ    1$              ;YES - SKIP
5573 015366 012737 000003 001202      MOV    #3,$REG10      ;SET VALUES FOR REPORT
5574 015374 010037 001204                      MOV    R0,$REG11
5575 015400 104010                      ERROR  10

```

```

5576
5577 015402 022737 176002 001544 1$:   CMP    #176002,T.BA   ;TEST IF BUS ADDRESS CORRECT
5578 015410 001412                      BEQ    2$              ;YES - SKIP
5579 015412 012737 054033 001450      MOV    #EM11,EM10N    ;SET MESSAGE
5580 015420 012737 176002 001202      MOV    #176002,$REG10 ;SET VALUES FOR REPORT
5581 015426 013737 001544 001204      MOV    T.BA,$REG11
5582 015434 104010                      ERROR  10

```



```

5583
5584 015436 005737 001542      2$:  TST      T.WC      ;TEST IF WORD COUNT CORRECT
5585 015442 001411              BEQ      3$          ;YES - SKIP
5586 015444 012737 054006 001450  MOV     #EM10,EM10N ;SET MESSAGE
5587 015452 005037 001202              CLR     $REG10      ;SET VALUES
5588 015456 013737 001542 001204  MOV     T.WC,$REG11
5589 015464 104010              ERROR   10
  
```

```

5590
5591 015466      3$:
5592
5593  ::*****
5594  :*TEST 44      UNIBUS PARITY ERROR
5595  :*
5596  :*      INITIALIZE A MEMORY LOCATION WITH BAD
5597  :*      PARITY. ISSUE A WRITE DATA OF 400 WORDS
5598  :*      STARTING AT A LOCATION 112 WORDS BEFORE
5599  :*      THE LOCATION WITH BAD PARITY. MAKE SURE
5600  :*      THAT UNIBUS PARITY ERROR SETS.
5601  :*
5602  :*      NOTE: THIS TEST IS ONLY EXECUTED IF
5603  :*      MEMORY PARITY OPTION EXISTS FOR
5604  :*      BUFFER. IT WILL NOT BE EXECUTED
5605  :*      ON 'ECC' TYPE MEMORY.
5606  :*
5607  :*
5608  :*
  
```

```

5609 015466 000004      TST44: SCOPE
5610 015470 012737 000012 001262  MOV     #10.,$TIMES ;:DO 10. ITERATIONS
5611 015476 104416              TSSINIT ;:CLEAR SUBSYSTEM
5612 015500 104003              ERROR   3          ;:BAD INIT ERROR
5613
5614 015502 032737 000100 001664  BIT     #PARPRE,OPTFLG ;:TEST IF PARITY OPTION PRESENT
5615 015510 001013              BNE     1$          ;:YES-SKIP
5616 015512 032737 000004 001664  BIT     #MEMPYB,OPTFLG ;:TEST IF PARITY OPTION REPORTED
5617 015520 001005              BNE     25$         ;:YES-SKIP TO EXIT
5618 015522 104401 051362              TYPE   ,OPR010      ;:PRINT BYPASS MESSAGE
5619 015526 052737 000004 001664  BIS     #MEMPYB,OPTFLG ;:SET OPTION REPORTED BIT
5620 015534 000137 016326      25$:  JMP     4$          ;:SKIP TO EXIT
5621
  
```

```

5622 015540      1$:
5623 015540 004437 035674      JSR     R4,LRLOAD   ;:LOAD 'L' REGS
5624 015544 000123              WRDATA ;:WRDATA
5625 015546 177400              -400   ;:-400 WORDS
5626 015550 072414              OBUFF  ;:OBUFF IS BUFF ADDRESS
5627 015552 010              .BYTE  10          ;:SECTOR 10
5628 015553 000              .BYTE  0           ;:TRACK 0
5629 015554 000312              312     ;:CYLINDER 312
5630
5631 015556 032737 002000 001664  BIT     #CP1170,OPTFLG ;:TEST IF 11/70
5632 015564 001002              BNE     3$          ;:YES - SKIP
5633 015566 005077 164100              CLR     @CSRPTR     ;:CLEAR PARITY IE
5634
  
```

```

5635 015572 004437 042260      3$:  JSR     R4,GENCOM   ;:GENERATE DATA
5636 015576 000007              7        ;:PATTERN 7
5637 015600 000400              400     ;:400 WORDS
5638
  
```

```

5639 015602 012746 000340      MOV      #PR7,-(SP)      ;PUT PRIORITY 7 ON STACK
5640 015606 012746 015614      MOV      #10$,-(SP)     ;PUT ADDRESS ON STACK
5641 015612 000002                RTI                    ;SET PRI
5642 015614                10$:
5643
5644 015614 012737 100200 072640      MOV      #100200,OBUF+224 ;SET WORD IN BUFFER
5645 015622 032737 002000 001664      BIT      #CP1170,OPTFLG  ;TEST IF 11/70
5646 015630 001011                BNE      5$           ;YES - SKIP
5647 015632 012777 000004 164032      MOV      #BIT2,@CSRPTR  ;SET WRITE WRONG PARITY BIT
5648 015640 012737 100200 072640      MOV      #100200,OBUF+224 ;SET BAD PARITY IN MEMORY
5649 015646 012777 000001 164016      MOV      #BIT0,@CSRPTR  ;CLEAR CONTROL BIT, SET IE BIT
5650
5651 015654 013746 001622                5$:      MOV      RKPRI,-(SP)    ;SET OLD PRIORITY
5652 015660 012746 015666                MOV      #11$,-(SP)    ;SET ADDRESS
5653 015664 000002                RTI                    ;RESTORE PRI
5654 015666 013704 001672                11$:      MOV      CSRPTR,R4     ;SET R4 WITH CSR POINTER
5655 015672 005000                CLR      R0            ;SET R0 FOR COUNTER
5656 015674 012701 001662                MOV      #INTSET,R1    ;SET R1 FOR POINTER TO INTERRUPT FLAG
5657 015700 012777 070400 163712      MOV      #SPCHLR,@RKVEC ;SET UP INTERRUPT VECTORS FOR
5658 015706 012777 070410 164002      MOV      #SPCPAR,@MMVECA ;RK611 AND PARITY ERROR
5659 015714 012737 016302 001264      MOV      #2$, $ESCAPE  ;SET UP ESCAPE FOR ERROR
5660 015722 104417                TLOADRK              ;LOAD RK REGS
5661 015724 032737 002000 001664      BIT      #CP1170,OPTFLG ;TEST IF 11/70
5662 015732 001434                BEQ      45$          ;NO - SKIP
5663 015734 012737 000016 177746      MOV      #16,177746   ;SET TO DISABLE CACHE AND UNIBUS ERROR
5664 015742 012777 000000 163740      MOV      #0,@KWLADD   ;TURN OFF CLOCK INTERRUPTS
5665 015750 012777 000000 163166      MOV      #0,@$TKS     ;TURN OFF KEYBOARD
5666 015756 012737 170000 177750      MOV      #170000,177750 ;SET FOR ERROR FORCE
5667
5668
5669 015764 105711                40$:      TSTB     (R1)          ;LOOP TO WAIT FOR INTERRUPT OR ABORT
5670 015766 003005                BGT      43$          ;WAIT. THE CODE BETWEEN THE STARS IS SET
5671 015770 005300                DEC      R0            ;SET UP SO ALL BYTES HAVE PARITY OF 1.
5672 015772 100774                BMI      40$          ;IF THIS CODE IS CHANGED, REMEMBER ALL
5673 015774 000240                NOP                    ;BYTES MUST HAVE AN EVEN NUMBER OF
5674 015776 003372                BGT      40$          ;BITS.
5675 016000 000240                NOP
5676 016002 005014                43$:      CLR      (R4)         ;CLEAR ERROR FORCE
5677
5678
5679 016004 005037 177746                CLR      177746       ;ENABLE CACHE
5680 016010 012777 000100 163672      MOV      #BIT6,@KWLADD ;TURN ON CLOCK INTERRUPTS
5681 016016 012777 000100 163120      MOV      #100,@$TKS   ;TURN ON KEYBOARD INTERRUPTS
5682 016024 104430                45$:      TWAT96              ;WAIT FOR INTERRUPT
5683 016026 000414                BR       46$          ;TO SLOW/NOT COMPLETE ERROR
5684 016030 032737 002000 001664      BIT      #CP1170,OPTFLG ;TEST IF 11/70
5685 016036 001024                BNE      48$          ;YES - SKIP
5686 016040 005077 163626                CLR      @CSRPTR     ;ELSE CLEAR CSR
5687 016044 005037 072640                CLR      OBUF+224    ;CLEAR THE BAD PARITY WORD
5688 016050 012777 000001 163614      MOV      #1,@CSRPTR   ;SET PARITY DETECT AGAIN
5689 016056 000414                BR       48$          ;SKIP
5690
5691 016060 032737 002000 001664 46$:      BIT      #CP1170,OPTFLG ;TEST IF 11/70
5692 016066 001007                BNE      47$          ;YES - SKIP
5693 016070 005077 163576                CLR      @CSRPTR     ;CLEAR CSR
5694 016074 005037 072640                CLR      OBUF+224    ;CLEAR BAD PARITY WORD

```

```

5695 016100 012777 000001 163564 MOV #1,@CSRPTR ;SET UP PARITY DETECT AGAIN
5696 016106 104002 47$: ERROR 2 ;REPORT TO SLOW ERROR
5697 016110 48$: TCHKWE ;CHECK OPERATION WITH ERROR
5698 016110 104422 0
5699 016112 000000 0
5700 016114 000000 UPERR ;UNIBUS PARITY ERROR
5701 016116 000100 ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
5702 016120 104004
5703
5704 016122 005037 001264 CLR $ESCAPE ;CLEAR ESCAPE
5705 016126 012737 050707 061002 MOV #OPER44,DF010A ;SET MESSAGES FOR REPORT
5706 016134 012737 054033 001450 MOV #EM11,EM10N
5707 016142 023727 001544 072642 CMP T.BA,#OBUFF+226 ;CHECK IF BA IN RANGE
5708 016150 103010 BHIS 14$ ;NOT TO LOW - SKIP
5709 016152 012737 072642 001202 MOV #OBUFF+226,$REG10 ;SET VALUES FOR REPORT
5710 016160 013737 001544 001204 MOV T.BA,$REG11
5711 016166 104010 ERROR 10
5712 016170 000413 BR 16$
5713
5714 016172 023727 001544 072646 14$: CMP T.BA,#OBUFF+232 ;CHECK IF BA IN RANGE
5715 016200 101407 BLOS 16$ ;YES - SKIP
5716 016202 012737 072646 001202 MOV #OBUFF+232,$REG10 ;SET VALUES
5717 016210 012737 001544 001204 MOV #T.BA,$REG11
5718 016216 104010 ERROR 10
5719
5720 016220 012737 054006 001450 16$: MOV #EM10,EM10N ;SET MESSAGE
5721 016226 023727 001542 177513 CMP T.WC,#-265 ;CHECK IF WORD COUNT WITHIN RANGE
5722 016234 103007 BHIS 20$ ;YES - SKIP
5723 016236 012737 177513 001202 MOV #-265,$REG10 ;SET VALUES
5724 016244 013737 001542 001204 MOV T.WC,$REG11
5725 016252 104010 ERROR 10
5726
5727 016254 023727 001542 177515 20$: CMP T.WC,#-263 ;STILL CHECKING IF WC IN RANGE
5728 016262 101407 BLOS 2$ ;YES - SKIP
5729 016264 012737 177515 001202 MOV #-263,$REG10 ;SET VALUES
5730 016272 013737 001542 001204 MOV T.WC,$REG11
5731 016300 104010 ERROR 10
5732
5733 016302 2$: CLR $ESCAPE ;REV 006
5734 016302 005037 001264 MOV #INTHLR,@RKVEC ;RESET INT. VECTORS FOR RK06
5735 016306 012777 034442 163304 JSR PC,OPTTST
5736 016314 004737 034522 MOV #100,@$TKS ;INSURE KEYBOARD IS ENABLED
5737 016320 012777 000100 162616
5738
5739 016326 4$:

```

```

.SBTTL **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.
:*

```

5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750

```
*****
5751
5752 016326 000004
5753 016330 012737 000012 001262 TST45: SCOPE
5754 016336 104416 MOV #10.,$TIMES ;:DO 10. ITERATIONS
5755 016340 104003 TSSINIT ;:CLEAR SUBSYSTEM
ERROR 3 ;:BAD INIT ERROR
5756
5757 016342 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
5758 016346 000123 WRDATA ;:WRDATA
5759 016350 177000 -1000 ;:-1000 WORDS
5760 016352 072414 OBUFF ;:OBUFF IS BUFF ADDRESS
5761 016354 000 .BYTE 0 ;:SECTOR 0
5762 016355 000 .BYTE 0 ;:TRACK 0
5763 016356 000312 312 ;:CYLINDER 312
5764
5765 016360 004437 042260 JSR R4,GENCOM ;:GENERATE DATA
5766 016364 000015 15 ;:PATTERN 15
5767 016366 001000 1000 ;:1000 WORDS
5768
5769 016370 104417 TLOADRK ;:LOAD RK REGS
5770 016372 104430 TWAT96 ;:WAIT FOR INTERRUPT
5771 016374 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
5772
5773 016376 104421 TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
5774 016400 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5775
5776 016402 004437 042260 JSR R4,GENCOM ;:CLEAR Ibuff
5777 016406 002007 2007 ;:TO ALL 1'S
5778 016410 001000 1000
5779
5780 016412 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
5781 016416 000121 RDDATA ;:RDDATA
5782 016420 177400 -400 ;:-400 WORDS
5783 016422 070414 Ibuff ;:IBUFF IS BUFF ADDRESS
5784 016424 000 .BYTE 0 ;:SECTOR 0
5785 016425 000 .BYTE 0 ;:TRACK 0
5786 016426 000312 312 ;:CYLINDER 312
5787
5788 016430 104417 TLOADRK ;:LOAD RK REGS
5789 016432 104424 TWAT32 ;:WAIT FOR INTERRUPT
5790 016434 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
5791
5792 016436 104421 TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
5793 016440 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5794
5795 016442 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
5796 016446 000121 RDDATA ;:RDDATA
5797 016450 177400 -400 ;:-400 WORDS
5798 016452 071414 Ibuff+1000 ;:IBUFF+1000 IS BUFF ADDRESS
5799 016454 001 .BYTE 1 ;:SECTOR 1
5800 016455 000 .BYTE 0 ;:TRACK 0
5801 016456 000312 312 ;:CYLINDER 312
5802
5803 016460 104417 TLOADRK ;:LOAD RK REGS
5804 016462 104424 TWAT32 ;:WAIT FOR INTERRUPT
5805 016464 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
5806
```

```
5807 016466 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5808 016470 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5809
5810 016472 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5811 016476 100000 100000 ;
5812 016500 001000 1000 ;1000 WORDS
5813 016502 000413 BR 2$ ;NO MISCOMPARES-EXIT
5814 016504 104015 ERROR 15 ;REPORT FIRST ERROR
5815
5816 016506 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5817 016512 005300 64$: DEC R0 ;DECREMENT COUNT
5818 016514 001406 BEQ 65$ ;IF ZERO - EXIT
5819 016516 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5820 016522 040000 40000
5821 016524 000402 BR 65$ ;NO MORE ERRORS - EXIT
5822 016526 104016 ERROR 16 ;REPORT NEXT ERROR
5823 016530 000770 BR 64$ ;LOOP
5824 016532
5825
5826 016532 2$:
5827 *****
5828 *TEST 46 TWO SECTOR WRITE DATA (PART 2)
5829 *****
5830 *
5831 * ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312,
5832 * TRACK 0, SECTOR 23. READ DATA BACK ONE SECTOR
5833 * AT A TIME AND MAKE SURE A MID-TRANSFER
5834 * SEEK DID NOT TAKE PLACE.
5835 *****
5836 016532 000004 TST46: SCOPE
5837 016534 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
5838 016542 104416 TSSINIT ;CLEAR SUBSYSTEM
5839 016544 104003 ERROR 3 ;BAD INIT ERROR
5840
5841 016546 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5842 016552 000123 WRDATA ;WRDATA
5843 016554 177000 -1000 ;-1000 WORDS
5844 016556 072414 OBUFF ;OBUFF IS BUFF ADDRESS
5845 016560 023 .BYTE 23 ;SECTOR 23
5846 016561 000 .BYTE 0 ;TRACK 0
5847 016562 000312 312 ;CYLINDER 312
5848
5849 016564 004437 042260 JSR R4,GENCOM ;GENERATE DATA
5850 016570 000016 16 ;PATTERN 16
5851 016572 001000 1000 ;1000 WORDS
5852
5853 016574 104417 TLOADRK ;LOAD RK REGS
5854 016576 104430 TWAT96 ;WAIT FOR INTERRUPT
5855 016600 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5856
5857 016602 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5858 016604 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5859
5860 :
5861 : IF THE TRACK ADDRESS AT THE END OF OPERATION IS IN ERROR
5862 : THE CONTROLLER DID A MID-TRANSFER SEEK AS THOUGH IT
: WERE IN 24(8) SECTORS PER TRACK MODE.
:
```

```
5863
5864 016606 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5865 016612 000121 RDDATA ;RDDATA
5866 016614 177400 -400 ;-400 WORDS
5867 016616 070414 Ibuff ;IBUFF IS BUFF ADDRESS
5868 016620 023 .BYTE 23 ;SECTOR 23
5869 016621 000 .BYTE 0 ;TRACK 0
5870 016622 000312 312 ;CYLINDER 312
5871
5872 016624 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
5873 016630 002007 2007
5874 016632 001000 1000
5875
5876 016634 104417 TLOADRK ;LOAD RK REGS
5877 016636 104424 TWAT32 ;WAIT FOR INTERRUPT
5878 016640 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5879
5880 016642 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5881 016644 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5882
5883 016646 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
5884 016652 000121 RDDATA ;RDDATA
5885 016654 177400 -400 ;-400 WORDS
5886 016656 071414 Ibuff+1000 ;IBUFF+1000 IS BUFF ADDRESS
5887 016660 024 .BYTE 24 ;SECTOR 24
5888 016661 000 .BYTE 0 ;TRACK 0
5889 016662 000312 312 ;CYLINDER 312
5890
5891 016664 104417 TLOADRK ;LOAD RK REGS
5892 016666 104424 TWAT32 ;WAIT FOR INTERRUPT
5893 016670 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5894
5895 016672 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
5896 016674 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5897
5898 016676 004437 042260 JSR R4,GENCOM ;COMPARE DATA
5899 016702 100000 100000 ;1000 WORDS
5900 016704 001000 1000 ;NO ERRORS-SKIP
5901 016706 000413 BR 15 ;REPORT FIRST ERROR
5902 016710 104015 ERROR 15
5903
5904 016712 013700 001634 MOV ERRLMT,RO ;GET ERROR LIMIT
5905 016716 005300 64$: DEC RO ;DECREMENT COUNT
5906 016720 001406 BEQ 65$ ;IF ZERO - EXIT
5907 016722 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5908 016726 040000 40000
5909 016730 000402 BR 65$ ;NO MORE ERRORS - EXIT
5910 016732 104016 ERROR 16 ;REPORT NEXT ERROR
5911 016734 000770 BR 64$ ;LOOP
5912 016736 65$:
5913 016736 1$:
5914
5915 *****
5916 *TEST 47 TWO SECTOR WRITE DATA (PART 3)
5917 *
5918 * ISSUE A WRITE DATA OF 401 WORDS TO CYLINDER 312,
* TRACK 0, SECTOR 10. READ DATA BACK ONE SECTOR AT
```

```
5919          : *      A TIME AND CHECK ZERO FILL OF SECOND SECTOR.
5920          : *
5921          : *
5922 016736 000004          : *****
5923 016740 012737 000012 001262 TST47: SCOPE
5924 016746 104416          MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
5925 016750 104003          TSSINIT          ;CLEAR SUBSYSTEM
5926          ERROR      3          ;BAD INIT ERROR
5927 016752 004437 035674          JSR      R4,LRLOAD      ;LOAD 'L' REGS
5928 016756 000123          WRDATA          ;WRDATA
5929 016760 177377          -401          ; -401 WORDS
5930 016762 072414          OBUFF          ;OBUFF IS BUFF ADDRESS
5931 016764      010          .BYTE      10          ;SECTOR 10
5932 016765      000          .BYTE      0          ;TRACK 0
5933 016766 000312          312          ;CYLINDER 312
5934
5935 016770 004437 042260          JSR      R4,GENCOM      ;GENERATE DATA
5936 016774 000002          2          ;PATTERN 2
5937 016776 001000          1000          ;1000 WORDS
5938
5939 017000 104417          TLOADRK          ;LOAD RK REGS
5940 017002 104430          TWAT96          ;WAIT FOR INTERRUPT
5941 017004 104002          ERROR      2          ;TO SLOW/NOT COMPLETE ERROR
5942
5943 017006 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
5944 017010 104004          ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5945          CLEAR LAST 377 WORDS OF OBUFF FOR EXPECTED ZEROS FROM ZERO FILL
5946 017012 012700 073416          MOV      #OBUFF+1002,R0 ;GET STARTING ADDRESS TO BE CLEARED
5947 017016 012701 000377          MOV      #377,R1      ;NUMBER OF WORDS
5948 017022 005020          CLR      (R0)+        ;CLEAR WORD
5949 017024 005301          DEC      R1          ;DEC COUNT
5950 017026 001375          BNE      1$          ;LOOP UNTIL COUNT ZERO
5951 017030 004437 042260          JSR      R4,GENCOM      ;CLEAR Ibuff TO ONES
5952 017034 002007          2007
5953 017036 001000          1000
5954
5955 017040 004437 035674          JSR      R4,LRLOAD      ;LOAD 'L' REGS
5956 017044 000121          RDDATA          ;RDDATA
5957 017046 177400          -400          ; -400 WORDS
5958 017050 070414          Ibuff          ;IBUFF IS BUFF ADDRESS
5959 017052      010          .BYTE      10          ;SECTOR 10
5960 017053      000          .BYTE      0          ;TRACK 0
5961 017054 000312          312          ;CYLINDER 312
5962
5963 017056 104417          TLOADRK          ;LOAD RK REGS
5964 017060 104424          TWAT32          ;WAIT FOR INTERRUPT
5965 017062 104002          ERROR      2          ;TO SLOW/NOT COMPLETE ERROR
5966
5967 017064 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
5968 017066 104004          ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
5969 017070 004437 035674          JSR      R4,LRLOAD      ;LOAD 'L' REGS
5970 017074 000121          RDDATA          ;RDDATA
5971 017076 177400          -400          ; -400 WORDS
5972 017100 071414          Ibuff+1000      ;IBUFF+1000 IS BUFF ADDRESS
5973 017102      011          .BYTE      11          ;SECTOR 11
5974 017103      000          .BYTE      0          ;TRACK 0
```

```
5975 017104 000312 312 ;CYLINDER 312
5976
5977 017106 104417 TLOADRK ;LOAD RK REGS
5978 017110 104424 TWAT32 ;WAIT FOR INTERRUPT
5979 017112 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
5980
5981 017114 004437 042260 JSR R4,GENCOM ;DATA COMPARE
5982 017120 100000 100000
5983 017122 001000 1000 ;1000 WORDS
5984 017124 000413 BR 2$ ;NO ERROR-SKIP
5985 017126 104015 ERROR 15 ;REPORT FIRST ERROR
5986
5987 017130 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
5988 017134 005300 64$: DEC R0 ;DECREMENT COUNT
5989 017136 001406 BEQ 65$ ;IF ZERO - EXIT
5990 017140 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
5991 017144 040000 40000
5992 017146 000402 BR 65$ ;NO MORE ERRORS - EXIT
5993 017150 104016 ERROR 16 ;REPORT NEXT ERROR
5994 017152 000770 BR 64$ ;LOOP
5995 017154 65$:
5996
5997 017154 2$:
5998
5999
6000
6001
6002
6003
6004
6005
6006
6007 017154 000004 TST50: SCOPE
6008 017156 012737 000012 001262 MOV #10, $TIMES ;:DO 10. ITERATIONS
6009 017164 104416 TSSINIT ;:CLEAR SUBSYSTEM
6010 017166 104003 ERROR 3 ;:BAD INIT ERROR
6011
6012 017170 004437 035674 JSR R4,LRLOAD ;:LOAD 'L' REGS
6013 017174 000123 WRDATA ;:WRDATA
6014 017176 177000 -1000 ;:-1000 WORDS
6015 017200 072414 OBUFF ;:OBUFF IS BUFF ADDRESS
6016 017202 025 .BYTE 25 ;:SECTOR 25
6017 017203 000 .BYTE 0 ;:TRACK 0
6018 017204 000312 312 ;:CYLINDER 312
6019
6020 017206 004437 042260 JSR R4,GENCOM ;:GENERATE DATA
6021 017212 000003 3 ;:PATTERN 3
6022 017214 001000 1000 ;:1000 WORDS
6023
6024 017216 104417 TLOADRK ;:LOAD RK REGS
6025 017220 104430 TWAT96 ;:WAIT FOR INTERRUPT
6026 017222 104002 ERROR 2 ;:TO SLOW/NOT COMPLETE ERROR
6027
6028 017224 104421 TCHKOP ;:CHECK OPERATION FOR ANY ERRORS
6029 017226 104004 ERROR 4 ;OR 5, 6, 7, 10 ;:REPORT ALL ERRORS
6030 ; A TRACK ERROR PRINTED OUT AT THE END OF THE OPERATION INDICATES A
```



```
6031 ; MID-TRANSFER HEAD SWITCH DID NOT OCCUR.
6032 017230 004437 042260 JSR R4,GENCOM
6033 017234 002007 2007
6034 017236 001000 1000
6035
6036 017240 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6037 017244 000121 RDDATA ;RDDATA
6038 017246 177400 -400 ;-400 WORDS
6039 017250 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6040 017252 025 .BYTE 25 ;SECTOR 25
6041 017253 000 .BYTE 0 ;TRACK 0
6042 017254 000312 312 ;CYLINDER 312
6043
6044 017256 104417 TLOADRK ;LOAD RK REGS
6045 017260 104425 TWAT48 ;WAIT FOR INTERRUPT
6046 017262 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6047
6048 017264 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6049 017266 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6050
6051 017270 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6052 017274 000121 RDDATA ;RDDATA
6053 017276 177400 -400 ;-400 WORDS
6054 017300 071414 Ibuff+1000 ;IBUFF+1000 IS BUFF ADDRESS
6055 017302 000 .BYTE 0 ;SECTOR 0
6056 017303 001 .BYTE 1 ;TRACK 1
6057 017304 000312 312 ;CYLINDER 312
6058
6059 017306 104417 TLOADRK ;LOAD RK REGS
6060 017310 104425 TWAT48 ;WAIT FOR INTERRUPT
6061 017312 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6062
6063 017314 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6064 017316 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6065
6066 017320 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6067 017324 100000 100000
6068 017326 001000 1000 ;1000 WORDS
6069 017330 000413 BR 1$ ;NO ERRORS-SKIP
6070 017332 104015 ERROR 15 ;REPORT FIRST ERROR
6071
6072 017334 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
6073 017340 005300 64$: DEC R0 ;DECREMENT COUNT
6074 017342 001406 BEQ 65$ ;IF ZERO - EXIT
6075 017344 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6076 017350 040000 40000
6077 017352 000402 BR 65$ ;NO MORE ERRORS - EXIT
6078 017354 104016 ERROR 16 ;REPORT NEXT ERROR
6079 017356 000770 BR 64$ ;LOOP
6080 017360 65$:
6081
6082 017360 1$:
6083 *****
6084 ;*TEST 51 MID-TRANSFER SEEK ON WRITE (PART 2)
6085 ;*
6086 ;* ISSUE A WRITE DATA OF 1000 WORDS TO CYLINDER 312.
```

```
6087      *      TRACK 2, SECTOR 25.  READ DATA BACK ONE SECTOR
6088      *      AT A TIME AND MAKE SURE A MID-TRANSFER SEEK
6089      *      DID TAKE PLACE.
6090      *
6091      *      *****
6092 017360 000004      TST51: SCOPE
6093 017362 012737 000012 001262  MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
6094 017370 104416      TSSINIT      ;CLEAR SUBSYSTEM
6095 017372 104003      ERROR      3      ;BAD INIT ERROR
6096
6097 017374 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
6098 017400 000123      WRDATA      ;WRDATA
6099 017402 177000      -1000      ;-1000 WORDS
6100 017404 072414      OBUFF      ;OBUFF IS BUFF ADDRESS
6101 017406      025      .BYTE      25      ;SECTOR 25
6102 017407      002      .BYTE      2      ;TRACK 2
6103 017410 000312      312      ;CYLINDER 312
6104
6105 017412 004437 042260      JSR      R4,GENCOM      ;GENERATE DATA
6106 017416 000004      4      ;PATTERN 4
6107 017420 001000      1000      ;1000 WORDS
6108
6109 017422 104417      TLOADRK      ;LOAD RK REGS
6110 017424 104430      TWAT96      ;WAIT FOR INTERRUPT
6111 017426 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
6112      *      A CYLINDER ERROR REPORTED AT THE END OF THE OPERATION INDICATES A
6113      *      MID-TRANSFER SEEK DID NOT OCCUR.
6114 017430 104421      TCHKOP      ;CHECK OPERATION FOR ANY ERRORS
6115 017432 104004      ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6116
6117 017434 004437 042260      JSR      R4,GENCOM      ;CLEAR IBUFF TO ALL ONES
6118 017440 002007      2007
6119 017442 001000      1000
6120
6121 017444 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
6122 017450 000121      RDDATA      ;RDDATA
6123 017452 177400      -400      ;-400 WORDS
6124 017454 070414      IBUFF      ;IBUFF IS BUFF ADDRESS
6125 017456      025      .BYTE      25      ;SECTOR 25
6126 017457      002      .BYTE      2      ;TRACK 2
6127 017460 000312      312      ;CYLINDER 312
6128
6129 017462 104417      TLOADRK      ;LOAD RK REGS
6130 017464 104425      TWAT48      ;WAIT FOR INTERRUPT
6131 017466 104002      ERROR      2      ;TO SLOW/NOT COMPLETE ERROR
6132
6133 017470 104421      TCHKOP      ;CHECK OPERATION FOR ANY ERRORS
6134 017472 104004      ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6135
6136 017474 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
6137 017500 000121      RDDATA      ;RDDATA
6138 017502 177400      -400      ;-400 WORDS
6139 017504 071414      IBUFF+1000 ;IBUFF+1000 IS BUFF ADDRESS
6140 017506      000      .BYTE      0      ;SECTOR 0
6141 017507      000      .BYTE      0      ;TRACK 0
6142 017510 000313      313      ;CYLINDER 313
```

```
6143
6144 017512 104417      TLOADRK      ;LOAD RK REGS
6145 017514 104425      TWAT48       ;WAIT FOR INTERRUPT
6146 017516 104002      ERROR 2      ;TO SLOW/NOT COMPLETE ERROR
6147
6148 017520 104421      TCHKOP      ;CHECK OPERATION FOR ANY ERRORS
6149 017522 104004      ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6150
6151 017524 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6152 017530 100000      100000
6153 017532 001000      1000        ;1000 WORDS
6154 017534 000413      BR 1$       ;NO MISCOMPARES-SKIP
6155 017536 104015      ERROR 15    ;REPORT 1ST ERROR
6156
6157 017540 013700 001634      MOV  ERRLMT,RO ;GET ERROR LIMIT
6158 017544 005300      64$: DEC RO    ;DECREMENT COUNT
6159 017546 001406      BEQ  65$    ;IF ZERO - EXIT
6160 017550 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6161 017554 040000      40000
6162 017556 000402      BR 65$     ;NO MORE ERRORS - EXIT
6163 017560 104016      ERROR 16   ;REPORT NEXT ERROR
6164 017562 000770      BR 64$     ;LOOP
6165 017564
6166
6167 017564      1$:
6168      ;*****
6169      ;*TEST 52      TWO SECTOR READ DATA (PART 1)
6170      ;*
6171      ;*      ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312,
6172      ;*      TRACK 0, SECTOR 0, VERIFY THAT CORRECT DATA IS
6173      ;*      READ.
6174      ;*
6175      ;*      NOTE: TWO SECTOR WRITE DATA (PART 1) MUST BE
6176      ;*      EXECUTED BEFORE THIS TEST.
6177      ;*
6178      ;*****
6179 017564 000004      TST52: SCOPE
6180 017566 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
6181 017574 104416      TSSINIT     ;CLEAR SUBSYSTEM
6182 017576 104003      ERROR 3    ;BAD INIT ERROR
6183
6184      ;GENERATE SAME DATA AS USED IN TWO SECTOR WRITE DATA (PART 1)
6185
6186      ; GENERATE SAME DATA AS USED IN TWO SECTOR WRITE DATA PART 1
6187 017600 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6188 017604 000015      15        ;PATTERN 15
6189 017606 001000      1000      ;1000 WORDS
6190
6191 017610 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6192 017614 002007      2007
6193 017616 001000      1000
6194
6195 017620 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6196 017624 000121      RDDATA    ;RDDATA
6197 017626 177000      -1000    ;-1000 WORDS
6198 017630 070414      Ibuff    ;IBUFF IS BUFF ADDRESS
```

6199	017632	000		.BYTE	0		:SECTOR 0
6200	017633	000		.BYTE	0		:TRACK 0
6201	017634	000312			312		:CYLINDER 312
6202							
6203	017636	104417		TLOADRK			:LOAD RK REGS
6204	017640	104430		TWAT96			:WAIT FOR INTERRUPT
6205	017642	104002		ERROR	2		:TO SLOW/NOT COMPLETE ERROR
6206							
6207	017644	104421		TCHKOP			:CHECK OPERATION FOR ANY ERRORS
6208	017646	104004		ERROR	4	;OR 5, 6, 7, 10	:REPORT ALL ERRORS
6209							
6210	017650	004437	042260	JSR	R4,GENCOM		:COMPARE DATA
6211	017654	100000			100000		
6212	017656	001000			1000		:1000 WORDS
6213	017660	000413		BR	1\$:NO MISCOMPARES-SKIP
6214	017662	104015		ERROR	15		
6215							
6216	017664	013700	001634	MOV	ERRLMT,RO		:GET ERROR LIMIT
6217	017670	005300		DEC	RO		:DECREMENT COUNT
6218	017672	001406		BEQ	65\$:IF ZERO - EXIT
6219	017674	004437	042260	JSR	R4,GENCOM		:CONTINUE DATA COMPARE
6220	017700	040000			40000		
6221	017702	000402		BR	65\$:NO MORE ERRORS - EXIT
6222	017704	104016		ERROR	16		:REPORT NEXT ERROR
6223	017706	000770		BR	64\$:LOOP
6224	017710						
6225							
6226	017710						
6227							
6228							
6229							
6230							
6231							
6232							
6233							
6234							
6235							
6236							
6237							
6238	017710	000004		TST53:	SCOPE		
6239	017712	012737	000012 001262	MOV	#10.,\$TIMES		::DO 10. ITERATIONS
6240	017720	104416		TSSINIT			:CLEAR SUBSYSTEM
6241	017722	104003		ERROR	3		:BAD INIT ERROR
6242							
6243							:GENERATE SAME DATA AS USED IN TWO SECTOR WRITE (PART 2)
6244							
6245	017724	004437	042260	JSR	R4,GENCOM		:GENERATE DATA
6246	017730	000016			16		:PATTERN 16
6247	017732	001000			1000		:1000 WORDS
6248							
6249	017734	004437	042260	JSR	R4,GENCOM		:CLEAR Ibuff TO ALL ONES
6250	017740	002007			2007		
6251	017742	001000			1000		
6252	017744	004437	035674	JSR	R4,LRLOAD		:LOAD 'L' REGS
6253	017750	000121		RDDATA			:RDDATA
6254	017752	177000			-1000		: -1000 WORDS

```
6255 017754 070414          Ibuff          ;IBUFF IS BUFF ADDRESS
6256 017756          023      .BYTE 23          ;SECTOR 23
6257 017757          000      .BYTE 0           ;TRACK 0
6258 017760 000312          312           ;CYLINDER 312
6259
6260 017762 104417          TLOADRK        ;LOAD RK REGS
6261 017764 104430          TWAT96         ;WAIT FOR INTERRUPT
6262 017766 104002          ERROR 2       ;TO SLOW/NOT COMPLETE ERROR
6263
6264 017770 104421          TCHKOP         ;CHECK OPERATION FOR ANY ERRORS
6265 017772 104004          ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6266
6267 017774 004437 042260    JSR R4,GENCOM  ;COMPARE DATA
6268 020000 100000          100000
6269 020002 001000          1000          ;1000 WORDS
6270 020004 000413          BR 1$         ;NO MISCOMPARES-SKIP
6271 020006 104015          ERROR 15      ;REPORT 1ST ERROR
6272
6273 020010 013700 001634    MOV ERRMT,RO  ;GET ERROR LIMIT
6274 020014 005300          DEC RO        ;DECREMENT COUNT
6275 020016 001406          BEQ 65$      ;IF ZERO - EXIT
6276 020020 004437 042260    JSR R4,GENCOM ;CONTINUE DATA COMPARE
6277 020024 040000          40000
6278 020026 000402          BR 65$      ;NO MORE ERRORS - EXIT
6279 020030 104016          ERROR 16     ;REPORT NEXT ERROR
6280 020032 000770          BR 64$      ;LOOP
6281 020034
6282
6283 020034          1$:
6284
6285          ;*****
6286          ;*TEST 54 TWO SECTOR READ DATA (PART 3)
6287          ;*
6288          ;* ISSUE A READ DATA OF 401 WORDS TO CYLINDER 312,
6289          ;* TRACK 0, SECTOR 10. VERIFY THAT ALL 401 WORDS
6290          ;* ARE PLACED IN MEMORY.
6291          ;*
6292          ;* NOTE: TWO SECTOR WRITE DATA (PART 3) MUST BE
6293          ;* EXECUTED BEFORE THIS TEST.
6294          ;*****
6295 020034 000004          TST54: SCOPE
6296 020036 012737 000012 001262 MOV #10, $TIMES ;DO 10. ITERATIONS
6297 020044 104416          TSSINIT       ;CLEAR SUBSYSTEM
6298 020046 104003          ERROR 3      ;BAD INIT ERROR
6299
6300          ; GENERATE SAME DATA AS USED IN TWO SECTOR WRITE (PART 3)
6301
6302 020050 004437 042260    JSR R4,GENCOM ;GENERATE DATA
6303 020054 000002          2           ;PATTERN 2
6304 020056 000401          401         ;401 WORDS
6305
6306 020060 004437 042260    JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6307 020064 002007          2007
6308 020066 001000          1000
6309
6310 020070 004437 035674    JSR R4,LRLOAD ;LOAD 'L' REGS
```

6311	020074	000121		RDDATA		:RDDATA
6312	020076	177377		-401		:-401 WORDS
6313	020100	070414		IBUFF		:IBUFF IS BUFF ADDRESS
6314	020102	010		.BYTE 10		:SECTOR 10
6315	020103	000		.BYTE 0		:TRACK 0
6316	020104	000312		312		:CYLINDER 312
6317						
6318	020106	104417		TLOADRK		:LOAD RK REGS
6319	020110	104430		TWAT96		:WAIT FOR INTERRUPT
6320	020112	104002		ERROR 2		:TO SLOW/NOT COMPLETE ERROR
6321						
6322	020114	104421		TCHKOP		:CHECK OPERATION FOR ANY ERRORS
6323	020116	104004		ERROR 4 ;OR 5, 6, 7, 10		:REPORT ALL ERRORS
6324						
6325	020120	004437	042260	JSR R4,GENCOM		:COMPARE DATA
6326	020124	100000		100000		
6327	020126	000401		401		:401 WORDS
6328	020130	000413		BR 1\$:NO MISCOMPARES-SKIP
6329	020132	104015		ERROR 15		:PRINT FIRST ERROR
6330						
6331	020134	013700	001634	MOV ERRMT,R0		:GET ERROR LIMIT
6332	020140	005300		DEC R0		:DECREMENT COUNT
6333	020142	001406		BEQ 65\$:IF ZERO - EXIT
6334	020144	004437	042260	JSR R4,GENCOM		:CONTINUE DATA COMPARE
6335	020150	040000		40000		
6336	020152	000402		BR 65\$:NO MORE ERRORS - EXIT
6337	020154	104016		ERROR 16		:REPORT NEXT ERROR
6338	020156	000770		BR 64\$:LOOP
6339	020160					
6340	020160					
6341						
6342						
6343						
6344						
6345						
6346						
6347						
6348						
6349						
6350						
6351						
6352	020160	000004		SCOPE		
6353	020162	012737	000012 001262	MOV #10.,\$TIMES		::DO 10. ITERATIONS
6354	020170	104416		TSSINIT		:CLEAR SUBSYSTEM
6355	020172	104003		ERROR 3		:BAD INIT ERROR
6356						
6357						
6358	020174	004437	042260	JSR R4,GENCOM		:GENERATE DATA
6359	020200	000003		3		:PATTERN 3
6360	020202	001000		1000		:1000 WORDS
6361						
6362	020204	004437	042260	JSR R4,GENCOM		:CLEAR Ibuff TO ALL ONES
6363	020210	002007		2007		
6364	020212	001000		1000		
6365						
6366	020214	004437	035674	JSR R4,LRLOAD		:LOAD 'L' REGS

64\$:

65\$:

1\$:

*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.

TST55:

: GENERATE SAME DATA AS USED IN MID TRANSFER SEEK ON WRITE (PART 1)

```
6367 020220 000121 RDDATA ;RDDATA
6368 020222 177000 -1000 ;-1000 WORDS
6369 020224 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6370 020226 025 .BYTE 25 ;SECTOR 25
6371 020227 000 .BYTE 0 ;TRACK 0
6372 020230 000312 312 ;CYLINDER 312
6373
6374 020232 104417 TLOADRK ;LOAD RK REGS
6375 020234 104430 TWAT96 ;WAIT FOR INTERRUPT
6376 020236 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6377
6378 020240 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6379 020242 104004 ERROR ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6380
6381 020244 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6382 020250 100000 100000 ;1000 WORDS
6383 020252 001000 BR 1$ ;NO MISCOMPARES-SKIP
6384 020254 000413 ERROR 15 ;PRINT FIRST ERROR
6385 020256 104015
6386
6387 020260 013700 001634 MOV ERRMT,R0 ;GET ERROR LIMIT
6388 020264 005300 64$: DEC R0 ;DECREMENT COUNT
6389 020266 001406 BEQ 65$ ;IF ZERO - EXIT
6390 020270 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6391 020274 040000 40000
6392 020276 000402 BR 65$ ;NO MORE ERRORS - EXIT
6393 020300 104016 ERROR 16 ;REPORT NEXT ERROR
6394 020302 000770 BR 64$ ;LOOP
6395 020304
6396 020304
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408 020304 000004 TST56: SCOPE
6409 020306 012737 000012 001262 MOV #10, $TIMES ;;DO 10. ITERATIONS
6410 020314 104416 TSSINIT ;CLEAR SUBSYSTEM
6411 020316 104003 ERROR 3 ;BAD INIT ERROR
6412
6413 ; GENERATE SAME DATA AS USED IN MID TRANSFER SEEK ON WRITE (PART 2)
6414 020320 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6415 020324 000004 4 ;PATTERN 4
6416 020326 001000 1000 ;1000 WORDS
6417
6418 020330 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6419 020334 002007 2007
6420 020336 001000 1000
6421
6422 020340 004437 035674 JSR R4,LRLOAD ;LOAD "L" REGS
```


6479
6480 020476 104421
6481 020500 104004
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491 020502 000004
6492 020504 012737 000012 001262
6493 020512 104416
6494 020514 104003
6495 020516 013737 036120 020540
6496 020524 004437 035674
6497 020530 000121
6498 020532 177377
6499 020534 070414
6500 020536 025
6501 020537 002
6502 020540 000000
6503
6504 020542 104417
6505 020544 104434
6506 020546 104002
6507
6508 020550 104422
6509 020552 001000
6510 020554 000000
6511 020556 000000
6512 020560 104004
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526 020562 000004
6527 020564 012737 000012 001262
6528 020572 012737 000312 001676
6529 020600 012737 020610 001110
6530 020606 005001
6531 020610
6532 020610 104416
6533 020612 104003
6534 020614 012737 010127 001600

```
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
*****
*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.
*****
TST60: SCOPE
MOV #10.,$TIMES ;;DO 10. ITERATIONS
TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
MOV LSTCYL,1$
JSR R4,LRLoad ;LOAD 'L' REGS
RDDATA
-401 ;401 WORDS
IBUFF ;IBUFF IS BUFF ADDR
.BYTE 25 ;SEC 25
.BYTE 2 ;TRK 2
1$: 0 ;CYL 632/1456

TLOADRK ;LOAD RK REGS
TWAT159 ;WAIT FOR INTERRUPT
ERROR 2 ;TO SLOW/NOT COMPLETE ERROR

TCHKWE ;CHECK OPERATION WITH EXPECTED ERROR
COERR ;CYLINDER OVERFLOW
0
0
ERROR 4; OR 5,6,7 ;REPORT ANY DISCREPANCIES
```

.SBTTL **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.
*****
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
```

```

6535 020622 053737 001720 001600 BIS DTYP, L.CS1 ;SET DRV TYP
6536 020630 013737 001626 001610 MOV DRVNUM, L.CS2 ;SET DRIVE NUMBER
6537 020636 012737 000074 001602 MOV #74, L.WC ;SET WORD COUNT
6538 020644 110137 001607 MOV R1, L.DT ;LOAD TRACK ADDRESS
6539 020650 012737 072414 001604 MOV #OBUFF, L.BA ; SET BUS ADDRESS
6540 020656 012737 000312 001614 MOV #312, L.DCYL ; CYLINDER ADDRESS
6541
6542 020664 004437 042260 JSR R4, GENCOM ;GENERATE HEADER
6543 020670 001200 1200 ;INCLUDE BAD SECTOR BITS
6544
6545 020672 104417 TLOADRK ;LOAD RK REGS
6546 020674 104434 TWAT159 ;WAIT FOR INTERRUPT
6547 020676 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6548
6549 020700 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6550 020702 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
6551
6552 020704 104415 SCOP1 ;LOCAL LOOP ON ERROR TO 1$
6553
6554 020706 005701 TST R1 ;R1 POINTING TO TRACK 0
6555 020710 001002 BNE 2$ ;NO-SKIP
6556 020712 005201 INC R1 ;BUMP TO TRACK 1
6557 020714 000735 BR 1$ ;LOOP
6558 020716 012737 020726 001110 2$: MOV #3$, $LPERR ;SET LOCAL LOOP ON ERROR
6559 020724 005001 CLR R1 ;CLEAR TRACK POINTER
6560 020726 3$:
6561 020726 104416 TSSINIT ;CLEAR SUBSYSTEM
6562 020730 104003 ERROR 3 ;BAD INIT ERROR
6563 020732 012737 010125 001600 MOV #RDHEAD!CFMT, L.CS1 ;LOAD READ 24(8) SECTOR FORMAT
6564 020740 013737 001626 001610 MOV DRVNUM, L.CS2 ;LOAD DRIVE NUMBER
6565 020746 110137 001607 MOV R1, L.DT ;LOAD TRACK
6566 020752 012737 000312 001614 MOV #312, L.DCYL ;LOAD CYLINDER
6567
6568 020760 004437 036362 JSR R4, RDSTHD ;GO READ STANDARD HEADER
6569 020764 104421 TCHKOP ;RETURN IF CERR W/O DATA LATE SET
6570 020766 104004 ERROR 4; OR 5,6,7 ;REPORT ALL OTHER ERRORS
6571 020770 104013 ERROR 13 ;REPORT DATA LATE
6572 020772 104002 ERROR 2 ;REPORT 'OPERATION TO SLOW' OR 'HEADER
6573 ;O NOT FOUND' MESSAGE
6574
6575 020774 104415 SCOP1 ;LOCAL LOOP TO 3$ ON ERROR
6576 020776 004437 042260 JSR R4, GENCOM ;GENERATE & COMPARE HEADERS
6577 021002 101200 101200 ;INCLUDING BAD SECTOR LISTS
6578 021004 000413 BR 4$ ;NO MISCOMPARES-SKIP
6579 021006 104015 ERROR 15 ;REPORT FIRST MISCOMPARE
6580
6581 021010 013700 001634 MOV ERRMT, R0 ;GET ERROR LIMIT
6582 021014 005300 64$: DEC R0 ;DECREMENT COUNT
6583 021016 001406 BEQ 65$ ;IF ZERO - EXIT
6584 021020 004437 042260 JSR R4, GENCOM ;CONTINUE DATA COMPARE
6585 021024 040000 40000
6586 021026 000402 BR 65$ ;NO MORE ERRORS - EXIT
6587 021030 104016 ERROR 16 ;REPORT NEXT ERROR
6588 021032 000770 BR 64$ ;LOOP
6589 021034 65$:
6590

```

```
6591 021034 104415          4$: SCOP1          ;LOCAL LOOP TO 3$
6592 021036 005701          TST R1          ;POINTING TO TRACK 1
6593 021040 001002          BNE 5$         ;NO-EXIT
6594 021042 005201          INC R1         ;BUMP TO TRACK 1
6595 021044 000730          BR 3$         ;LOOP
6596
6597 021046
6598
6599
6600
6601
6602
6603
6604
6605
6606 021046 000004          5$:
6607 021050 012737 000012 001262  :*****
6608 021056 012737 000312 001676  :*TEST 62      24 SECTOR FORMAT DATA TRANSFER (PART 1)
6609 021064 104416          :*
6610 021066 104003          :*   ISSUE A WRITE DATA OF 400 WORDS IN 24 SECTOR FORMAT
6611          :*   TO CYLINDER 312, TRACK 0, SECTOR 0. READ SECTOR BACK
6612          :*   AND MAKE SURE IT IS CORRECT.
6613          :*
6614          :*
6615          :*
6616 021070 004437 042260          TST62: SCOPE
6617 021074 000013          MOV #10, $TIMES ;:DO 10. ITERATIONS
6618 021076 000400          MOV #312, REFMT ;:SET REFORMAT SWITCH
6619          TSSINIT          ;:CLEAR SUBSYSTEM
6620          ERROR 3          ;:BAD INIT ERROR
6621 021100 004437 042260          JSR R4, GENCOM ;:GENERATE DATA
6622 021104 002007          13          ;:PATTERN 13
6623 021106 000400          400         ;:400 WORDS
6624 021110 004437 042260          JSR R4, GENCOM ;:CLEAR Ibuff TO ALL ONES
6625 021104 002007          2007        ;
6626 021106 000400          400         ;
6627 021110 004437 035674          JSR R4, LRLOAD ;:LOAD "L" REGS
6628 021114 010123          WRDATA!CFMT ;:WRDATA!CFMT
6629 021116 177400          -400        ;:-400 WORDS
6630 021120 072414          OBUFF       ;:OBUFF IS BUFF ADDRESS
6631 021122 000         .BYTE 0      ;:SECTOR 0
6632 021123 000         .BYTE 0      ;:TRACK 0
6633 021124 000312          312        ;:CYLINDER 312
6634
6635 021126 104417          TLOADRK     ;:LOAD RK REGS
6636 021130 104430          TWAT96     ;:WAIT FOR INTERRUPT
6637 021132 104002          ERROR 2    ;:TO SLOW/NOT COMPLETE ERROR
6638
6639 021134 104421          TCHKOP     ;:CHECK OPERATION FOR ANY ERRORS
6640 021136 104004          ERROR 4 ;OR 5, 6, 7, 10 ;:REPORT ALL ERRORS
6641
6642 021140 004437 035674          JSR R4, LRLOAD ;:LOAD "L" REGS
6643 021144 010121          RDDATA!CFMT ;:RDDATA!CFMT
6644 021146 177400          -400        ;:-400 WORDS
6645 021150 070414          Ibuff      ;:IBUFF IS BUFF ADDRESS
6646 021152 000         .BYTE 0      ;:SECTOR 0
6647 021153 000         .BYTE 0      ;:TRACK 0
6648 021154 000312          312        ;:CYLINDER 312
6649
6650 021156 104417          TLOADRK     ;:LOAD RK REGS
6651 021160 104424          TWAT32     ;:WAIT FOR INTERRUPT
6652 021162 104002          ERROR 2    ;:TO SLOW/NOT COMPLETE ERROR
6653
6654
```

```
6647 021164 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6648 021166 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6649
6650 021170 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6651 021174 100000 100000
6652 021176 000400 400 ;400 WORDS
6653 021200 000413 BR 1$ ;NO MISCOMPARES-SKIP
6654 021202 104015 ERROR 15 ;REPORT 1ST ERROR
6655
6656 021204 013700 001634 MOV ERRMT,RO ;GET ERROR LIMIT
6657 021210 005300 64$: DEC RO ;DECREMENT COUNT
6658 021212 001406 BEQ 65$ ;IF ZERO - EXIT
6659 021214 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6660 021220 040000 40000
6661 021222 000402 BR 65$ ;NO MORE ERRORS - EXIT
6662 021224 104016 ERROR 16 ;REPORT NEXT ERROR
6663 021226 000770 BR 64$ ;LOOP
6664 021230 65$:
6665
6666 021230 1$:
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677
6678 021230 000004 TST63: SCOPE
6679 021232 012737 000012 001262 MOV #10,$TIMES ;;DO 10. ITERATIONS
6680 021240 012737 000312 001676 MOV #312,REFMT ;SET REFORMAT SWITCH
6681 021246 004737 034522 JSR PC,OPTTST ;SET UP OPTIONS
6682 021252 104416 TSSINIT ;CLEAR SUBSYSTEM
6683 021254 104003 ERROR 3 ;BAD INIT ERROR
6684
6685 021256 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6686 021262 000015 15 ;PATTERN 15
6687 021264 001000 1000 ;1000 WORDS
6688
6689 021266 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6690 021272 002007 2007
6691 021274 001000 1000
6692
6693 021276 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6694 021302 010123 WRDATA!CFMT ;WRDATA!CFMT
6695 021304 177000 -1000 ;-1000 WORDS
6696 021306 072414 OBUFF ;OBUFF IS BUFF ADDRESS
6697 021310 023 .BYTE 23 ;SECTOR 23
6698 021311 000 .BYTE 0 ;TRACK 0
6699 021312 000312 312 ;CYLINDER 312
6700
6701 021314 104417 TLOADRK ;LOAD RK REGS
6702 021316 104430 TWAT96 ;WAIT FOR INTERRUPT
```

```
6703 021320 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6704
6705 021322 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6706 021324 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6707
6708 021326 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6709 021332 010121 RDDATA!CFMT ;RDDATA!CFMT
6710 021334 177000 -1000 ;-1000 WORDS
6711 021336 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6712 021340 023 .BYTE 23 ;SECTOR 23
6713 021341 000 .BYTE 0 ;TRACK 0
6714 021342 000312 312 ;CYLINDER 312
6715
6716 021344 104417 TLOADRK ;LOAD RK REGS
6717 021346 104426 TWAT64 ;WAIT FOR INTERRUPT
6718 021350 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6719
6720 021352 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6721 021354 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6722
6723 021356 004437 042260 JSR R4,GENCOM ;COMPARE DATA
6724 021362 100000 100000
6725 021364 001000 1000 ;1000 WORDS
6726 021366 000413 BR 1$ ;NO MISCOMPARES-SKIP
6727 021370 104015 ERROR 15 ;REPORT FIRST ERROR
6728
6729 021372 013700 001634 MOV ERRLMT,R0 ;GET ERROR LIMIT
6730 021376 005300 64$: DEC R0 ;DECREMENT COUNT
6731 021400 001406 BEQ 65$ ;IF ZERO - EXIT
6732 021402 004437 042260 JSR R4,GENCOM ;CONTINUE DATA COMPARE
6733 021406 040000 40000
6734 021410 000402 BR 65$ ;NO MORE ERRORS - EXIT
6735 021412 104016 ERROR 16 ;REPORT NEXT ERROR
6736 021414 000770 BR 64$ ;LOOP
6737 021416 65$:
6738
6739 021416 1$:
6740
6741 .SBTTL **SPECIAL DATA TRANSFER TESTS
6742
6743 ;*****
6744 ;*TEST 64 MULTI-SECTOR DATA TRANSFER AND BSE
6745 ;*
6746 ;* FORMAT CYLINDER 312, TRACK 0 IN 26 SECTOR FORMAT WITH
6747 ;* SECTOR 1 MARKED BAD. ISSUE A WRITE DATA OF 1000 WORDS
6748 ;* TO CYLINDER 312, TRACK 0, SECTOR 0. MAKE SURE BAD SECTOR
6749 ;* ERROR SETS AND RKDA IS CORRECT. READ SECTOR 0 AND
6750 ;* MAKE SURE IT IS CORRECT.
6751 ;*
6752 ;* ISSUE A READ DATA OF 1000 WORDS TO CYLINDER 312, TRACK 0,
6753 ;* SECTOR 0. MAKE SURE BAD SECTOR ERROR SETS AND THE
6754 ;* PREVIOUS SECTOR IS LOADED CORRECTLY INTO MEMORY.
6755 ;*
6756 ;*****
6757 021416 000004 TST64: SCOPE
6758 021420 012737 000012 001262 MOV #10.,$TIMES ;:DO 10. ITERATIONS
```

```

6759 021426 012737 000312 001676 MOV #312,REFMT ;SET REFORMAT SWITCH
6760 021434 104416 TSSINIT ;CLEAR SUBSYSTEM
6761 021436 104003 ERROR 3 ;BAD INIT ERROR
6762
6763 021440 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6764 021444 000127 WRHEAD ;WRHEAD
6765 021446 177676 -102 ;-102 WORDS
6766 021450 072414 OBUFF ;OBUFF IS BUFF ADDRESS
6767 021452 000 .BYTE 0 ;SECTOR 0
6768 021453 000 .BYTE 0 ;TRACK 0
6769 021454 000312 312 ;CYLINDER 312
6770
6771 021456 004437 042260 JSR R4,GENCOM ;BUILD HEADERS
6772 021462 000600 600
6773
6774 021464 042737 040000 072424 BIC #BIT14,OBUFF+10 ;MARK SECTOR 1 BAD
6775 021472 042737 040000 072426 BIC #BIT14,OBUFF+12 ;CORRECT HURC
6776
6777 021500 104417 TLOADRK ;LOAD RK REGS
6778 021502 104431 TWAT112 ;WAIT FOR INTERRUPT
6779 021504 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6780
6781 021506 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6782 021510 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
6783
6784 021512 004437 042260 JSR R4,GENCOM ;GENERATE DATA
6785 021516 000016 16 ;PATTERN 16
6786 021520 001000 1000 ;1000 WORDS
6787
6788 021522 004437 042260 JSR R4,GENCOM ;CLEAR Ibuff TO ALL ONES
6789 021526 002007 2007
6790 021530 001000 1000
6791
6792 021532 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6793 021536 000123 WRDATA ;WRDATA
6794 021540 177000 -1000 ;-1000 WORDS
6795 021542 072414 OBUFF ;OBUFF IS BUFF ADDRESS
6796 021544 000 .BYTE 0 ;SECTOR 0
6797 021545 000 .BYTE 0 ;TRACK 0
6798 021546 000312 312 ;CYLINDER 312
6799
6800 021550 104417 TLOADRK ;LOAD RK REGS
6801 021552 104424 TWAT32 ;WAIT FOR INTERRUPT
6802 021554 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6803
6804 021556 104422 TCHKWE ;CHECK OPERATION WITH-EXPECTED ERR
6805 021560 000000 0
6806 021562 000100 BSERR ;BAD SECTOR ERROR
6807 021564 000000 0
6808 021566 104004 ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
6809 021570 005037 050002 CLR GRP4ER ;CLEAR GROUP 4 ERRORS
6810 021574 004437 037642 JSR R4,CHKCTS ;CHECK CYL, TRK, SECT CORRECT AFTER ABORTED WRITE
6811 021600 032737 000020 050002 BIT #TRKERR,GRP4ER ;TRK IN ERROR?
6812 021606 001416 BEQ 1$ ;NO-SKIP
6813 021610 012737 054170 001450 MOV #EM13,EM10N ;'TRACK ADDRESS INCORRECT'
6814 021616 013737 047756 001202 MOV EXPTRK,$REG10 ;EXPECTED VALUE
  
```

I 10

CZR6KFO RK6 FCTNL CTRL DIAG MACY11 30A(1052) 19-JUN-80 14:14 PAGE 126
 CZR6KF.P11 19-JUN-80 14:04 T64 MULTI-SECTOR DATA TRANSFER AND BSE SEQ 0125

6815	021624	013737	047770	001204		MOV	REALTRK,\$REG11	;REAL VALUE
6816	021632	012737	050467	061002		MOV	#OPER37,DF010A	;"AFTER WRITE DATA TERMINATED WITH BSE"
6817	021640	104010				ERROR	10	
6818	021642	000527				BR	5\$;EXIT
6819								
6820	021644	032737	000040	050002	1\$:	BIT	#SECERR,GRP4ER	;SECTOR IN ERROR?
6821	021652	001422				BEQ	3\$;NO-SKIP
6822	021654	012737	054220	001450		MOV	#EM14,EM10N	;"SECTOR ADDRESS INCORRECT"
6823	021662	012737	050467	061002		MOV	#OPER37,DF010A	;"AFTER WRITE DATA ABORTED WITH BSE"
6824	021670	013737	047754	001202		MOV	EXPSEC,\$REG10	;EXPECTED VALUE
6825	021676	013737	047772	001204		MOV	REALSEC,\$REG11	;REAL VALUE
6826	021704	104010				ERROR	10	
6827	021706	000505				BR	5\$;EXIT
6828	021710	104415				SCOP1		;LOCAL LOOP TO BEGINNING OF TEST
6829	021712	012737	021720	001110		MOV	#3\$,\$LPERR	;SET LOCAL LOOP ON ERROR
6830	021720				3\$:			
6831	021720	104416				TSSINIT		;CLEAR SUBSYSTEM
6832	021722	104003				ERROR	3	;BAD INIT ERROR
6833	021724	004437	035674			JSR	R4,LRLOAD	;LOAD "L" REGS
6834	021730	000121				RDDATA		;RDDATA
6835	021732	177400				-400		;-400 WORDS
6836	021734	070414				IBUFF		;IBUFF IS BUFF ADDRESS
6837	021736	000				.BYTE	0	;SECTOR 0
6838	021737	000				.BYTE	0	;TRACK 0
6839	021740	000312				312		;CYLINDER 312
6840								
6841	021742	104417				TLOADRK		;LOAD RK REGS
6842	021744	104424				TWAT32		;WAIT FOR INTERRUPT
6843	021746	104002				ERROR	2	;TO SLOW/NOT COMPLETE ERROR
6844								
6845	021750	104421				TCHKOP		;CHECK OPERATION FOR ANY ERRORS
6846	021752	104004				ERROR	4 ;OR 5, 6, 7, 10	;REPORT ALL ERRORS
6847								
6848	021754	004437	042260			JSR	R4,GENCOM	;COMPARE DATA
6849	021760	100000				100000		
6850	021762	000400				400		;400 WORDS
6851	021764	000413				BR	4\$;NO MISCOMPARES-EXIT
6852	021766	104015				ERROR	15	;REPORT FIRST ERROR
6853								
6854	021770	013700	001634			MOV	ERRLMT,RO	;GET ERROR LIMIT
6855	021774	005300			64\$:	DEC	RO	;DECREMENT COUNT
6856	021776	001406				BEQ	65\$;IF ZERO - EXIT
6857	022000	004437	042260			JSR	R4,GENCOM	;CONTINUE DATA COMPARE
6858	022004	040000				40000		
6859	022006	000402				BR	65\$;NO MORE ERRORS - EXIT
6860	022010	104016				ERROR	16	;REPORT NEXT ERROR
6861	022012	000770				BR	64\$;LOOP
6862	022014				65\$:			
6863								
6864	022014	004437	042260		4\$:	JSR	R4,GENCOM	;CLEAR IBUFF
6865	022020	002007				2007		
6866	022022	001000				1000		
6867								
6868	022024	004437	035674			JSR	R4,LRLOAD	;LOAD "L" REGS
6869	022030	000121				RDDATA		;RDDATA
6870	022032	177000				-1000		;-1000 WORDS

6871	022034	070414			IBUFF			:IBUFF IS BUFF ADDRESS
6872	022036	000			.BYTE	0		:SECTOR 0
6873	022037	000			.BYTE	0		:TRACK 0
6874	022040	000312			312			:CYLINDER 312
6875								
6876	022042	104417			TLOADRK			:LOAD RK REGS
6877	022044	104424			TWAT32			:WAIT FOR INTERRUPT
6878	022046	104002			ERROR	2		:TO SLOW/NOT COMPLETE ERROR
6879								
6880	022050	104422			TCHKWE			:CHECK OPERATION WITH EXPECTED ERROR
6881	022052	000000			0			
6882	022054	000100			BSERR			:BAD SECTOR ERROR
6883	022056	000000			0			
6884	022060	104004			ERROR	4; OR 5,6,7		:REPORT ALL DISCREPANCIES
6885								
6886	022062	004437	042260		JSR	R4,GENCOM		:COMPARE DATA AGAIN
6887	022066	100000			100000			
6888	022070	000400			400			:400 WORDS
6889	022072	000413			BR	5\$:NO MISCOMPARES
6890	022074	104015			ERROR	15		:REPORT FIRST ERROR
6891								
6892	022076	013700	001634		MOV	ERRLMT,RO		:GET ERROR LIMIT
6893	022102	005300		66\$:	DEC	RO		:DECREMENT COUNT
6894	022104	001406			BEQ	67\$:IF ZERO - EXIT
6895	022106	004437	042260		JSR	R4,GENCOM		:CONTINUE DATA COMPARE
6896	022112	040000			40000			
6897	022114	000402			BR	67\$:NO MORE ERRORS - EXIT
6898	022116	104016			ERROR	16		:REPORT NEXT ERROR
6899	022120	000770			BR	66\$:LOOP
6900	022122			67\$:				
6901								
6902	022122			5\$:				
6903					:*****			
6904					:*TEST 65 FORMAT TEST			
6905					:*			
6906					:* FORMAT CYLINDER 312, TRACKS 0 AND 1 IN 26 SECTOR FORMAT.			
6907					:* MAKE SURE NO ERRORS SET. READ SECTORS 0-25 AND MAKE			
6908					:* SURE DATA CHECK DOES NOT OCCUR.			
6909					:*			
6910					:*****			
6911	022122	000004		TST65:	SCOPE			
6912	022124	012737	000001 001262		MOV	#1,\$TIMES		::DO 1 ITERATION
6913	022132	005000			CLR	RO		:CLEAR TRACK COUNTER
6914	022134	012737	022142 001110		MOV	#1\$,\$LPERR		:SET LOCAL LOOP
6915								
6916	022142			1\$:				
6917	022142	104416			TSSINIT			:CLEAR SUBSYSTEM
6918	022144	104003			ERROR	3		:BAD INIT ERROR
6919								
6920	022146	013737	001626 001610		MOV	DRVNUM,L.CS2		:LOAD DRIVE NUMBER
6921	022154	012737	000127 001600		MOV	#WRHEAD,L.CS1		:LOAD WRITE HEADER
6922	022162	053737	001720 001600		BIS	DTYPE,L.CS1		:SET DRV TYP
6923	022170	110037	001607		MOV	RO,L.DT		:LOAD DESIRED TRACK FROM TRACK COUNTER
6924	022174	012737	072414 001604		MOV	#OBUFF,L.BA		:LOAD BUS ADDRESS
6925	022202	012737	177676 001602		MOV	#-102,L.WC		: WORD COUNT
6926	022210	012737	000312 001614		MOV	#312,L.DCYL		: CYLINDER


```
6927
6928 022216 004437 042260 JSR R4,GENCOM ;BUILD HEADER
6929 022222 001200 1200 ;WITH BSE FLAGGED
6930
6931 022224 104417 TLOADRK ;LOAD RK REGS
6932 022226 104431 TWAT112 ;WAIT FOR INTERRUPT
6933 022230 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6934
6935 022232 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6936 022234 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
6937
6938 022236 104415 SCOP1 ;LOCAL LOOP TO 1$
6939
6940 022240 005700 TST R0 ;R0 AT ZERO?
6941 022242 001002 BNE 2$ ;NO-EXIT
6942 022244 005200 INC R0 ;BUMP COUNTER
6943 022246 000735 BR 1$ ;LOOP
6944 022250 005000 CLR R0 ;CLEAR SECTOR COUNTER
6945 022252 012737 022260 001110 2$: MOV #113$, $LPERR ;SET LOCAL LOOP ON ERROR
6946
6947 022260 113$:
6948 022260 104416 TSSINIT ;CLEAR SUBSYSTEM
6949 022262 104003 ERROR 3 ;BAD INIT ERROR
6950
6951 022264 004437 035674 3$: JSR R4,LRLOAD ;LOAD 'L' REGS
6952 022270 000121 RDDATA ;RDDATA
6953 022272 177400 -400 ;-400 WORDS
6954 022274 070414 Ibuff ;IBUFF IS BUFF ADDRESS
6955 022276 000 .BYTE 0 ;SECTOR 0
6956 022277 000 .BYTE 0 ;TRACK 0
6957 022300 000312 312 ;CYLINDER 312
6958
6959 022302 110037 001606 MOVb R0,L.DS ;LOAD SECTOR COUNTER INTO DESIRED SECTOR
6960
6961 022306 104417 TLOADRK ;LOAD RK REGS
6962 022310 104424 TWAT32 ;WAIT FOR INTERRUPT
6963 022312 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
6964
6965 022314 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
6966 022316 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
6967
6968 022320 104415 SCOP1 ;LOCAL LOOP TO 3$ ON ERROR
6969
6970 022322 022700 000024 CMP #24,R0 ;LAST SECTOR READ?
6971 022326 001402 BEQ 4$ ;YES-EXIT
6972 022330 005200 INC R0 ;BUMP SECTOR COUNTER
6973 022332 000754 BR 3$ ;LOOP
6974
6975 022334 005037 001676 4$: CLR REfmt ;CLEAR REFORMAT SWITCH
6976
6977 .SBTTL **WRITE CHECK TESTS
6978
6979 ;*****
6980 ;*TEST 66 WRITE-CHECK WITH NO ERROR
6981 ;*
6982 ;* WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH A KNOWN PATTERN.
```

```
6983      :*      DO A WRITE-CHECK OF 400 WORDS. MAKE SURE NO
6984      :*      ERROR OCCURS.
6985      :*
6986      :*****
6987 022340 000004 TST66: SCOPE
6988 022342 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
6989 022350 104416 TSSINIT ;CLEAR SUBSYSTEM
6990 022352 104003 ERROR 3 ;BAD INIT ERROR
6991
6992 022354 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
6993 022360 000123 WRDATA ;WRDATA
6994 022362 177400 -400 ;-400 WORDS
6995 022364 072414 OBUFF ;OBUFF IS BUFF ADDRESS
6996 022366 000 .BYTE 0 ;SECTOR 0
6997 022367 000 .BYTE 0 ;TRACK 0
6998 022370 000312 312 ;CYLINDER 312
6999
7000 022372 004437 042260 JSR R4,GENCOM ;GENERATE DATA
7001 022376 000002 2 ;PATTERN 2
7002 022400 000400 400 ;400 WORDS
7003
7004 022402 104417 TLOADRK ;LOAD RK REGS
7005 022404 104430 TWAT96 ;WAIT FOR INTERRUPT
7006 022406 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7007
7008 022410 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7009 022412 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7010
7011 022414 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7012 022420 000131 WRTCHK ;WRTCHK
7013 022422 177400 -400 ;-400 WORDS
7014 022424 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7015 022426 000 .BYTE 0 ;SECTOR 0
7016 022427 000 .BYTE 0 ;TRACK 0
7017 022430 000312 312 ;CYLINDER 312
7018
7019 022432 104417 TLOADRK ;LOAD RK REGS
7020 022434 104424 TWAT32 ;WAIT FOR INTERRUPT
7021 022436 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7022
7023 022440 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7024 022442 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7025      :*****
7026      :*TEST 67 WRITE CHECK ERROR (PART 1)
7027      :*
7028      :* WRITE CYLINDER 312, TRACK 0, SECTOR 0 WITH ALL ZEROES.
7029      :* WRITE CHECK CYLINDER 312, TRACK 0, SECTOR 0 WITH SAME
7030      :* DATA EXCEPT WORD 110 HAS ONE OF THE FOLLOWING
7031      :* CONFIGURATIONS:
7032      :*
7033      :* 000001 000020 000400 010000
7034      :* 000002 000040 001000 020000
7035      :* 000004 000100 002000 040000
7036      :* 000010 000200 004000 100000
7037      :*
7038      :* MAKE SURE WRITE CHECK ERROR SET FOR EACH
```

```
7039      :*      OF THE CONFIGURATIONS AND THAT THE BUS ADDRESS
7040      :*      AND WORD COUNT IS CORRECT.
7041      :*
7042      :*****
7043 022444 000004      TST67: SCOPE
7044 022446 012737 000012 001262  MOV      #10,$TIMES      ;;DO 10. ITERATIONS
7045 022454 012700 000001      MOV      #BIT0,RO        ;SET LO ORDER BIT IN RO FOR
7046      TSSINIT          ;CAUSING WRITE CHECK ERROR
7047 022460 104416      ERROR      3          ;CLEAR SUBSYSTEM
7048 022462 104003      JSR      R4,GENCOM      ;BAD INIT ERROR
7049 022464 004437 042260      1          ;GENERATE DATA, ALL 0'S
7050 022470 000001      400
7051 022472 000400
7052
7053 022474 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
7054 022500 000123      WRDATA          ;WRDATA
7055 022502 177400      -400          ;-400 WORDS
7056 022504 072414      OBUFF          ;OBUFF IS BUFF ADDRESS
7057 022506      000      .BYTE      0          ;SECTOR 0
7058 022507      000      .BYTE      0          ;TRACK 0
7059 022510 000312      312          ;CYLINDER 312
7060
7061 022512 104417      TLOADRK        ;LOAD RK REGS
7062 022514 104430      TWAT96        ;WAIT FOR INTERRUPT
7063 022516 104002      ERROR      2          ;TO SLOW/NOT COMPLETE ERROR
7064
7065 022520 104421      TCHKOP        ;CHECK OPERATION FOR ANY ERRORS
7066 022522 104004      ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7067
7068 022524 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
7069 022530 000131      WRTCHK        ;WRTCHK
7070 022532 177400      -400          ;-400 WORDS
7071 022534 072414      OBUFF          ;OBUFF IS BUFF ADDRESS
7072 022536      000      .BYTE      0          ;SECTOR 0
7073 022537      000      .BYTE      0          ;TRACK 0
7074 022540 000312      312          ;CYLINDER 312
7075
7076 022542 104417      TLOADRK        ;LOAD RK REGS
7077 022544 104424      TWAT32        ;WAIT FOR INTERRUPT
7078 022546 104002      ERROR      2          ;TO SLOW/NOT COMPLETE ERROR
7079
7080 022550 104421      TCHKOP        ;CHECK OPERATION FOR ANY ERRORS
7081 022552 104004      ERROR      4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7082
7083 022554 104415      SCOPI        ;LOCAL LOOP ON WRITE CHECK
7084 022556 012737 022564 001110  MOV      #1$, $LPERR      ;SET LOCAL LOOP
7085 022564 010037 072634 1$:  MOV      RO,OBUFF+220      ;CAUSE ERROR BIT IN BUFFER
7086 022570 104416      TSSINIT          ;CLEAR SUBSYSTEM
7087 022572 104003      ERROR      3          ;BAD INIT ERROR
7088 022574 004437 035674      JSR      R4,LRLOAD      ;LOAD 'L' REGS
7089 022600 000131      WRTCHK        ;WRTCHK
7090 022602 177400      -400          ;-400 WORDS
7091 022604 072414      OBUFF          ;OBUFF IS BUFF ADDRESS
7092 022606      000      .BYTE      0          ;SECTOR 0
7093 022607      000      .BYTE      0          ;TRACK 0
7094 022610 000312      312          ;CYLINDER 312
```

```

7095
7096 022612 104417      TLOADRK      ;LOAD RK REGS
7097 022614 104424      TWAT32       ;WAIT FOR INTERRUPT
7098 022616 104002      ERROR 2     ;TO SLOW/NOT COMPLETE ERROR
7099
7100 022620 104422      TCHKWE      ;CHECK OPERATION WITH EXPECTED ERROR
7101 022622 000000      0
7102 022624 000004      WCKERR      ;WRITE CHECK ERROR
7103 022626 000000      0
7104 022630 104004      ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
7105
7106 022632 104415      SCOP1       ;LOCAL LOOP ON ERROR TO 1$
7107
7108 :
7109 :
7110 :
7111 :
7112 :
7113 :
7114 :
7115 :
7116 :
7117 :
7118 022634 023727 001544 072636  CMP T.BA,#OBUFF+222 ;CHECK BA HALT AT PROPER PLACE
7119 022642 001416      BEQ 2$      ;YES-SKIP
7120 022644 101040      BHI 6$     ;IF TO HI - SKIP
7121 022646 012737 054033 001450  MOV #EM11,EM10N ;"INCORRECT BA"
7122 022654 012737 072636 001202  MOV #OBUFF+222,$REG10 ;GOOD VALUE
7123 022662 013737 001544 001204  MOV T.BA,$REG11 ;BAD VALUE
7124 022670 012737 050523 061002  MOV #OPER41,DF010A ;"WRITE CHECK ABORTED WITH WCE"
7125 022676 104010      ERROR 10
7126
7127 022700 023727 001542 177511 2$:  CMP T.WC,#-267   ;CHECK WORD COUNT AT CORRECT VALUE
7128 022706 001461      BEQ 3$      ;YES-SKIP
7129 022710 101037      BHI 7$     ;IF HIGHER SKIP
7130 022712 012737 054006 001450  MOV #EM10,EM10N ;"INCORRECT WC"
7131 022720 012737 050523 061002  MOV #OPER41,DF010A ;"WRITE CHECK ABORTED WITH WCE"
7132 022726 012737 177511 001202  MOV #-267,$REG10 ;GOOD VALUE
7133 022734 013737 001542 001204  MOV T.WC,$REG11 ;ERROR VALUE
7134 022742 104010      ERROR 10
7135 022744 000442      BR 3$      ;EXIT
7136
7137 022746 023727 001544 072642 6$:  CMP T.BA,#OBUFF+226 ;TEST IF BA AT HI SIDE
7138 022754 101415      BLOS 7$    ;YES - SKIP
7139 022756 012737 054033 001450  MOV #EM11,EM10N ;SET MESSAGE
7140 022764 012737 072642 001202  MOV #OBUFF+226,$REG10 ;GOOD VALUE
7141 022772 013737 001544 001204  MOV T.BA,$REG11 ;ERROR VALUE
7142 023000 012737 050523 061002  MOV #OPER41,DF010A ;"WRITE CHECK ABORTED WITH WCE"
7143 023006 104010      ERROR 10
7144
7145 023010 023727 001542 177513 7$:  CMP T.WC,#-265   ;TEST IF WORD COUNT AT HI SIDE
7146 023016 101415      BLOS 3$    ;YES - SKIP
7147 023020 012737 054006 001450  MOV #EM10,EM10N ;SET MESSAGE
7148 023026 012737 050523 061002  MOV #OPER41,DF010A ;"WC ABORTED WITH WCE"
7149 023034 012737 177513 001202  MOV #-265,$REG10 ;GOOD VALUE
7150 023042 013737 001542 001204  MOV T.WC,$REG11 ;ERROR VALUE

```

7151 023050 104010
7152
7153 023052 104415
7154
7155 023054 032700 100000
7156 023060 001002
7157 023062 006300
7158 023064 000637
7159
7160 023066
7161
7162
7163
7164
7165
7166
7167
7168
7169
7170
7171
7172
7173
7174
7175
7176
7177
7178
7179 023066 000004
7180 023070 012737 000012 001262
7181 023076 012700 177776
7182
7183 023102 104416
7184 023104 104003
7185 023106 004437 042260
7186 023112 000007
7187 023114 000400
7188
7189 023116 004437 035674
7190 023122 000123
7191 023124 177400
7192 023126 072414
7193 023130 000
7194 023131 000
7195 023132 000312
7196
7197 023134 104417
7198 023136 104430
7199 023140 104002
7200
7201 023142 104421
7202 023144 104004
7203
7204 023146 004437 035674
7205 023152 000131
7206 023154 177400

```
ERROR 10
3$: SCOP1 ;LOCAL LOOP ON ERROR TO 1$
BIT #BIT15,R0 ;BIT 15 SET?
BNE 4$ ;YES-EXIT
ASL R0 ;SHIFT ERROR BIT
BR 1$ ;LOOP

4$:
*****
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.
*****
TST70: SCOPE
MOV #10,$TIMES ;;DO 10. ITERATIONS
MOV #177776,R0 ;LOAD R0 FOR CAUSING WRITE CHECK ERROR

TSSINIT ;CLEAR SUBSYSTEM
ERROR 3 ;BAD INIT ERROR
JSR R4,GENCOM ;GENERATE DATA
7 ;ALL 1'S
400 ;400 WORDS

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 LOW/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
WRCHK ;WRCHK
-400 ;-400 WORDS
```

```
7207 023156 072414          OBUFF          ;OBUFF IS BUFF ADDRESS
7208 023160      000        .BYTE 0          ;SECTOR 0
7209 023161      000        .BYTE 0          ;TRACK 0
7210 023162 000312        312          ;CYLINDER 312
7211
7212 023164 104417        TLOADRK        ;LOAD RK REGS
7213 023166 104424        TWAT32         ;WAIT FOR INTERRUPT
7214 023170 104002        ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
7215
7216 023172 104421        TCHKOP        ;CHECK OPERATION FOR ANY ERRORS
7217 023174 104004        ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7218
7219 023176 104415        SCOPI         ;LOCAL LOOP TO START OF TEST
7220 023200 012737 023206 001110  MOV #1$, $LPERR ;SET LOCAL LOOP
7221
7222 023206 010037 072634 1$:  MOV R0, OBUFF+220 ;PUT WORD IN OBUFF TO CAUSE WCE
7223 023212 104416        TSSINIT       ;CLEAR SUBSYSTEM
7224 023214 104003        ERROR 3       ;BAD INIT ERROR
7225
7226 023216 004437 035674        JSR R4, LRLOAD ;LOAD 'L' REGS
7227 023222 000131        WRTCHK        ;WRTCHK
7228 023224 177400        -400         ;-400 WORDS
7229 023226 072414        OBUFF        ;OBUFF IS BUFF ADDRESS
7230 023230      000        .BYTE 0        ;SECTOR 0
7231 023231      000        .BYTE 0        ;TRACK 0
7232 023232 000312        312          ;CYLINDER 312
7233
7234 023234 104417        TLOADRK        ;LOAD RK REGS
7235 023236 104424        TWAT32         ;WAIT FOR INTERRUPT
7236 023240 104002        ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
7237
7238 023242 104422        TCHKWE        ;CHECK OPERATION WITH EXPECTED ERROR
7239 023244 000000        0
7240 023246 000004        WCKERR        ;WRITE CHECK ERROR
7241 023250 000000        0
7242 023252 104004        ERROR 4; OR 5,6,7 ;REPORT ALL DISCREPANCIES
7243
7244 023254 104415        SCOPI         ;LOCAL LOOP TO 1$
7245
7246 :
7247 :
7248 :
7249 :
7250 :
7251 :
7252 :
7253 :
7254 :
7255 :
7256 023256 023727 001544 072636  CMP T.BA, #OBUFF+222 ;CHECK BA HALT AT PROPER PLACE
7257 023264 001416        BEQ 2$        ;YES-SKIP
7258 023266 101040        BHI 6$        ;IF TO HI - SKIP
7259 023270 012737 054033 001450  MOV #EM11, EM10N ;'INCORRECT BA'
7260 023276 012737 072636 001202  MOV #OBUFF+222, $REG10 ;GOOD VALUE
7261 023304 013737 001544 001204  MOV T.BA, $REG11 ;BAD VALUE
7262 023312 012737 050523 061002  MOV #OPER41, DF010A ;'WRITE CHECK ABORTED WITH WCE'
```



```

7319 023540 000123
7320 023542 177400
7321 023544 072414
7322 023546 000
7323 023547 000
7324 023550 000312
7325
7326 023552 104417
7327 023554 104430
7328 023556 104002
7329
7330 023560 104421
7331 023562 104004
7332
7333 023564 005037 072636
7334
7335 023570 004437 035674
7336 023574 000131
7337 023576 177670
7338 023600 072414
7339 023602 000
7340 023603 000
7341 023604 000312
7342
7343 023606 104417
7344 023610 104424
7345 023612 104002
7346
7347 023614 104421
7348 023616 104004
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368
7369
7370
7371 023620 000004
7372 023622 012737 000002 001262
7373 023630 032737 000400 001664
7374 023636 001043

```

```

WRDATA          ;WRDATA
-400            ; -400 WORDS
OBUF           ; OBUF 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  OBUF+222

JSR  R4,LRLOAD ; LOAD 'L' REGS
WRTCHK        ; WRTCHK
-110         ; -110 WORDS
OBUF           ; OBUF 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,$TIMES      ;;DO 2 ITERATIONS
        BIT  #FPFMT,OPTFLG  ;TEST IF FIRST PASS SWITCH SET
        BNE  24$           ;YES - SKIP FORMAT

```



```

7375
7376 023640 105037 023663      CLRB  21$      ;CLEAR ADDRESS POINTERS
7377 023644 005037 023664      CLR   22$
7378
7379 023650 004437 035674      20$: JSR   R4,LRLOAD ;LOAD 'L' REGISTERS
7380 023654 000127             WRHEAD ;WRITE HEADER
7381 023656 177676             -102  ;102 WORDS
7382 023660 072414             OBUFF ;OBUFF IS BUFF ADDRESS
7383 023662 000              .BYTE  0 ;SECTOR 0
7384 023663 000              .BYTE  0 ;TRACK ADDRESS (VARIABLE)
7385 023664 000000           22$: 0 ;CYLINDER 0 (VARIABLE)
7386
7387 023666 004437 042260      JSR   R4,GENCOM ;GO GENERATE HEADERS
7388 023672 001200           1200 ;WITH BAD SECTOR ERRORS
7389
7390 023674 104417             TLOADRK ;LOAD RK REGS
7391 023676 104431             TWAT112 ;WAIT FOR INTERRUPT
7392 023700 104002           ERROR  2 ;TO SLOW/NOT COMPLETE ERROR
7393
7394 023702 104421             TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7395 023704 104004           ERROR  4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7396
7397 023706 123727 023663 000002  CMPB  21$,#2 ;TEST IF LAST TRACK
7398 023714 001403             BEQ   23$ ;YES - SKIP
7399 023716 105237 023663             INCB  21$ ;ELSE BUMP TRACK
7400 023722 000752             BR    20$ ;LOOP
7401
7402 023724 105037 023663 23$: CLRB  21$ ;CLEAR TRACK POINTER
7403 023730 023727 023664 000003  CMP   22$,#3 ;TEST IF LAST CYLINDER WRITTEN
7404 023736 001403             BEQ   24$ ;YES - SKIP
7405 023740 005237 023664             INC   22$ ;ELSE BUMP CYLINDER
7406 023744 000741             BR    20$ ;LOOP
7407
7408 023746 013737 023760 001110 24$: MOV   1$, $LPERR ;SET LOCAL LOOP ON ERROR
7409 023754 012703 000400           MOV   #400,R3 ;SET COUNT FOR SECTOR CLEARING
7410 023760 29$:
7411 023760 104416             TSSINIT ;CLEAR SUBSYSTEM
7412 023762 104003           ERROR  3 ;BAD INIT ERROR
7413 023764 004437 035674 JSR   R4,LRLOAD ;LOAD 'L' REGS
7414 023770 000123             WRDATA ;WRDATA
7415 023772 177400           -400 ;-400 WORDS
7416 023774 072414             OBUFF ;OBUFF IS BUFF ADDRESS
7417 023776 000              .BYTE  0 ;SECTOR 0
7418 023777 000              .BYTE  0 ;TRACK 0
7419 024000 000000           0 ;CYLINDER 0
7420
7421 024002 104417             TLOADRK ;LOAD RK REGS
7422 024004 104434             TWAT159 ;WAIT FOR INTERRUPT
7423 024006 104002           ERROR  2 ;TO SLOW/NOT COMPLETE ERROR
7424
7425 024010 104421             TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7426 024012 104004           ERROR  4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7427
7428 024014 104415             SCOP1 ;LOCAL LOOP ON ERROR TO 1$
7429 024016 005303           DEC   R3 ;DECREMENT COUNT
7430 024020 012762 072414 000004 2$: MOV   #OBUFF,RKBA(R2) ;SET BA

```

7431	024026	012762	177400	000002	MOV	#-400,RKWC(R2)	:AND WC AGAIN
7432	024034	005037	001662		CLR	INTSET	:CLEAR INTERRUPT FLAG
7433	024040	013762	001626	000010	MOV	DRVNUM,RKCS2(R2)	:SET DRIVE NUMBER
7434	024046	012737	000123	036124	MOV	#WRDATA,HOLD	
7435	024054	053737	001720	036124	BIS	DTYPE,HOLD	
7436	024062	013762	036124	000000	MOV	HOLD,RKCS1(R2)	:DO WRITE DATA
7437							
7438	024070	104425			TWAT48		:WAIT FOR INTERRUPT
7439	024072	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
7440							
7441	024074	032762	000200	000014	BIT	#BSE,RKER(R2)	:BAD SECTOR ERROR?
7442	024102	001415			BEQ	3\$:NO-SKIP
7443	024104	032737	000200	001664	BIT	#BSERPT,OPTFLG	:TEST IF BSE TO MANY HAS BEEN REPORTED
7444	024112	001007			BNE	5\$:YES - SKIP
7445	024114	052737	000200	001664	BIS	#BSERPT,OPTFLG	:SET FLAG
7446	024122	012737	052457	001360	MOV	#OPR017,EM1N	:SET MESSAGE
7447	024130	104001			ERROR	1	:FIRST 256 SECTOR NOT BSE FREE"
7448	024132	000137	024466		JMP	14\$:GO TO EXIT
7449							
7450	024136						
7451	024136	104421			TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7452	024140	104004			ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
7453	024142	104415			SCOP1		:LOCAL LOOP TO 1\$ (RESTART SECTOR CLEAR)
7454							
7455	024144	005303			DEC	R3	:DECREMENT COUNT
7456	024146	001324			BNE	2\$:LOOP IF NOT ZERO
7457							
7458	024150	004437	035674		JSR	R4,LRLOAD	:LOAD 'L' REGS
7459	024154	000117			SEEK		:SEEK
7460	024156	000000			0		:0 WORDS
7461	024160	000000			0		:0 IS BUFF ADDRESS
7462	024162	000			.BYTE	0	:SECTOR 0
7463	024163	000			.BYTE	0	:TRACK 0
7464	024164	000000			0		:CYLINDER 0
7465							
7466	024166	104417			TLOADRK		:LOAD RK REGS
7467	024170	104423			TWAT16		:WAIT FOR INTERRUPT
7468	024172	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
7469	024174	005037	001662		CLR	INTSET	:CLEAR FIRST INTERRUPT
7470	024200	104421			TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7471	024202	104004			ERROR	4 ;OR 5, 6, 7	:REPORT ALL ERRORS
7472	024204	104427			TWAT80		:WAIT FOR SECOND INTERRUPT
7473	024206	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
7474	024210	004437	042260		JSR	R4,GENCOM	:GENERATE DATA
7475	024214	004006			4006		:PATTERN 6, 1ST WORD REPEATED
7476	024216	000400			400		:400 WORDS
7477							
7478	024220						
7479	024220	104416			TSSINIT		:CLEAR SUBSYSTEM
7480	024222	104003			ERROR	3	:BAD INIT ERROR
7481	024224	004437	035674		JSR	R4,LRLOAD	:LOAD 'L' REGS
7482	024230	000123			WRDATA		:WRDATA
7483	024232	000000			0		:0 WORDS
7484	024234	072414			OBUFF		:OBUFF IS BUFF ADDRESS
7485	024236	000			.BYTE	0	:SECTOR 0
7486	024237	000			.BYTE	0	:TRACK 0

7487	024240	000000			0		:CYLINDER 0
7488							
7489	024242	052737	000020	001610	BIS	#BAI,L.CS2	
7490							
7491	024250	104417			TLOADRK		:LOAD RK REGS
7492	024252	104437			TWAT8S		:WAIT FOR SECOND INTERRUPT
7493	024254	104002			ERROR	2	:ELSE REPORT TO SLOW/NOT COMPLETE ERROR
7494							
7495	024256				7\$:		
7496	024256	104421			TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7497	024260	104004			ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
7498							
7499	024262	104415			SCOP1		:INTERNAL LOOP ON ERROR TO 4\$
7500	024264	012703	000400		MOV	#400,R3	:SET COUNTER FOR READ-COMPARE LOOP
7501	024270	005037	024332		CLR	10\$:CLEAR SECTOR AND TRACK VALUES
7502	024274	005037	024334		CLR	12\$:CLEAR CYL VALUE
7503	024300	013737	024314	001110	MOV	8\$, \$LPERR	:SET LOCAL LOOP ON ERROR
7504	024306	042737	000020	001610	BIC	#BAI,L.CS2	:CLEAR BAI
7505							
7506	024314				8\$:		
7507	024314	104416			TSSINIT		:CLEAR SUBSYSTEM
7508	024316	104003			ERROR	3	:BAD INIT ERROR
7509	024320	004437	035674		9\$:		
7510	024324	000121			JSR	R4,LRLOAD	:LOAD RK REGS
7511	024326	177400			RDDATA		:READ DATA
7512	024330	070414			-400		:400 WORDS
7513	024332	000			IBUFF		:INTO IBUFF
7514	024333	000			10\$:	.BYTE 0	:SECTOR (VARIABLE)
7515	024334	000000			11\$:	.BYTE 0	:TRACK (VARIABLE)
7516					12\$:	0	:CYL (VARIABLE)
7517	024336	104417			TLOADRK		:LOAD RK REGS
7518	024340	104425			TWAT48		:WAIT FOR INTERRUPT
7519	024342	104002			ERROR	2	:TO SLOW/NOT COMPLETE ERROR
7520							
7521	024344	104421			TCHKOP		:CHECK OPERATION FOR ANY ERRORS
7522	024346	104004			ERROR	4 ;OR 5, 6, 7, 10	:REPORT ALL ERRORS
7523							
7524	024350	104415			SCOP1		:LOCAL LOOP ON ERROR TO 8\$
7525							
7526	024352	004437	042260		JSR	R4,GENCOM	:COMPARE DATA
7527	024356	100000			100000		
7528	024360	000400			400		:400 WORDS
7529	024362	000413			BR	13\$:NO MISCOMPARE-EXIT LOOP
7530	024364	104015			ERROR	15	:REPORT FIRST ERROR
7531							
7532	024366	013700	001634		MOV	ERRLMT,R0	:GET ERROR LIMIT
7533	024372	005300			64\$:	DEC R0	:DECREMENT COUNT
7534	024374	001406			BEQ	65\$:IF ZERO - EXIT
7535	024376	004437	042260		JSR	R4,GENCOM	:CONTINUE DATA COMPARE
7536	024402	040000			40000		
7537	024404	000402			BR	65\$:NO MORE ERRORS - EXIT
7538	024406	104016			ERROR	16	:REPORT NEXT ERROR
7539	024410	000770			BR	64\$:LOOP
7540	024412				65\$:		
7541							
7542	024412	104415			13\$:	SCOP1	:LOCAL LOOP TO 8\$

7543									
7544	024414	005303				DEC	R3		:DEC READ LOOP COUNT
7545	024416	001423				BEQ	14\$:IF ZERO-EXIT
7546									
7547	024420	105237	024332			INCB	10\$:BUMP SECTOR
7548	024424	123727	024332	000026		CMPB	10\$,#26		:FINISHED WITH TRACK?
7549	024432	001332				BNE	9\$:NO-LOOP
7550	024434	105037	024332			CLRB	10\$:CLEAR SECTOR
7551	024440	105237	024333			INCB	11\$:BUMP TRACK
7552	024444	123727	024333	000003		CMPB	11\$,#3		:FINISHED WITH CYLINDER?
7553	024452	001322				BNE	9\$:NO-LOOP
7554	024454	105037	024333			CLRB	11\$:CLEAR TRACK
7555	024460	005237	024334			INC	12\$:BUMP CYL.
7556	024464	000715				BR	9\$:LOOP
7557									
7558	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.

:*****

7576	024466	000004				TST73: SCOPE			
7577	024470	012737	000002	001262		MOV	#2,\$TIMES		::DO 2 ITERATIONS
7578	024476	104416				TSSINIT			:CLEAR SUBSYSTEM
7579	024500	104003				ERROR	3		:BAD INIT ERROR
7580	024502	012700	000620			MOV	#400.,R0		:SET COUNT FOR INTERRUPT WAIT
7581	024506	005037	072414			CLR	OBUFF		
7582									
7583	024512	004437	035674			JSR	R4,LRLOAD		:LOAD 'L' REGS
7584	024516	000123				WRDATA			:WRDATA
7585	024520	000000				0			:0 WORDS
7586	024522	072414				OBUFF			:OBUFF IS BUFF ADDRESS
7587	024524	000				.BYTE	0		:SECTOR 0
7588	024525	000				.BYTE	0		:TRACK 0
7589	024526	000000				0			:CYLINDER 0
7590	024530	052737	000020	001610		BIS	#BAI,L.CS2		:SET BAI
7591	024536	104417				TLOADRK			:LOAD RK REGS
7592	024540	104434			1\$:	TWAT159			:WAIT FOR INTERRUPT
7593	024542	000401				BR	2\$:NO INTERRUPT-SKIP
7594	024544	000403				BR	3\$:INTERRUPT-SKIP
7595									
7596	024546	005300			2\$:	DEC	R0		:DEC WAIT COUNTER
7597	024550	001373				BNE	1\$:NO ZERO-LOOP
7598	024552	104002				ERROR	2		:TO SLOW/NOT COMPLETE ERROR

```

7599
7600 024554 032762 000200 000014 3$: BIT #BSE,RKER(R2) ;DID BSE SET
7601 024562 001415 BEQ 4$ ;NO-SKIP
7602
7603 024564 032737 000200 001664 BIT #BSERPT,OPTFLG ;TEST IF TO MANY BAD SECTORS REPORTED
7604 024572 001007 BNE 12$ ;YES - SKIP
7605 024574 052737 000200 001664 BIS #BSERPT,OPTFLG ;SET FLAG
7606 024602 012737 052457 001360 MOV #OPRO17,EM1N ;SET MESSAGE
7607 024610 104001 ERROR 1 ;'FIRST 256 SECTORS NOT BSE FREE'
7608 024612 000137 025160 12$: JMP 11$ ;EXIT
7609
7610 024616 4$:
7611 024616 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7612 024620 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7613 024622 013737 036120 024652 MOV LSTCYL,13$
7614 024630 013737 036120 025026 MOV LSTCYL,14$
7615
7616 024636 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7617 024642 000117 SEEK
7618 024644 000000 0 ;0 WORDS
7619 024646 000000 0 ;0 IS BUFF ADDR
7620 024650 000 .BYTE 0 ;SEC 0
7621 024651 000 .BYTE 0 ;TRK 0
7622 024652 000000 13$: 0 ;CYL 632/1456
7623
7624 024654 104417 TLOADRK ;LOAD RK REGS
7625 024656 104423 TWAT16 ;WAIT FOR INTERRUPT
7626 024660 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7627 024662 005037 001662 CLR INTSET ;CLEAR INTERRUPT FLAG
7628
7629 024666 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7630 024670 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7631
7632 024672 104427 TWAT80 ;WAIT FOR 2ND INTERRUPT
7633 024674 104002 ERROR 2
7634
7635 024676 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7636 024700 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7637
7638 024702 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7639 024706 000105 CLEAR ;CLEAR
7640 024710 000000 0 ;0 WORDS
7641 024712 000000 0 ;0 IS BUFF ADDRESS
7642 024714 000 .BYTE 0 ;SECTOR 0
7643 024715 000 .BYTE 0 ;TRACK 0
7644 024716 000000 0 ;CYLINDER 0
7645
7646 024720 104417 TLOADRK ;LOAD RK REGS
7647 024722 104423 TWAT16 ;WAIT FOR INTERRUPT
7648 024724 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7649
7650 024726 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7651 024730 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7652
7653 024732 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7654 024736 000123 WRDATA ;WRDATA
  
```

```

7655 024740 000000      0      ;0 WORDS
7656 024742 072414      OBUFF  ;OBUFF IS BUFF ADDRESS
7657 024744      000      .BYTE 0  ;SECTOR 0
7658 024745      000      .BYTE 0  ;TRACK 0
7659 024746 000000      0      ;CYLINDER 0
7660
7661 024750 012737 135143 072414  MOV #135143,OBUFF ;SET WORD FOR OUTPUT
7662 024756 012700 000621      MOV #401.,R0      ;SET COUNT FOR INTERRUPT WAIT
7663 024762 052737 000020 001610  BIS #BAI,L.CS2   ;SET BUS ADDRESS INC INHIBIT
7664
7665 024770 104417      TLOADRK ;LOAD RK REGS
7666 024772 104434      TWAT159 ;WAIT FOR INTERRUPT
5$: 7667 024774 000401      BR 6$   ;NO INTERRUPT-BRANCH
7668 024776 000403      BR 7$   ;INTERRUPT-BRANCH
7669
7670 025000 005300      6$: DEC R0      ;DEC WAIT COUNT
7671 025002 001373      BNE 5$   ;LOOP IF NOT ZERO
7672 025004 104002      ERROR 2  ;TO SLOW/NOT COMPLETE ERROR
7673
7674 025006      7$:
7675 025006 104421      TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7676 025010 104004      ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7677
7678 025012 004437 035674  JSR R4,LRLOAD ;LOAD 'L' REGS
7679 025016 000117      SEEK
7680 025020 000000      0      ;0 WORDS
7681 025022 000000      0      ;0 IS BUFF ADDR
7682 025024      000      .BYTE 0  ;SEC 0
7683 025025      000      .BYTE 0  ;TRK 0
7684 025026 000000      14$: 0      ;CYL 632/1456
7685 025030 104417      TLOADRK ;LOAD RK REGS
7686 025032 104423      TWAT16 ;WAIT FOR INTERRUPT
7687 025034 104002      ERROR 2  ;TO SLOW/NOT COMPLETE ERROR
7688 025036 005037 001662  CLR INTSET   ;CLEAR INTERRUPT FLAG
7689
7690 025042 104421      TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7691 025044 104004      ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7692
7693 025046 104427      TWAT80 ;WAIT FOR SECOND INIT
7694 025050 104002      ERROR 2  ;TO SLOW/NOT COMPLETE ERROR
7695 025052 104421      TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7696 025054 104004      ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7697
7698 025056 004437 035674  JSR R4,LRLOAD ;LOAD 'L' REGS
7699 025062 000105      CLEAR ;CLEAR
7700 025064 000000      0      ;0 WORDS
7701 025066 000000      0      ;0 IS BUFF ADDRESS
7702 025070      000      .BYTE 0  ;SECTOR 0
7703 025071      000      .BYTE 0  ;TRACK 0
7704 025072 000000      0      ;CYLINDER 0
7705
7706 025074 104417      TLOADRK ;LOAD RK REGS
7707 025076 104423      TWAT16 ;WAIT FOR INTERRUPT
7708 025100 104002      ERROR 2  ;TO SLOW/NOT COMPLETE ERROR
7709
7710 025102 104421      TCHKOP ;CHECK OPERATION FOR ANY ERRORS
  
```

```
7711 025104 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7712
7713 025106 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7714 025112 000131 WRTCHK ;WRTCHK
7715 025114 000000 0 ;0 WORDS
7716 025116 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7717 025120 000 .BYTE 0 ;SECTOR 0
7718 025121 000 .BYTE 0 ;TRACK 0
7719 025122 000000 0 ;CYLINDER 0
7720 025124 052737 000020 001610 BIS #BAI,L.CS2 ;SET BAI FLAG
7721 025132 012700 000621 MOV #401.,R0 ;SET WAIT COUNT
7722
7723 025136 104417 TLOADRK ;LOAD RK REGS
7724 025140 104434 8$: TWAT159 ;WAIT FOR INTERRUPT
7725 025142 000401 BR 9$ ;NO INTERRUPT-SKIP
7726 025144 000403 BR 10$ ;INTERRUPT-SKIP
7727
7728 025146 005300 9$: DEC R0 ;DEC WAIT COUNT
7729 025150 001373 BNE 8$ ;NOT ZERO-LOOP
7730 025152 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7731
7732 025154 104421 10$: TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7733 025154 104004 ERROR 4 ;OR 5, 6, 7, 10 ;REPORT ALL ERRORS
7734 025156 104004
7735
7736 025160 11$:
7737 :*****
7738 :*TEST 74 CONTROLLER TIME OUT
7739 :*
7740 :* SEEK TO CYLINDER 632. ISSUE A RECALIBRATE AND DO NOT
7741 :* WAIT FOR SECOND INTERRUPT. NOW ISSUE A READ HEADER
7742 :* OF CYLINDER 0, TRACK 0. MAKE SURE CONTROLLER TIME
7743 :* OUT SETS.
7744 :* CYLINDER 1456 IS USED FOR THE RK07.
7745 :*
7746 :*****
7747 025160 000004 TST74: SCOPE
7748 025162 012737 000005 001262 MOV #5.,$TIMES ;;DO 5. ITERATIONS
7749 025170 104416 TSSINIT ;CLEAR SUBSYSTEM
7750 025172 104003 ERROR 3 ;BAD INIT ERROR
7751 025174 013737 036120 025216 MOV LSTCYL,1$
7752
7753 025202 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7754 025206 000117 SEEK
7755 025210 000000 0 ;0 WORDS
7756 025212 000000 0 ;0 IS BUFF ADDR
7757 025214 000 .BYTE 0 ;SEC 0
7758 025215 000 .BYTE 0 ;TRK 0
7759 025216 000000 1$: 0 ;CYL 632/1456
7760
7761 025220 104417 TLOADRK ;LOAD RK REGS
7762 025222 104423 TWAT16 ;WAIT FOR INTERRUPT
7763 025224 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7764
7765 025226 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7766 025230 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
```

7767								
7768	025232	005037	001662	CLR	INTSET			;CLEAR INTERRUPT FLAG
7769	025236	104427		TWAT80				;WAIT FOR SECOND INTERRUPT
7770	025240	104002		ERROR	2			;TO SLOW/NOT COMPLETE ERROR
7771	025242	104421		TCHKOP				;CHECK OPERATION FOR ANY ERRORS
7772	025244	104004		ERROR	4 ;OR 5, 6, 7			;REPORT ALL ERRORS
7773								
7774	025246	004437	035674	JSR	R4,LRLOAD			;LOAD 'L' REGS
7775	025252	000105		CLEAR				;CLEAR
7776	025254	000000		0				;0 WORDS
7777	025256	000000		0				;0 IS BUFF ADDRESS
7778	025260	000		.BYTE	0			;SECTOR 0
7779	025261	000		.BYTE	0			;TRACK 0
7780	025262	000000		0				;CYLINDER 0
7781								
7782	025264	104417		TLOADRK				;LOAD RK REGS
7783	025266	104423		TWAT16				;WAIT FOR INTERRUPT
7784	025270	104002		ERROR	2			;TO SLOW/NOT COMPLETE ERROR
7785								
7786	025272	104421		TCHKOP				;CHECK OPERATION FOR ANY ERRORS
7787	025274	104004		ERROR	4 ;OR 5, 6, 7			;REPORT ALL ERRORS
7788								
7789	025276	004437	035674	JSR	R4,LRLOAD			;LOAD 'L' REGS
7790	025302	000113		RECAL				;RECAL
7791	025304	000000		0				;0 WORDS
7792	025306	000000		0				;0 IS BUFF ADDRESS
7793	025310	000		.BYTE	0			;SECTOR 0
7794	025311	000		.BYTE	0			;TRACK 0
7795	025312	000000		0				;CYLINDER 0
7796								
7797	025314	104417		TLOADRK				;LOAD RK REGS
7798	025316	104423		TWAT16				;WAIT FOR INTERRUPT
7799	025320	104002		ERROR	2			;TO SLOW/NOT COMPLETE ERROR
7800								
7801	025322	004437	035674	JSR	R4,LRLOAD			;LOAD 'L' REGS
7802	025326	000125		RDHEAD				;RDHEAD
7803	025330	000000		0				;0 WORDS
7804	025332	000000		0				;0 IS BUFF ADDRESS
7805	025334	000		.BYTE	0			;SECTOR 0
7806	025335	000		.BYTE	0			;TRACK 0
7807	025336	000000		0				;CYLINDER 0
7808								
7809	025340	104417		TLOADRK				;LOAD RK REGS
7810	025342	104436		TWAT2S				;WAIT FOR INTERRUPT
7811	025344	104002		ERROR	2			;TO SLOW/NOT COMPLETE ERROR
7812								
7813	025346	104422		TCHKWE				;CHECK OPERATION WITH EXPECTED ERROR
7814	025350	000000		0				
7815	025352	000000		0				
7816	025354	000002		CTOERR				;CONTROLLER TIME OUT
7817	025356	104004		ERROR	4; OR 5,6,7			;REPORT ANY DISCREPANCIES
7818	025360	104416		TSSINIT				;CLEAR SUBSYSTEM
7819	025362	104003		ERROR	3			;BAD INIT ERROR
7820	025364	005037	001662	CLR	INTSET			;CLEAR INT FLAG
7821	025370	012762	000100 000000	MOV	#IE,RKCS1(R2)			;SET INT ENABLE
7822	025376	104437		TWAT8S				;WAIT FOR SECOND INT

7823 025400 104002

ERROR 2

7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840
7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878

.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  
ERROR 3 ;BAD INIT ERROR
```

000004
012737 000003 001262

104416
104003

004437 035674
000113
000000
000000
000
000
000000

104417
104423
104002

005037 001662
104437
104002

104421
104004

004437 035674
000117
000000
000000
000
000
000002
012737 000020 001616
104417
104423
104002

104416
104003

```
7879 025514 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7880 025520 000123 WRDATA ;WRDATA
7881 025522 177400 -400 ;-400 WORDS
7882 025524 072414 OBUFF ;OBUFF IS BUFF ADDRESS
7883 025526 000 .BYTE 0 ;SECTOR 0
7884 025527 000 .BYTE 0 ;TRACK 0
7885 025530 000001 1 ;CYLINDER 1
7886
7887 025532 104417 TLOADRK ;LOAD RK REGS
7888 025534 104423 TWAT16 ;WAIT FOR INTERRUPT
7889 025536 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7890
7891 025540 104422 TCHKWE ;CHECK OPERATION WITH ERROR
7892 025542 000002 SKIERR ;SEEK INCOMPLETE
7893 025544 000000 0
7894 025546 000000 0
7895 025550 104004 ERROR 4 ;OR 5,6,OR7 ;REPORT ALL DISCREPANCIES
7896
7897 025552 104416 TSSINIT ;CLEAR SUBSYSTEM
7898 025554 104003 ERROR 3 ;BAD INIT ERROR
7899
7900
7901 025556 3$:
7902 025556 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
7903 025562 000101 SELDRV ;SELDRV
7904 025564 000000 0 ;0 WORDS
7905 025566 000000 0 ;0 IS BUFF ADDRESS
7906 025570 000 .BYTE 0 ;SECTOR 0
7907 025571 000 .BYTE 0 ;TRACK 0
7908 025572 000000 0 ;CYLINDER 0
7909
7910 025574 012737 000001 001616 MOV #1,L.MR1 ;SET TO GET STATUS PAIR 1
7911 025602 104417 TLOADRK ;LOAD RK REGS
7912 025604 104423 TWAT16 ;WAIT FOR INTERRUPT
7913 025606 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
7914
7915 025610 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
7916 025612 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
7917
7918 025614 032737 000040 001574 BIT #S.HDHM,T.MR2 ;TEST IF HEADS HOME
7919 025622 001755 BEQ 3$ ;NO - GET STATUS AGAIN
7920
7921 025624 104416 TSSINIT ;CLEAR SUBSYSTEM
7922 025626 104003 ERROR 3 ;BAD INIT ERROR
7923
7924 025630 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET IE
7925
7926 025636 005037 001662 CLR INTSET ;CLEAR INT FLAG
7927 025642 104437 TWAT8S ;WAIT FOR SECOND INTERRUPT
7928 025644 000401 BR 1$
7929 025646 000404 BR 2$
7930
7931 025650 012737 060314 001372 1$: MOV #DH016,DH2N ;"SUBSYSTEM CLEAR TO RESET LIMIT ERROR
7932 ;ALLOWING HEADS TO RELOAD"
7933 025656 104002 ERROR 2
7934
```

7935 025660
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945 025660 000004
7946 025662 012737 000012 001262
7947 025670 104416
7948 025672 104003
7949
7950 025674 004437 035674
7951 025700 000121
7952 025702 177400
7953 025704 070414
7954 025706 000
7955 025707 000
7956 025710 000000
7957
7958 025712 104417
7959
7960 025714 012762 000001 000022
7961
7962 025722 104423
7963 025724 104002
7964
7965 025726 104422
7966 025730 000000
7967 025732 000000
7968 025734 000020
7969 025736 104004
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983 025740 000004
7984 025742 012737 000012 001262
7985 025750 104416
7986 025752 104003
7987
7988 025754 004437 042260
7989 025760 000001
7990 025762 000400

2\$:

```
*****  
*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  
400
```

7991								
7992	025764	004437	035674		JSR	R4,LRLOAD		;LOAD 'L' REGS
7993	025770	000123			WRDATA			;WRDATA
7994	025772	177400			-400			;-400 WORDS
7995	025774	072414			OBUFF			;OBUFF IS BUFF ADDRESS
7996	025776	000			.BYTE	0		;SECTOR 0
7997	025777	000			.BYTE	0		;TRACK 0
7998	026000	000000			0			;CYLINDER 0
7999								
8000	026002	104417			TLOADRK			;LOAD RK REGS
8001	026004	104430			TWAT96			;WAIT FOR INTERRUPT
8002	026006	104002			ERROR	2		;TO SLOW/NOT COMPLETE ERROR
8003								
8004	026010	104421			TCHKOP			;CHECK OPERATION FOR ANY ERRORS
8005	026012	104004			ERROR	4 ;OR 5, 6, 7, 10		;REPORT ALL ERRORS
8006								
8007	026014	004437	042260		JSR	R4,GENCOM		;GENERATE DATA OF ZEROS
8008	026020	000002			2			
8009	026022	000400			400			
8010								
8011	026024	004437	035674		JSR	R4,LRLOAD		;LOAD 'L' REGS
8012	026030	000123			WRDATA			;WRDATA
8013	026032	177630			-150			;-150 WORDS
8014	026034	072414			OBUFF			;OBUFF IS BUFF ADDRESS
8015	026036	000			.BYTE	0		;SECTOR 0
8016	026037	000			.BYTE	0		;TRACK 0
8017	026040	000000			0			;CYLINDER 0
8018								
8019	026042	104417			TLOADRK			;START OPERATION
8020								
8021	026044	005737	001662	1\$:	TST	INTSET		;CHECK IF INTERRUPT HAS OCCURRED
8022	026050	001026			BNE	2\$;YES - MUCH TO SOON. REPORT ERROR
8023	026052	005762	000002		TST	RKWC(R2)		;TEST IF NPR'S DONE
8024	026056	001372			BNE	1\$;NO - LOOP
8025								
8026	026060	052762	100000 000000		BIS	#CCLR,RKCS1(R2)		;CLEAR CONTROLLER (CROWBAR WRITE)
8027								
8028	026066	004437	035674		JSR	R4,LRLOAD		;LOAD 'L' REGS
8029	026072	000121			RDDATA			;RDDATA
8030	026074	177400			-400			;-400 WORDS
8031	026076	070414			IBUFF			;IBUFF IS BUFF ADDRESS
8032	026100	000			.BYTE	0		;SECTOR 0
8033	026101	000			.BYTE	0		;TRACK 0
8034	026102	000000			0			;CYLINDER 0
8035								
8036	026104	104417			TLOADRK			;LOAD RK REGS
8037	026106	104425			TWAT48			;WAIT FOR INTERRUPT
8038	026110	104002			ERROR	2		;TO SLOW/NOT COMPLETE ERROR
8039								
8040	026112	104422			TCHKWE			;CHECK OPERATION WITH ERROR
8041	026114	000000			0			
8042	026116	000003			DCKERR!ECHERR			;DATA CHECK AND ECC HARD
8043	026120	000000			0			
8044	026122	104004			ERROR	4 ;OR 5,6,7		;REPORT ALL DISCREPANCIES
8045								
8046	026124	000402			BR	3\$;SKIP TO EXIT

8047 026126
8048 026126 104421
8049 026130 104004
8050 026132

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

8051
8052
8053
8054
8055
8056
8057
8058
8059
8060
8061
8062

```
*****  
*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  
* TRANSITIONS ON DATA LINE.  
*****
```

8063 026132 000004
8064 026134 012737 000005 001262
8065
8066 026142 104416
8067 026144 104003
8068
8069 026146 004437 035674
8070 026152 000117
8071 026154 000000
8072 026156 000000
8073 026160 000
8074 026161 000
8075 026162 000632
8076
8077 026164 104417
8078 026166 104423
8079 026170 104002
8080 026172 005037 001662
8081 026176 104430
8082 026200 104002
8083
8084 026202 004437 035674
8085 026206 000113
8086 026210 000000
8087 026212 000000
8088 026214 000
8089 026215 000
8090 026216 000000
8091
8092 026220 104417
8093 026222 104423
8094 026224 104002
8095
8096 026226 004437 035674
8097 026232 000125
8098 026234 000000
8099 026236 000000
8100 026240 000
8101 026241 000
8102 026242 000000

```
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  
0 ;CYLINDER 0
```

```

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

```

.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.
*****

```

```

8155 026356 000004 TST101: SCOPE
8156 026360 012737 000012 001262 MOV #10.,$TIMES ;;DO 10. ITERATIONS
8157 026366 104416 TSSINIT ;CLEAR SUBSYSTEM
8158 026370 104003 ERROR 3 ;BAD INIT ERROR

```

```
8159
8160 026372 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8161 026376 000101 SELDRV ;SELDV
8162 026400 000000 0 ;0 WORDS
8163 026402 000000 0 ;0 IS BUFF ADDRESS
8164 026404 000 .BYTE 0 ;SECTOR 0
8165 026405 000 .BYTE 0 ;TRACK 0
8166 026406 000000 0 ;CYLINDER 0
8167
8168 026410 104417 TLOADRK ;LOAD RK REGS
8169 026412 104423 TWAT16 ;WAIT FOR INTERRUPT
8170 026414 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8171
8172 026416 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8173 026420 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8174
8175 026422 104416 TSSINIT ;CLEAR SUBSYSTEM
8176 026424 104003 ERROR 3 ;BAD INIT ERROR
8177
8178 026426 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8179 026432 000101 SELDRV ;SELDV
8180 026434 000000 0 ;0 WORDS
8181 026436 000000 0 ;0 IS BUFF ADDRESS
8182 026440 000 .BYTE 0 ;SECTOR 0
8183 026441 000 .BYTE 0 ;TRACK 0
8184 026442 000000 0 ;CYLINDER 0
8185 026444 012737 000043 001616 MOV #3!DMD,L.MR1 ;SET DIAG MODE AND MESSAGE PAIR 3
8186 026452 005037 001610 CLR L.CS2 ;SELECT DRIVE 0
8187
8188 026456 104417 TLOADRK ;LOAD RK REGS
8189
8190 026460 004437 036316 JSR R4,MCLOCK ;CLOCK THROUGH PHASE ADDRESS 6
8191 026464 001027 1027
8192
8193 026466 042762 000040 000026 BIC #DMD,RKMR1(R2) ;CLEAR MAINTENANCE MODE
8194
8195 026474 104424 TWAT32 ;WAIT FOR INTERRUPT
8196 026476 104002 ERROR 2 ;TO SLOW/NOT COMPLETE
8197
8198 026500 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8199 026502 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8200
8201 *****
8202 *TEST 102 DRIVE OFF TRACK
8203 *
8204 * ISSUE A SUBSYSTEM CLEAR. ISSUE A RECALIBRATE. ISSUE
8205 * OFFSET OF +1200 MICRO-INCHES. PUT CONTROLLER IN DIAGNOSTIC
8206 * MODE. ISSUE A WRITE DATA OF 1 WORD TO CYLINDER 0,
8207 * TRACK 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
8208 * MESSAGES. TURN OFF DIAGNOSTIC MODE. DRIVE OFF TRACK
8209 * SHOULD SET IN DRIVE. REPEAT FOR ALL AVAILIABLE DRIVES.
8210 *****
8211 026504 000004 TST102: SCOPE
8212 026506 012737 000005 001262 MOV #5.,$TIMES ;;DO 5. ITERATIONS
8213 026514 104416 TSSINIT ;CLEAR SUBSYSTEM
8214 026516 104003 ERROR 3 ;BAD INIT ERROR
```



```
8271 026656 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8272 026662 000105 CLEAR ;CLEAR
8273 026664 000000 0 ;0 WORDS
8274 026666 000000 0 ;0 IS BUFF ADDRESS
8275 026670 000 .BYTE 0 ;SECTOR 0
8276 026671 000 .BYTE 0 ;TRACK 0
8277 026672 000000 0 ;CYLINDER 0
8278
8279 026674 104417 TLOADRK ;LOAD RK REGS
8280 026676 104423 TWAT16 ;WAIT FOR INTERRUPT
8281 026700 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8282
8283 026702 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8284 026704 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8285
8286 026706 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8287 026712 000023 23 ;WRITE DATA W/O INTERRUPT ENABLE
8288 026714 177777 -1 ;-1 WORDS
8289 026716 072414 OBUFF ;OBUFF IS BUFF ADDRESS
8290 026720 000 .BYTE 0 ;SECTOR 0
8291 026721 000 .BYTE 0 ;TRACK 0
8292 026722 000000 0 ;CYLINDER 0
8293 026724 012737 000040 001616 MOV #DMD,L.MR1 ;SET DIAGNOSTIC MODE
8294
8295 026732 104417 TLOADRK
8296
8297 026734 004437 036316 JSR R4,MCLOCK ;CLOCK THROUGH SEEK & DRIVE CLEAR
8298 026740 001064 1064
8299
8300 026742 005062 000026 CLR RKMR1(R2) ;CLEAR DIAGNOSTIC MODE
8301 026746 104426 TWAT64 ;WAIT FOR INTERRUPT
8302 026750 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
8303 026756 104423 TWAT16 ;STALL FOR 16 MS
8304 026760 000240 NOP
8305 026762 000240 NOP
8306
8307 026764 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8308 026770 000101 SELDRV ;SELDRV
8309 026772 000000 0 ;0 WORDS
8310 026774 000000 0 ;0 IS BUFF ADDRESS
8311 026776 000 .BYTE 0 ;SECTOR 0
8312 026777 000 .BYTE 0 ;TRACK 0
8313 027000 000000 0 ;CYLINDER 0
8314 027002 005037 001616 CLR L.MR1 ;RESET DIAG MODE
8315
8316 027006 104417 TLOADRK ;LOAD RK REGS
8317 027010 104423 TWAT16 ;WAIT FOR INTERRUPT
8318 027012 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8319
8320 027014 104422 TCHKWE ;CHECK OPERATION WITH ERROR EXPECTED
8321 027016 000400 DROTERR ;DRIVE OFF TRACK
8322 027020 000000 0
8323 027022 000000 0
8324 027024 104004 ERROR 4; OR 5,6,7 ;REPORT ANY DISCREPANCIES
8325
8326 ;*****
; *TEST 103 FILE UNSAFE
```

```
8327
8328
8329
8330
8331
8332
8333
8334
8335
8336
8337
8338
8339 027026 000004
8340 027030 012737 000005 001262
8341 027036 012737 177777 001676
8342 027044 104416
8343 027046 104003
8344
8345 027050 004437 035674
8346 027054 000113
8347 027056 000000
8348 027060 000000
8349 027062 000
8350 027063 000
8351 027064 000012
8352
8353 027066 104417
8354 027070 104423
8355 027072 104002
8356
8357 027074 104421
8358 027076 104004
8359
8360 027100 005037 001662
8361 027104 104437
8362 027106 104002
8363
8364 027110 104421
8365 027112 104004
8366
8367 027114 004437 035674
8368 027120 000105
8369 027122 000000
8370 027124 000000
8371 027126 000
8372 027127 000
8373 027130 000012
8374
8375 027132 104417
8376 027134 104423
8377 027136 104002
8378
8379 027140 104421
8380 027142 104004
8381
8382 027144 004437 035674
```

TST103: SCOPE
MOV #5,\$TIMES ;DO 5. ITERATIONS
MOV #-1,REFMT ;SET REFORMAT SWITCH
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
12 ;CYLINDER 12
TLOADRK ;LOAD RK REGS
TWAT16 ;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
CLR INTSET ;CLEAR INT FLAG
TWAT8S ;WAIT FOR SECOND INT
ERROR 2
TCHKOP ;CHECK OPERATION FOR ANY ERRORS
ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
JSR R4,LRLOAD ;LOAD 'L' REGS
CLEAR ;CLEAR
0 ;0 WORDS
0 ;0 IS BUFF ADDRESS
.BYTE 0 ;SECTOR 0
.BYTE 0 ;TRACK 0
12 ;CYLINDER 12
TLOADRK ;LOAD RK REGS
TWAT16 ;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,LRLOAD ;LOAD 'L' REGS

8383	027150	010125			RDHEAD!CFMT		;RDHEAD!CFMT
8384	027152	000000			0		;0 WORDS
8385	027154	000000			0		;0 IS BUFF ADDRESS
8386	027156	000			.BYTE 0		;SECTOR 0
8387	027157	000			.BYTE 0		;TRACK 0
8388	027160	000012			12		;CYLINDER 12
8389							
8390	027162	104417			TLOADRK		;LOAD RK REGS
8391	027164	104424			TWAT32		;WAIT FOR INTERRUPT
8392	027166	104002			ERROR 2		;TO SLOW/NOT COMPLETE ERROR
8393							
8394	027170	104421			TCHKOP		;CHECK OPERATION FOR ANY ERRORS
8395	027172	104004			ERROR 4 ;OR 5, 6, 7		;REPORT ALL ERRORS
8396							
8397	027174	004437	035674		JSR R4,LRLOAD		;LOAD 'L' REGS
8398	027200	000101			SELDRV		;SELDRV
8399	027202	000000			0		;0 WORDS
8400	027204	000000			0		;0 IS BUFF ADDRESS
8401	027206	000			.BYTE 0		;SECTOR 0
8402	027207	000			.BYTE 0		;TRACK 0
8403	027210	000012			12		;CYLINDER 12
8404							
8405	027212	104417			TLOADRK		;LOAD RK REGS
8406	027214	104423			TWAT16		;WAIT FOR INTERRUPT
8407	027216	104002			ERROR 2		;TO SLOW/NOT COMPLETE ERROR
8408							
8409	027220	004437	035674		JSR R4,LRLOAD		;LOAD 'L' REGS
8410	027224	000127			WRHEAD		;WRHEAD
8411	027226	177777			-1		;-1 WORDS
8412	027230	072414			OBUFF		;OBUFF IS BUFF ADDRESS
8413	027232	000			.BYTE 0		;SECTOR 0
8414	027233	000			.BYTE 0		;TRACK 0
8415	027234	000012			12		;CYLINDER 12
8416	027236	012737	000040 001616		MOV #DMD,L.MR1		;SET DIAGNOSTIC-MODE
8417							
8418	027244	104417			TLOADRK		;LOAD RK REGS
8419	027246	004437	036316		JSR R4,MCLOCK		;CLOCK THROUGH SEEK AND DRIVE CLEAR
8420	027252	001064			1064		
8421							
8422	027254	052762	000200 000026		BIS #MIND,RKMR1(R2)		;SET INDEX
8423							
8424	027262	004437	036316		JSR R4,MCLOCK		;CLOCK INDEX
8425	027266	001001			1001		
8426							
8427	027270	042762	000200 000026		BIC #MIND,RKMR1(R2)		;CLEAR INDEX
8428							
8429	027276	004437	036316		JSR R4,MCLOCK		;CLOCK CLEAR
8430	027302	001001			1001		
8431							
8432	027304	005062	000026		CLR RKMR1(R2)		;CLEAR DIAGNOSTIC MODE
8433							
8434	027310	104426			TWAT64		;WAIT FOR INTERRUPT
8435	027312	104002			ERROR 2		;TO SLOW/NOT COMPLETE ERROR
8436							
8437	027314	104421			TCHKOP		;CHECK OPERATION FOR ANY ERRORS
8438	027316	104004			ERROR 4 ;OR 5, 6, 7, 10		;REPORT ALL ERRORS

```
8439
8440 027320 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8441 027324 000101 SELDRV ;SELDRV
8442 027326 000000 0 ;0 WORDS
8443 027330 000000 0 ;0 IS BUFF ADDRESS
8444 027332 000 .BYTE 0 ;SECTOR 0
8445 027333 000 .BYTE 0 ;TRACK 0
8446 027334 000012 12 ;CYLINDER 12
8447
8448 027336 005037 001616 CLR L.MR1 ;CLEAR DIAG MODE
8449
8450 027342 104417 TLOADRK ;LOAD RK REGS
8451 027344 104423 TWAT16 ;WAIT FOR INTERRUPT
8452 027346 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8453
8454 027350 104422 TCHKWE ;CHECK OPERATION WITH EXPECTED ERROR
8455 027352 040400 UNSERR!DROTERR ;UNSAFE AND DRIVE OFF TRACK
8456 027354 000000 0
8457 027356 000000 0
8458 027360 104004 ERROR 4; OR 5,6,7 ;REPORT ANY DISCREPANCIES
8459
8460 027362 104416 TSSINIT ;CLEAR SUBSYSTEM
8461 027364 104003 ERROR 3 ;BAD INIT ERROR
8462
8463 027366 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
8464 027372 000101 SELDRV ;SELDRV
8465 027374 000000 0 ;0 WORDS
8466 027376 000000 0 ;0 IS BUFF ADDRESS
8467 027400 000 .BYTE 0 ;SECTOR 0
8468 027401 000 .BYTE 0 ;TRACK 0
8469 027402 000012 12 ;CYLINDER 12
8470
8471 027404 012737 000001 001616 MOV #1,L.MR1 ;SET MESSAGE SELECT ONE
8472
8473 027412 1$:
8474 027412 104417 TLOADRK ;LOAD RK REGS
8475 027414 104423 TWAT16 ;WAIT FOR INTERRUPT
8476 027416 104002 ERROR 2 ;TO SLOW/NOT COMPLETE ERROR
8477
8478 027420 104421 TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8479 027422 104004 ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8480
8481 027424 032737 000040 001574 BIT #S.HDHM,T.MR2 ;TEST IF HEADS HOME
8482 027432 001767 BEQ 1$
8483
8484 027434 104416 TSSINIT ;CLEAR SUBSYSTEM
8485 027436 104003 ERROR 3 ;BAD INIT ERROR
8486
8487 027440 005037 001662 CLR INTSET ;CLEAR INT FLAG
8488 027444 104434 TWAT159 ;WAIT FOR APPROX 160 MS
8489 027446 000240 NOP ;DON'T CARE ERROR RETURN
8490
8491 027450 104416 TSSINIT ;CLEAR SUBSYSTEM
8492 027452 104003 ERROR 3 ;BAD INIT ERROR
8493
8494 027454 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
```

```
8495  
8496 027462 104437          TWAT8S          ;WAIT FOR SECOND INTERRUPT FROM HDS LOADED  
8497 027464 104002          ERROR 2  
8498  
8499 027466 005037 001616    CLR L.MR1        ;CLEAR MR1  
8500  
8501 027472 004437 035674    JSR R4,LRLOAD   ;LOAD 'L' REGS  
8502 027476 000105          CLEAR          ;CLEAR  
8503 027500 000000          0              ;0 WORDS  
8504 027502 000000          0              ;0 IS BUFF ADDRESS  
8505 027504 000          .BYTE 0         ;SECTOR 0  
8506 027505 000          .BYTE 0         ;TRACK 0  
8507 027506 000012          12             ;CYLINDER 12  
8508  
8509 027510 104417          TLOADRK        ;LOAD RK REGS  
8510 027512 104423          TWAT16        ;WAIT FOR INTERRUPT  
8511 027514 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR  
8512  
8513 027516 104421          TCHKOP        ;CHECK OPERATION FOR ANY ERRORS  
8514 027520 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS  
8515  
8516 027522 004437 042260    JSR R4,GENCOM  ;BUILD HEADERS  
8517 027526 001200          1200  
8518  
8519 027530 004437 035674    JSR R4,LRLOAD   ;LOAD 'L' REGS  
8520 027534 000127          WRHEAD        ;WRHEAD  
8521 027536 177676          -102          ;-102 WORDS  
8522 027540 072414          OBUF          ;OBUF IS BUFF ADDRESS  
8523 027542 000          .BYTE 0         ;SECTOR 0  
8524 027543 000          .BYTE 0         ;TRACK 0  
8525 027544 000012          12             ;CYLINDER 12  
8526  
8527 027546 104417          TLOADRK        ;LOAD RK REGS  
8528 027550 104426          TWAT64        ;WAIT FOR INTERRUPT  
8529 027552 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR  
8530  
8531 027554 104421          TCHKOP        ;CHECK OPERATION FOR ANY ERRORS  
8532 027556 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS  
8533  
8534  
8535  
8536  
8537  
8538  
8539  
8540  
8541  
8542  
8543  
8544  
8545  
8546 027560 000004          TST104: SCOPE  
8547 027562 012737 000001 001262  MOV #1,$TIMES  ;;DO 1 ITERATION  
8548  
8549 027570 104416          TSSINIT       ;CLEAR SUBSYSTEM  
8550 027572 104003          ERROR 3       ;BAD INIT ERROR
```

```
*****  
*TEST 104 DUMMY TEST FOR PREVIOUS TEST EXIT  
* THIS TEST IS PRESENT TO MAKE $SWOBTB 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.  
*****
```

```
8551
8552 027574 013762 001626 000010      MOV   DRVNUM,RKCS2(R2) ;LOAD DRIVE NUMBER
8553 027602 005037 036126              CLR   HOLD1
8554 027606 012737 000001 036124      MOV   #1,HOLD
8555 027614 053737 001720 036124      BIS   DTYPE,HOLD
8556 027622 013762 036124 000000      MOV   HOLD,RKCS1(R2) ;SELECT THE DRIVE
8557 027630 032762 000200 000012 1$:    BIT   #DRDY,RKDS(R2) ;TEST IF DRIVE READY
8558 027636 001004              BNE   2$ ;BR IF YES
8559 027640 005237 036126              INC   HOLD1 ;ELSE TRY AGAIN
8560 027644 001371              BNE   1$
8561 027646 104002              ERROR  2 ;NO RDY
8562
8563 027650              2$:
8564 027650 104416              TSSINIT ;CLEAR SUBSYSTEM
8565 027652 104003              ERROR  3 ;BAD INIT ERROR
8566
8567 027654 004437 042260      JSR   R4,GENCOM ;GENERATE HEADERS FOR CYL 0
8568 027660 001200              1200
8569
8570 027662 004437 035674      JSR   R4,LRLOAD ;LOAD 'L' REGS
8571 027666 000127              WRHEAD ;WRHEAD
8572 027670 177676              -102 ;-102 WORDS
8573 027672 072414              OBUFF ;OBUFF IS BUFF ADDRESS
8574 027674 000              .BYTE 0 ;SECTOR 0
8575 027675 000              .BYTE 0 ;TRACK 0
8576 027676 000000              0 ;CYLINDER 0
8577
8578 027700 104417              TLOADRK ;LOAD RK REGS
8579 027702 104426              TWAT64 ;WAIT FOR INTERRUPT
8580 027704 104002              ERROR  2 ;TO SLOW/NOT COMPLETE ERROR
8581
8582 027706 104421              TCHKOP ;CHECK OPERATION FOR ANY ERRORS
8583 027710 104004              ERROR  4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8584 027712 005037 001676      CLR   REFORMAT ;CLEAR REFORMAT SWITCH
8585 .SBTTL **MULTI-DRIVE OPERATIONS
8586 027716 000004      NEWDRV: SCOPE
8587 027720 012737 000001 001262      MOV   #1,$TIMES ;DO ONLY ONCE
8588 027726 032737 000200 001630      BIT   #BIT7,DRVBIT ;WERE WE TESTING DRIVE 7?
8589 027734 001026              BNE   3$ ;YES-SKIP
8590
8591 027736 005237 001626      1$:    INC   DRVNUM ;BUMP TO NEXT SEQUENTIAL ADDRESS
8592 027742 006337 001630              ASL   DRVBIT ;BUMP DRIVEBIT TO THAT POSITION
8593 027746 033737 001630 001354      BIT   DRVBIT,$DEVN ;IS THIS DRIVE TO BE TESTED?
8594 027754 001005              BNE   2$ ;YES-EXIT
8595 027756 032737 000400 001630      BIT   #BIT8,DRVBIT ;ALL DRIVES TESTED?
8596 027764 001012              BNE   3$ ;YES-EXIT
8597 027766 000763              BR    1$ ;ELSE CHECK NEXT DRIVE AVAILABLE
8598
8599 027770 112737 000004 001102 2$:    MOVB  #4,$STNM ;SET TEST NUMBER FOR REPORTS
8600 027776 013701 001626              MOV   DRVNUM,R1
8601 030002 004737 036150              JSR   PC,GTP1 ;GET DRV TYP FOR NEXT DRV TO TST
8602 030006 000137 004720              JMP   TSTLUP ;GO TO TEST LOOP TO CHECK THIS DRIVE
8603 030012 005037 001630      3$:    CLR   DRVBIT ;CLEAR DRIVE BIT
8604 030016 005037 001626              CLR   DRVNUM ;CLEAR DRIVE NUMBER
8605
8606
```

:::.....

```
8607      ;*TEST 105      RESET ATTENTIONS WITH UNIBUS INIT
8608      ;*
8609      ;*      DO A RECALIBRATE ON ALL AVAILABLE DRIVES.
8610      ;*      ISSUE A RESET.  MAKE SURE ALL ATTENTION RESET.
8611      ;*
8612      ;*****
8613 030022 000004      TST105: SCOPE
8614 030024 012737 000012 001262      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
8615 030032 005000      CLR      R0              ;CLEAR DRIVE POSITION COUNTER
8616 030034 012701 000001      MOV      #1,R1          ;PRESET BIT FOR POSITION 0 IN TESTING FOR AVAIL
8617 030040 013703 001354      MOV      $DEVN,R3      ;GET DEVICE MAP
8618 030044 104416      TSSINIT          ;CLEAR SUBSYSTEM
8619 030046 104003      ERROR      3          ;BAD INIT ERROR
8620 030050 030103      1$: BIT      R1,R3      ;TEST IF THIS DRIVE AVAILABLE
8621 030052 001006      BNE      2$          ;YES-SKIP TO SEEK
8622 030054 006301      3$: ASL      R1          ;SHIFT DRIVE SELECT BIT
8623 030056 005200      INC      R0          ;BUMP DRIVE POSITION COUNTER
8624 030060 032701 000400      BIT      #BIT8,R1      ;ALL DRIVE POSITIONS CHECKED
8625 030064 001771      BEQ      1$          ;NO-LOOP
8626 030066 000443      BR      4$          ;SKIP TO RESET
8627
8628 030070 010037 001610      2$: MOV      R0,L.CS2      ;LOAD DRIVE NUMBER
8629 030074 004737 036134      JSR      PC,GTYP0
8630 030100 012737 000113 001600      MOV      #RECAL,L.CS1  ;LOAD RECALIBRATE
8631
8632 030106 104417      TLOADRK          ;LOAD RK REGS
8633 030110 104423      TWAT16          ;WAIT FOR INTERRUPT
8634 030112 104002      ERROR      2          ;TO SLOW/NOT COMPLETE ERROR
8635
8636 030114 005037 001662      CLR      INTSET      ;CLEAR INTERRUPT FLAG
8637 030120 012705 000764      MOV      #500.,R5     ;SET COUNT FOR 8 SECONDS
8638 030124 012762 000012 000000      MOV      #12,RKCS1(R2) ;RESET INTERRUPT ENABLE
8639 030132 016237 000016 001556 12$: MOV      RKASOF(R2),T.ASOF ;GET ATTENTION REGISTER
8640 030140 113704 001557      MOV      T.ASOF+1,R4  ;ADJUST FOR CHECK OF ATTENTION
8641 030144 042704 177400      BIC      #177400,R4   ;CLEAR UNUSED BITS
8642 030150 030104      BIT      R1,R4        ;CHECK IF ATT SET FROM DRIVE RECAL'ED
8643 030152 001006      BNE      10$         ;YES - SKIP
8644
8645 030154 104423      TWAT16          ;WAIT FOR 16 MS
8646 030156 000240      NOP              ;DON'T CARE RETURNS
8647 030160 000240      NOP
8648 030162 005305      DEC      R5          ;TATOL WAIT TIME IS 8 SECONDS
8649 030164 001362      BNE      12$         ;CHECK ATTENTION EACH 16 MS
8650 030166 104002      ERROR      2          ;REPORT IF NO ATTENTION IN 8 SEC
8651
8652 030170      10$:
8653 030170 104421      TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8654 030172 104004      ERROR      4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8655
8656 030174 000727      BR      3$          ;LOOP FOR NEXT DRIVE
8657 030176 000005      4$: RESET          ;UNIBUS RESET
8658 030200 004737 045146      JSR      PC,$TKINT    ;RESET KEYBOARD INTERRUPT ENABLE
8659
8660 030204 012701 000031      5$: MOV      #25.,R1      ;DO A SHORT DELAY
8661 030210 005301      DEC      R1
8662 030212 001376      BNE      5$
```

```

8663 030214 004737 034522      JSR    PC,OPTTST      ;SET UP OPTIONS
8664                                TGETRK                ;GET RK611 REGS
8665 030220 104420                                TSTB    T.ASOF+1      ;ALL ATTENTION RESET?
8666                                BEQ     6$             ;YES-SKIP
8667 030222 105737 001557      MOV     #EMDA,EM12N   ;'DRIVE ATT NOT RESET RESULT OF
8668 030226 001407      MOV     #EMUR,DF011A ;UNIBUS RESET''
8669                                ERROR   12
8670 030230 012737 056540 001470
8671 030236 012737 056664 061022
8672 030244 104012
8673
8674 030246
8675
8676
8677
8678
8679
8680
8681
8682
8683 030246 000004      6$:
8684 030250 012737 000012 001262  ;*****
8685 030256 005000      ;*TEST 106      RESET ATTENTIONS WITH SUBSYSTEM CLEAR
8686 030260 C12701 000001      ;*
8687 030264 013703 001354      ;*      DO A RECALIBRATE ON ALL AVAILABLE DRIVES.
8688 030270 104416      ;*      ISSUE A SUBSYSTEM CLEAR.  MAKE SURE ALL ATTENTIONS
8689 030272 104003      ;*      RESET.
8690 030274 030103      ;*****
8691 030276 001006      TST106: SCOPE
8692 030300 006301      MOV     #10.,$TIMES  ;;DO 10. ITERATIONS
8693 030302 005200      CLR     R0            ;CLEAR DRIVE POSITION COUNTER
8694 030304 032701 000400      MOV     #1,R1         ;PRESET TO TEST POSITION 0
8695 030310 001771      MOV     $DEVN,R3     ;CUT DEVICE MAP
8696 030312 000443      TSSINIT              ;CLEAR SUBSYSTEM
8697                                ERROR   3             ;BAD INIT ERROR
8698 030314 010037 001610      1$: BIT     R1,R3     ;THIS DRIVE AVAILABLE?
8699 030320 004737 036134      BNE    2$            ;YES-SKIP TO SEEK
8700 030324 012737 000113 001600      3$: ASL     R1         ;SHIFT TO NEXT DRIVE POSITION
8701 030332 104417      INC     R0            ;DUMP POSITION COUNTER
8702 030334 104423      BIT     #BIT8,R1     ;ALL POSITIONS CHECKED
8703 030336 104002      BEQ    1$            ;NO-LOOP
8704                                BR     4$            ;YES-SKIP TO CLEAR
8705 030340 005037 001662      2$: MOV     R0,L.CS2   ;LOAD DRIVE NUMBER
8706 030344 012705 000764      JSR    PC,GTYP0
8707 030350 012762 000012 000000      MOV     #RECAL,L.CS1 ;LOAD RECALIBRATE
8708 030356 016237 000016 001556 12$: TLOADRK ;LOAD RK REGS
8709 030364 113704 001557      TWAT16 ;WAIT FOR INTERRUPT
8710 030370 042704 177400      ERROR   2           ;TO SLOW/NOT COMPLETE ERROR
8711 030374 030104      CLR     INTSET        ;CLEAR INT FLAG
8712 030376 001006      MOV     #500.,R5     ;SET COUNT FOR 8 SECONDS
8713                                MOV     #12,RKCS1(R2) ;RESET INTERRUPT ENABLE
8714                                MOV     RKASOF(R2),T.ASOF ;GET ATTENTION REGISTER
8715                                MOVVB  T.ASOF+1,R4    ;ADJUST FOR CHECK OF ATTENTION
8716                                BIC     #177400,R4   ;CLEAR UNUSED BITS
8717                                BIT     R1,R4         ;CHECK IF ATT SET FROM DRIVE RECAL'ED
8718                                BNE    10$          ;YES - SKIP
8719                                TWAT16              ;WAIT FOR 16 MS
8720                                NOP                    ;DON'T CARE RETURNS
8721                                NOP
8722                                DEC     R5            ;TATOL WAIT TIME IS 8 SECONDS
8723                                BNE    12$          ;CHECK ATTENTION EACH 16 MS
    
```



```
8719 030412 104002          ERROR 2          ;REPORT IF NO ATTENTION IN 8 SEC
8720
8721 030414          10$:
8722 030414 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8723 030416 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8724
8725 030420 000727          BR 3$          ;LOOP FOR NEXT DRIVE
8726
8727 030422 052762 000040 000010 4$:  BIS #SCLR,RKCS2(R2) ;DO SUBSYSTEM CLEAR
8728 030430 012701 000031          MOV #25.,R1      ;DO A SHORT DELAY
8729 030434 005301          5$:  DEC R1
8730 030436 001376          BNE 5$
8731
8732 030440 104420          'TGETRK          ;GET RK611 REGS
8733
8734 030442 105737 001557          TSTB T.ASOF+1    ;TEST ALL ATTENTION RESET
8735 030446 001407          BEQ 6$          ;YES-SKIP
8736
8737 030450 012737 056540 001470          MOV #EMDA,EM12N ;'DRIVE ATT NOT RESET AS RESULT OF
8738 030456 012737 056754 061022          MOV #EMSCLR,DF011A ;SUBSYSTEM CLEAR''
8739 030464 104012          ERROR 12
8740
8741 030466          6$:
8742          ;*****
8743          ;*TEST 107          SVAL AND ATTENTION FROM OTHER DRIVE
8744          ;*
8745          ;*          DO A RECALIBRATE ON ONE AVAILABLE DRIVE. DO A SELECT
8746          ;*          ON ANOTHER AVAILABLE DRIVE. MAKE SURE STATUS VALID
8747          ;*          IS SET. WAIT FOR SECOND INTERRUPT FROM RECALIBRATE
8748          ;*          MAKE SURE STATUS VALID REMAINS SET AND DRIVE STATUS
8749          ;*          CHANGE REMAINS RESET.
8750          ;*
8751          ;*          REPEAT FOR ALL COMBINATIONS OF TWO AVAILIABLE DRIVES.
8752          ;*
8753          ;*          NOTE: THIS TEST WILL ONLY BE DONE IF AT LEAST
8754          ;*          TWO DRIVES ARE AVAILABLE.
8755          ;*
8756          ;*****
8757 030466 000004          TST107: SCOPE
8758 030470 012737 000012 001262          MOV #10.,$TIMES ;:DO 10. ITERATIONS
8759 030476 013746 001354          MOV $DEVN,-(SP) ;:PUT DEVICE MAP ON STACK
8760 030502 004437 036242          JSR R4,BITCNT   ;:COUNT NUMBER OF BITS(# OF DRIVES)
8761 030506 022627 000001          CMP (SP)+,#1   ;:COMPARE TO 1
8762 030512 101007          BHI 2$         ;:SKIP IF MORE THAN 1
8763 030514 005737 001304          TST $PASS      ;:CHECK IF PASS 0
8764 030520 001002          BNE 1$         ;:NO-SKIP
8765
8766 030522 104401 052574          TYPE ,OPR018   ;:'OVERLAPPED OPERATION BYPASSED''
8767 030526 000137 031772          1$:  JMP $EOP        ;:GET OUT.
8768
8769 030532 012737 177777 001224 2$:  MOV #-1,$TMP1   ;:SET LOOP CONTROL FLAG
8770 030540 013705 001354          MOV $DEVN,R5   ;:GET DEVICE MAP
8771 030544 005000          CLR R0         ;:CLEAR FOR DRIVE #A
8772 030546 005001          CLR R1         ;:CLEAR FOR DRIVE #B
8773 030550 012703 000001          MOV #1,R3      ;:SET DRIVE POSITION A
8774 030554 012704 000001          MOV #1,R4      ;:SET DRIVE POSITION B
```

```

8775 030560 012737 030570 001110      MOV    #3$,SLPERR      ;SET LOCAL LOOP ON ERROR
8776 030566 000503                    BR      11$           ;GO SET UP POINTERS
8777
8778 030570      3$:
8779 030570 104416      TSSINIT                ;CLEAR SUBSYSTEM
8780 030572 104003      ERROR    3           ;BAD INIT ERROR
8781
8782 030574 010037 001610      MOV    R0,L.CS2       ;LOAD DRIVE A
8783 030600 010037 001626      MOV    R0,DRVNUM      ;LOAD FOR REPORT
8784 030604 004737 036134      JSR    PC,GTYP0
8785 030610 012737 000113 001600      MOV    #RECAL,L.CS1   ;LOAD RECAL
8786
8787 030616 104417      TLOADRK                ;LOAD RK REGS
8788 030620 104423      TWAT16                ;WAIT FOR INTERRUPT
8789 030622 104002      ERROR    2           ;TO SLOW/NOT COMPLETE ERROR
8790 030624 104421      TCHKOP                ;CHECK OPERATION FOR ANY ERRORS
8791 030626 104004      ERROR    4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8792
8793 030630 005037 001662      CLR    INTSET         ;CLEAR INT FLAG
8794
8795 030634 010137 001610      MOV    R1,L.CS2       ;LOAD DRIVE B
8796 030640 010137 001626      MOV    R1,DRVNUM      ;LOAD FOR REPORT
8797 030644 004737 036150      JSR    PC,GTYP1
8798 030650 012737 000101 001600      MOV    #SELDRV,L.CS1  ;LOAD DRIVE SELECT
8799
8800 030656 104417      TLOADRK                ;LOAD RK REGS
8801 030660 104423      TWAT16                ;WAIT FOR INTERRUPT
8802 030662 104002      ERROR    2           ;TO SLOW/NOT COMPLETE ERROR
8803
8804 030664 104421      TCHKOP                ;CHECK OPERATION FOR ANY ERRORS
8805 030666 104004      ERROR    4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8806
8807 030670 032737 100000 001552      BIT    #SVAL,T.DS     ;CHECK IF STATUS VALID SET
8808 030676 001007      BNE    4$             ;YES - SKIP
8809 030700 012737 056354 001460      MOV    #EMSVAL,EM11N  ;"STATUS VALID NOT SET RESULT OF
8810 030706 012737 050160 061022      MOV    #OPER00,DF011A ;DRIVE SELECT"
8811 030714 104011      ERROR    11
8812
8813 030716 005037 001662      4$: CLR    INTSET         ;CLEAR INT FLAG
8814 030722 104436      TWAT2S                ;WAIT FOR SEEK COMPLETE INTERRUPT
8815 030724 000401      BR      44$          ;NONE RECEIVED - SKIP
8816 030726 000406      BR      55$          ;RECEIVED - SKIP
8817
8818 030730 010037 001626      44$: MOV    R0,DRVNUM      ;SET DRIVE FOR REPORT
8819 030734 012737 060411 001372      MOV    #DH017,DH2N    ;"COMMAND - SELECT AFTER RECAL"
8820 030742 104002      ERROR    2
8821
8822 030744 104420      55$: TGETRK                ;GET RK REGS
8823 030746 032737 100000 001552      BIT    #SVAL,T.DS     ;TEST IF SVAL STILL SET
8824 030754 001007      BNE    5$             ;YES - SKIP
8825
8826 030756 012737 056354 001510      MOV    #EMSVAL,EM14N  ;"STATUS VALID RESET RESULT OF
8827 030764 012737 060434 061022      MOV    #DH018,DF011A ;RECAL COMPLETE ATTENTION AFTER SEL"
8828 030772 104014      ERROR    14
8829
8830 030774 104415      5$: SCOP1             ;LOCAL LOOP TO 3$
  
```

```

8831
8832
8833
8834
8835
8836
8837
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847 030776 005237 001224 11$: INC $TMP1 ;INCREMENT PASS CONTROL
8848 031002 001024 BNE 16$ ;SKIP IF NOT ZERO
8849 ;(IT WILL BE ZERO ON THE 1ST PASS)
8850
8851 031004 030305 12$: BIT R3,R5 ;TEST IF BIT POSITION FOR A AT AVAIL DRIVE
8852 031006 001006 BNE 13$ ;YES-SKIP
8853
8854 031010 005200 22$: INC R0 ;BUMP R0 (DRIVE A)
8855 031012 006303 ASL R3 ;SHIFT DRIVE SELECT BIT ONE POSITION
8856 031014 032703 000400 BIT #BIT8,R3 ;IF BIT 8 IS SET, ALL DRIVES HAVE
8857 031020 001771 BEQ 12$ ;BEEN CHECKED; IF NOT CHECK NEXT POSITION
8858 031022 000464 BR 50$ ;DONE-EXIT
8859
8860 031024 010001 13$: MOV R0,R1 ;SET DRIVE B TO THE SAME AS A
8861 031026 010304 MOV R3,R4
8862 031030 005201 14$: INC R1 ;BUMP R1 (DRIVE B)
8863 031032 006304 ASL R4 ;SHIFT SELECTOR BIT ONE POSITION
8864 031034 030405 BIT R4,R5 ;IS THIS DRIVE AVAIL?
8865 031036 001004 BNE 15$ ;YES-SKIP
8866 031040 032704 000400 BIT #BIT8,R4 ;WERE ALL POSITIONS CHECKED?
8867 031044 001771 BEQ 14$ ;NO-LOOP
8868 031046 000452 BR 50$ ;DONE-EXIT
8869
8870 031050 000137 030570 15$: JMP 3$ ;GO DO THE TEST ON THE DRIVE A & B
8871 ;CONTAINED IN R0 & R1
8872 031054 032737 000001 001224 16$: BIT #BIT0,$TMP1 ;IS PASS FLAGS ODD?
8873 031062 001410 BEQ 17$ ;NO-SKIP
8874
8875 031064 010046 MOV R0,-(SP) ;
8876 031066 010346 MOV R3,-(SP) ;SWAP R0 & R1, R3 & R4
8877 031070 010403 MOV R4,R3 ;TO EXCHANGE DRIVE A & B
8878 031072 010100 MOV R1,R0
8879 031074 012604 MOV (SP)+,R4
8880 031076 012601 MOV (SP)+,R1
8881 031100 000137 030570 JMP 3$ ;REPEAT TEST ON THIS COMBO.
8882
8883 031104 032737 000002 001224 17$: BIT #BIT1,$TMP1 ;TEST IF PASS FLAGS AT HALF MODULE 4?
8884 031112 001410 BEQ 19$ ;NO-SKIP TO BUMP DRIVE B
8885 031114 005200 18$: INC R0 ;BUMP DRIVE A
8886 031116 006303 ASL R3 ;SHIFT DRIVE SELECT BIT

```

```
8887 031120 030305 BIT R3,R5 ;AVAILABLE?
8888 031122 001014 BNE 20$ ;YES-SKIP
8889 031124 032703 000400 BIT #BIT8,R3 ;ALL CHECKED?
8890 031130 001771 BEQ 18$ ;NO-SKIP
8891 031132 000412 BR 21$ ;GO TO NEXT PASS
8892
8893 031134 005201 19$: INC R1 ;BUMP DRIVE B
8894 031136 006304 ASL R4 ;SHIFT DRIVE SELECT BIT
8895 031140 030405 BIT R4,R5 ;AVAILABLE?
8896 031142 001004 BNE 20$ ;YES-SKIP
8897 031144 032704 000400 BIT #BIT8,R4 ;ALL CHECKED?
8898 031150 001771 BEQ 19$ ;NO-LOOP
8899 031152 000404 BR 23$ ;YES-SKIP TO NEXT PASS
8900
8901 031154 000137 030570 20$: JMP 3$ ;GO TEST THIS COMBO
8902
8903 031160 010100 21$: MOV R1,R0 ;SET DRIVE 0 TO LOW POSITION THIS PASS
8904 031162 010403 MOV R4,R3 ;SET SELECT BITS TO AGREE
8905 031164 005037 001224 23$: CLR $TMP1 ;CLEAR PASS FLAGS
8906 031170 000137 031010 JMP 22$ ;GO SET UP A & B
8907 031174
8908
8909 *****
8910 :*TEST 110 OVERLAPPED OPERATIONS
8911 :*
8912 :* DO A RECALIBRATE ON BOTH DRIVE A AND DRIVE B. ISSUE A
8913 :* SEEK ON DRIVE A TO CYLINDER 1. IMMEDIATELY ISSUE A WRITE
8914 :* DATA TO CYLINDER 100, TRACK 0, HEAD 0 ON DRIVE B.
8915 :* AT THE END OF THE DATA TRANSFER NO ERRORS SHOULD
8916 :* BE SET AND DRIVE A HAS ATTENTION SET.
8917 :*
8918 :* REPEAT FOR ALL COMBINATIONS OF TWO DRIVES.
8919 :*
8920 :* NOTE: IF ONLY ONE DRIVE IS AVAILABLE THE
8921 :* TEST WILL NOT BE DONE.
8922 :*
8923 *****
8924 TST110: SCOPE
8925 MOV #5,$TIMES ;;DO 5. ITERATIONS
8926 031204 012737 000005 001262 2$: MOV #-1,$TMP1 ;SET LOOP CONTROL FLAG
8927 031212 013705 001354 001224 MOV $DEV0,R5 ;GET DEVICE MAP
8928 031216 005000 CLR R0 ;CLEAR FOR DRIVE #A
8929 031220 005001 CLR R1 ;CLEAR FOR DRIVE #B
8930 031222 012703 000001 MOV #1,R3 ;SET DRIVE POSITION A
8931 031226 012704 000001 MOV #1,R4 ;SET DRIVE POSITION B
8932 031232 012737 031242 001110 MOV #3$,$LPERR ;SET LOCAL LOOP ON ERROR
8933 031240 000555 BR 11$ ;GO SET UP POINTERS
8934 031242 3$:
8935 031242 104416 TSSINIT ;CLEAR SUBSYSTEM
8936 031244 104003 ERROR 3 ;BAD INIT ERROR
8937
8938 031246 010037 001626 MOV R0,DRVNUM ;STORE DRIVE FOR REPORT
8939 031252 010037 001610 MOV R0,L.CS2 ;SETUP DRIVE A TO RECAL
8940 031256 004737 036134 JSR PC,GTYP0
8941 031262 012737 000113 001600 MOV #RECAL,L.CS1
8942
```

```

8943 031270 104417          TLOADRK          ;LOAD RK REGS
8944 031272 104423          TWAT16           ;WAIT FOR INTERRUPT
8945 031274 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8946 031276 005037 001662  CLR INTSET      ;CLEAR INTERRUPT FLAG
8947
8948 031302 104437          TWAT8S          ;WAIT FOR SECOND INTERRUPT
8949 031304 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8950
8951 031306 012737 000105 001600  MOV #CLEAR,L.CS1 ;SET UP TO CLEAR DRIVE
8952 031314 104417          TLOADRK          ;LOAD RK REGS
8953 031316 104423          TWAT16           ;WAIT FOR INTERRUPT
8954 031320 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8955 031322          4$:
8956 031322 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8957 031324 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8958 031326 032737 040000 001540  BIT #DI,T.CS1   ;TEST IF DI STILL SET
8959 031334 001372          BNE 4$         ;YES - LOOP
8960 031336 010137 001626          MOV R1,DRVNUM   ;STORE DRIVE FOR REPORT
8961 031342 010137 001610          MOV R1,L.CS2    ;SETUP DRIVE B TO RECAL
8962 031346 004737 036150          JSR PC,GTYP1
8963 031352 012737 000113 001600  MOV #RECAL,L.CS1
8964
8965 031360 104417          TLOADRK          ;LOAD RK REGS
8966 031362 104423          TWAT16           ;WAIT FOR INTERRUPT
8967 031364 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8968 031366 005037 001662  CLR INTSET      ;CLEAR INTERRUPT FLAG
8969 031372 104437          TWAT8S          ;WAIT FOR SECOND INTERRUPT
8970 031374 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8971 031376 012737 000105 001600  MOV #CLEAR,L.CS1 ;SET UP DRIVE CLEAR
8972 031404 104417          TLOADRK          ;LOAD RK REGS
8973 031406 104423          TWAT16           ;WAIT FOR INTERRUPT
8974 031410 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8975 031412          5$:
8976 031412 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
8977 031414 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
8978 031416 032737 040000 001540  BIT #DI,T.CS1   ;TEST IF DI STILL SET
8979 031424 001372          BNE 5$         ;YES - LOOP
8980
8981 031426 010037 001626          MOV R0,DRVNUM   ;STORE DRIVE FOR REPORT
8982 031432 010037 001610          MOV R0,L.CS2    ;SETUP DRIVE A TO SEEK
8983 031436 004737 036134          JSR PC,GTYP0
8984 031442 012737 000001 001614  MOV #1,L.DCYL   ;TO CYL 1
8985 031450 012737 000117 001600  MOV #SEEK,L.CS1
8986
8987 031456 104417          TLOADRK          ;LOAD RK REGS
8988 031460 104423          TWAT16           ;WAIT FOR INTERRUPT
8989 031462 104002          ERROR 2         ;TO SLOW/NOT COMPLETE ERROR
8990
8991 031464 010137 001626          MOV R1,DRVNUM   ;STORE DRIVE FOR REPORT
8992 031470 010137 001610          MOV R1,L.CS2    ;SETUP DRIVE B TO WRITE DATA
8993 031474 004737 036150          JSR PC,GTYP1
8994 031500 012737 000100 001614  MOV #100,L.DCYL ;AT CYL 100
8995 031506 012737 177400 001602  MOV #-400,L.WC  ;400 WORDS
8996 031514 012737 072414 001604  MOV #OBUFF,L.BA
8997 031522 012737 000123 001600  MOV #WRDATA,L.CS1
8998

```

```

8999 031530 104417          TLOADRK          ;LOAD RK REGS-DO WRITE
9000
9001 031532 104427          TWAT80          ;WAIT FOR INTERRUPT
9002 031534 104002          ERROR 2        ;TO SLOW/NOT COMPLETE ERROR
9003
9004 031536 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
9005 031540 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
9006
9007 031542 010037 001626    MOV R0,DRVNUM   ;STORE DRIVE FOR REPORT
9008 031546 130337 001557    BITB R3,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS DRIVE A
9009 031552 001007          BNE 10$        ;YES-SKIP
9010 031554 012737 056540 001460  MOV #EMDA,EM11N ;'DRIVE ATTENTION NOT SET RESULT OF
9011 031562 012737 050272 061022  MOV #EMSK,DF011A ;SEEK''
9012 031570 104011          ERROR 11
9013
9014 031572 104415          10$: SCOP1      ;LOCAL LOOP TO 3$
9015
9016          :
9017          : THE FOLLOWING CODE CAUSES THE TEST TO BE RUN ON EVERY COMBINATION
9018          : OF DRIVES AVAILABLE. THE FIRST PASS OF THE PROGRAM WILL USE THE
9019          : LOWEST NUMBER DRIVE AS A AND THE NEXT HIGHER NUMBER DRIVE AS
9020          : B. THE SECOND PASS SWAPS DRIVE A & B. THE THIRD PASS USES
9021          : THE LOWEST NUMBER DRIVE AS B AND THE 3RD HIGHEST NUMBER DRIVE
9022          : AS A. THE FORTH PASS SWAPS A & B AGAIN. THIS CONTINUES
9023          : UNTIL ALL DRIVES HAVE BEEN TESTED WITH THE LOWEST NUMBER
9024          : DRIVE.
9025          :
9026          : THE SECOND HIGHEST NUMBER DRIVE IS THEN USED AS A AND THE
9027          : THIRD HIGHEST AS B. THEY ARE SWAPPED ON THE NEXT PASS.
9028          :
9029          : THIS TECHNIQUE IS CONTINUED UNTIL ALL COMBINATIONS ARE
9030          : CHECKED.
9031 031574 005237 001224          11$: INC $TMP1   ;INCREMENT PASS CONTROL
9032 031600 001024          BNE 16$        ;SKIP IF NOT ZERO
9033          ;(IT WILL BE ZERO ON THE 1ST PASS)
9034
9035 031602 030305          12$: BIT R3,R5   ;TEST IF BIT POSITION FOR A AT AVAIL DRIVE
9036 031604 001006          BNE 13$        ;YES-SKIP
9037
9038 031606 005200          22$: INC R0      ;BUMP R0 (DRIVE A)
9039 031610 006303          ASL R3         ;SHIFT DRIVE SELECT BIT ONE POSITION
9040 031612 032703 000400          BIT #BIT8,R3  ;IF BIT 8 IS SET, ALL DRIVES HAVE
9041 031616 001771          BEQ 12$        ;BEEN CHECKED; IF NOT CHECK NEXT POSITION
9042 031620 000464          BR 50$        ;DONE-EXIT
9043
9044 031622 010001          13$: MOV R0,R1   ;SET DRIVE B TO THE SAME AS A
9045 031624 010304          MOV R3,R4
9046 031626 005201          14$: INC R1     ;BUMP R1 (DRIVE B)
9047 031630 006304          ASL R4         ;SHIFT SELECTOR BIT ONE POSITION
9048 031632 030405          BIT R4,R5     ;IS THIS DRIVE AVAIL?
9049 031634 001004          BNE 15$        ;YES-SKIP
9050 031636 032704 000400          BIT #BIT8,R4  ;WERE ALL POSITIONS CHECKED?
9051 031642 001771          BEQ 14$        ;NO-LOOP
9052 031644 000452          BR 50$        ;DONE-EXIT
9053
9054 031646 000137 031242          15$: JMP 3$     ;GO DO THE TEST ON THE DRIVE A & B

```

```

9055                                     ;CONTAINED IN R0 & R1
9056 031652 032737 000001 001224 16$: BIT #BIT0,$TMP1 ;IS PASS FLAGS ODD?
9057 031660 001410 BEQ 17$ ;NO-SKIP
9058
9059 031662 010046 MOV R0,-(SP) ;
9060 031664 010346 MOV R3,-(SP) ;SWAP R0 & R1, R3 & R4
9061 031666 010403 MOV R4,R3 ;TO EXCHANGE DRIVE A & B
9062 031670 010100 MOV R1,R0
9063 031672 012604 MOV (SP)+,R4
9064 031674 012601 MOV (SP)+,R1
9065 031676 000137 031242 JMP 3$ ;REPEAT TEST ON THIS COMBO.
9066
9067 031702 032737 000002 001224 17$: BIT #BIT1,$TMP1 ;TEST IF PASS FLAGS AT HALF MODULE 4?
9068 031710 001410 BEQ 19$ ;NO-SKIP TO BUMP DRIVE B
9069 031712 005200 18$: INC R0 ;BUMP DRIVE A
9070 031714 006303 ASL R3 ;SHIFT DRIVE SELECT BIT
9071 031716 030305 BIT R3,R5 ;AVAILABLE?
9072 031720 001014 BNE 20$ ;YES-SKIP
9073 031722 032703 000400 BIT #BIT8,R3 ;ALL CHECKED?
9074 031726 001771 BEQ 18$ ;NO-SKIP
9075 031730 000412 BR 21$ ;GO TO NEXT PASS
9076
9077 031732 005201 19$: INC R1 ;BUMP DRIVE B
9078 031734 006304 ASL R4 ;SHIFT DRIVE SELECT BIT
9079 031736 030405 BIT R4,R5 ;AVAILABLE?
9080 031740 001004 BNE 20$ ;YES-SKIP
9081 031742 032704 000400 BIT #BIT8,R4 ;ALL CHECKED?
9082 031746 001771 BEQ 19$ ;NO-LOOP
9083 031750 000404 BR 23$ ;YES-SKIP TO NEXT PASS
9084
9085 031752 000137 031242 20$: JMP 3$ ;GO TEST THIS COMBO
9086
9087 031756 010100 21$: MOV R1,R0 ;SET DRIVE 0 TO LOW POSITION THIS PASS
9088 031760 010403 MOV R4,R3 ;SET SELECT BITS TO AGREE
9089 031762 005037 001224 23$: CLR $TMP1 ;CLEAR PASS FLAGS
9090 031766 000137 031606 JMP 22$ ;GO SET UP A & B
9091 031772
9092
9093
9094
9095
9096
9097
9098
9099
9100
9101
9102
9103
9104
9105
9106
9107
9108
9109
9110
  
```

```

;ADD WAITING LOOP FOR APT OPERATION ON JAN-9-78 REV 007
.SBTTL END OF PASS ROUTINE
  
```

```

;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY'
;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO TSTAPT
  
```

```

$EOP:
SCOPE
CLR $TSTNM ;:ZERO THE TEST NUMBER
CLR $TIMES ;:ZERO THE NUMBER OF ITERATIONS
INC $PASS ;:INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;:DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;:LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ;:YES
  
```

```
9111 032024 012737          MOV      (PC)+,@(PC)+      ;;RESTORE COUNTER
9112 032026 000001 $ENDCT: .WORD      1
9113 032030 032020          $EOPCT
9114 032032 104401 032040    TYPE      ,65$           ;;TYPE ASCIZ STRING
9115 032036 000407          BR        64$           ;;GET OVER THE ASCIZ
9116          ;;65$: .ASCIZ <12><15>/END PASS #/
9117 032056 013746 001304    64$:      MOV      $PASS,-(SP)  ;;SAVE $PASS FOR TYPEOUT
9118          ;;TYPE PASS NUMBER
9119          ;;GO TYPE--DECIMAL ASCII WITH SIGN
9120 032062 104405          TYPDS
9121 032064 104401 032072    TYPE      ,67$           ;;TYPE ASCIZ STRING
9122 032070 000421          BR        66$           ;;GET OVER THE ASCIZ
9123          ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
9124 032134 013746 001112    66$:      MOV      $ERTTL,-(SP)  ;;SAVE $ERTTL FOR TYPEOUT
9125          ;;TOTAL NUMBER OF ERRORS
9126          ;;GO TYPE--DECIMAL ASCII WITH SIGN
9127 032140 104405          TYPDS
9128 032142 104401 001273    TYPE      , $CRLF        ;;TYPE CARRIAGE RETURN, LINE FEED
9129 032146 005037 001112    CLR      $ERTTL        ;;CLEAR ERROR TOTAL
9130 032152 013700 000042    $GET42: MOV      @#42,R0    ;;GET MONITOR ADDRESS
9131 032156 001405          BEQ      $DOAGN        ;;BRANCH IF NO MONITOR
9132 032160 000005          RESET                ;;CLEAR THE WORLD
9133 032162 004710    $ENDAD: JSR      PC,(R0)  ;;GO TO MONITOR
9134 032164 000240          NOP
9135 032166 000240          NOP
9136 032170 000240          NOP
9137          ;;DOAGN:
9138 032172 000137          JMP      @(PC)+        ;;RETURN
9139 032174 032202    $RTNAD: .WORD      TSTAPT
9140 032176 377 377 000 $ENULL: .BYTE      -1,-1,0  ;;NULL CHARACTER STRING
9141          .EVEN
9142 032202 122737 000001 001316 TSTAPT: CMPB     #APTENV,$ENV  ;;RUNNING UNDER APT MODE ?
9143 032210 001007          BNE      2$           ;;BRANCH IF NOT
9144 032212 023727 001304 000002    CMP      $PASS,#2     ;;TWO PASS DONE ?
9145 032220 103403          BLO      2$           ;;BRANCH IF NOT
9146 032222 005237 001102    1$:      INC      $TSTNM   ;;INCREMENT TEST NUMBER
9147 032226 000775          BR        1$           ;;WAITING LOOP FOR APT DUMP THE NEXT PROG
9148 032230 000137 003150    2$:      JMP      NEWPAS   ;;REPEAT THE SAME TEST IF NOT UNDER APT
9149          .SBTTL ROUTINE TO SIZE MEMORY
9150
9151          ;;*****
9152          ;;CALL:
9153          ;;* JSR      PC,$SIZE
9154          ;;* RETURN
9155          ;;*$LSTAD WILL CONTAIN:
9156          ;;* WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
9157          ;;* WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
9158          ;;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
9159          ;;*$KT11 IS THE MEMORY MANAGEMENT KEY
9160          ;;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
9161          ;;* MUST BE SETUP BEFORE THE CALL
9162          ;;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
9163          ;;* DETERMINED BY ROUTINE
9164
9165 032234 010046    $SIZE: MOV      R0,-(SP)  ;;SAVE R0 ON THE STACK
9166 032236 010146          MOV      R1,-(SP)  ;;SAVE R1 ON THE STACK
```


9167	032240	010246				MOV	R2,-(SP)	::SAVE R2 ON THE STACK
9168	032242	010346				MOV	R3,-(SP)	::SAVE R3 ON THE STACK
9169	032244	013746	000114			MOV	@#114,-(SP)	::SAVE MEMORY ERROR VECTOR PS & PC
9170	032250	013746	000116			MOV	@#116,-(SP)	
9171	032254	012737	000116	000114		MOV	#116,@#114	::IGNORE PARITY ERRORS WHILE SIZING
9172	032262	012737	000002	000116		MOV	#RTI,@#116	
9173	032270	013746	000004			MOV	@#ERRVEC,-(SP)	::SAVE PRESENT ERROR VECTOR PS & PC
9174	032274	013746	000006			MOV	@#ERRVEC+2,-(SP)	
9175	032300	010600				MOV	SP,R0	::SAVE THE STACK POINTER
9176							::SET THE ERRVEC PS TO THE PRESENT PS	
9177	032302	104400				TRAP		::PUSH OLD PSW AND PC ON STACK
9178	032304	012637	000006			MOV	(SP)+,@#ERRVEC+2	::SAVE THE PSW IN @#ERRVEC+2
9179	032310	012701	003776			MOV	#3776,R1	::SETUP ADDRESS
9180	032314	105727				TSTB	(PC)+	::USE MEMORY MANAGEMENT?
9181	032316	000200			\$KT11:	.WORD	200	::SET TO USE MEMORY MANAGEMENT
9182	032320	100062				BPL	\$SCORE	::BR IF NO
9183	032322	012737	032460	000004		MOV	#\$KTNEX,@#ERRVEC	::SET FOR TIMEOUT
9184	032330	005737	177572			TST	@#SR0	::KT11 ARE YOU THERE?
9185	032334	052737	100000	032316		BIS	#100000,\$KT11	::YES--SET KT11 KEY
9186	032342	005046				CLR	-(SP)	::INITIALIZE FOR 'PAR' LOADING
9187	032344	012702	172340			MOV	#KIPAR0,R2	::ADDRESS OF FIRST 'PAR'
9188	032350	012703	000010			MOV	#^D8,R3	::LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
9189	032354	012762	077406	177740	1\$:	MOV	#77406,-40(R2)	::PDR = 4K, UP, READ/WRITE
9190	032362	011622				MOV	(SP),(R2)+	::LOAD 'PAR'
9191	032364	062716	000200			ADD	#200,(SP)	::UPDATE FOR NEXT 'PAR'
9192	032370	077307				SOB	R3,1\$::LOOP UNTIL ALL EIGHT ARE LOADED
9193	032372	012742	177600			MOV	#177600,-(R2)	::SETUP KIPAR7 FOR I/O
9194	032376	005042				CLR	-(R2)	::SETUP KIPAR6 FOR TESTING
9195	032400	012737	032416	000004		MOV	#2\$,@#ERRVEC	::CATCH TIMEOUT IF NO SR3
9196	032406	012737	000020	172516		MOV	#20,@#SR3	::ENABLE 22 BIT MODE
9197	032414	000401				BR	3\$::THIS PDP-11 HAS A SR3 REGISTER
9198	032416	022626			2\$:	CMP	(SP)+,(SP)+	::CLEAN OFF THE STACK--NO SR3
9199	032420	005237	177572		3\$:	INC	@#SR0	::TURN ON MEMORY MANAGEMENT
9200	032424	012737	032450	000004		MOV	#\$KTOUT,@#ERRVEC	::SET FOR TIME OUT
9201	032432	005737	143776		4\$:	TST	@#143776	::TRAP ON NON-EX-MEM
9202	032436	062712	000040			ADD	#40,(R2)	::MAKE A 1K STEP
9203	032442	023712	172356			CMP	@#KIPAR7,(R2)	::LAST ONE?
9204	032446	101371				BHI	4\$::NO--TRY IT
9205	032450	011202			\$KTOUT:	MOV	(R2),R2	::GET LAST BANK+1
9206	032452	005037	177572			CLR	@#SR0	::TURN OFF MEMORY MANAGEMENT
9207	032456	000421				BR	\$SIZEX	
9208	032460	042737	100000	032316	\$KTNEX:	BIC	#100000,\$KT11	::KT11 NON-EXISTENT
9209	032466	012737	032516	000004	\$SCORE:	MOV	#\$CROUT,@#ERRVEC	::SET FOR TIMEOUT
9210	032474	005002				CLR	R2	::SET UP BANK
9211	032476	062701	004000		1\$:	ADD	#4000,R1	::INCREMENT BY 1K
9212	032502	062702	000040			ADD	#40,R2	::1K STEP
9213	032506	005711				TST	(R1)	::TRAP ON TIME OUT
9214	032510	022701	177776			CMP	#177776,R1	::LAST ONE
9215	032514	001370				BNE	1\$::NO--TRY AGAIN
9216	032516	162701	004000		\$CROUT:	SUB	#4000,R1	
9217	032522	162702	000040		\$SIZEX:	SUB	#40,R2	::DROP BACK
9218	032526	010006				MOV	R0,SP	::RESTORE THE STACK
9219	032530	012637	000006			MOV	(SP)+,@#ERRVEC+2	::RESTORE ERROR VECTOR
9220	032534	012637	000004			MOV	(SP)+,@#ERRVEC	
9221	032540	012637	000116			MOV	(SP)+,@#116	::RESTORE MEMORY ERROR VECTOR
9222	032544	012637	000114			MOV	(SP)+,@#114	

```
9223 032550 010137 032572      MOV      R1,$LSTAD      ;;LAST ADDRESS
9224 032554 010237 032574      MOV      R2,$LSTBK      ;;LAST BANK
9225 032560 012603              MOV      (SP)+,R3      ;;RESTORE R3
9226 032562 012602              MOV      (SP)+,R2      ;;RESTORE R2
9227 032564 012601              MOV      (SP)+,R1      ;;RESTORE R1
9228 032566 012600              MOV      (SP)+,R0      ;;RESTORE R0
9229 032570 000207              RTS      PC
9230 032572 000000      $LSTAD: .WORD 0      ;;CONTAINS THE LAST ADDRESS
9231 032574 000000      $LSTBK: .WORD 0      ;;CONTAINS THE LAST BANK
9232                                .SBTTL SCOPE HANDLER ROUTINE
9233
9234                                ;*****
9235                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
9236                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
9237                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
9238                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9239                                ;*SW14=1      LOOP ON TEST
9240                                ;*SW11=1      INHIBIT ITERATIONS
9241                                ;*SW09=1      LOOP ON ERROR
9242                                ;*SW08=1      LOOP ON TEST IN SWR<7:0>
9243                                ;*CALL
9244                                ;*      SCOPE      ;;SCOPE=IOT
9245
9246 032576      $SCOPE:
9247 032576 104407      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
9248                                ;;GO TO ERROR ROUTINE IF RETURN PC LESS THAN 1002
9249                                ;;OTHERWISE CONTINUE
9250 032600 021627 001002      CMP      (SP),#1002      ;;UNEXPECTED TRAP OR INTERRUPT
9251 032604 101002      BHI      1$      ;;ARE TRAPPED HERE VIA IOT
9252 032606 000137 033614      JMP      $ERROR      ;;GO PROCESS UNEXPECTED TRAP
9253 032612 032777 040000 146320 1$: BIT      #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
9254 032620 001131      BNE      $OVER      ;;YES IF SW14=1
9255                                ;*****START OF CODE FOR THE XOR TESTER*****
9256 032622 000416      $XTSTR: BR      6$      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
9257                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
9258 032624 013746 000004      MOV      @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
9259 032630 012737 032650 000004      MOV      #5$,@#ERRVEC      ;;SET FOR TIMEOUT
9260 032636 005737 177060      TST      @#177060      ;;TIME OUT ON XOR?
9261 032642 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
9262 032646 000500      BR      $SVLAD      ;;GO TO THE NEXT TEST
9263 032650 022626 5$: CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
9264 032652 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
9265 032656 000440      BR      7$      ;;LOOP ON THE PRESENT TEST
9266 032660 6$: ;*****END OF CODE FOR THE XOR TESTER*****
9267 032660 032777 000400 146252      BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
9268 032666 001421      BEQ      2$      ;;BR IF NO
9269 032670 005046      CLR      -(SP)      ;;CLEAR A TEMP. LOCATION
9270 032672 117716 146242      MOV      @SWR,(SP)      ;;PICKUP THE DESIRED TEST NUMBER
9271 032676 001414      BEQ      8$      ;;BRANCH IF BAD TEST NUMBER IN SWR
9272 032700 022716 000110      CMP      #110,(SP)      ;;CHECK THE NUMBER IN THE SWR
9273 032704 002411      BLT      8$      ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
9274 032706 011637 001102      MOV      (SP),$TSTNM      ;;UPDATE THE TEST NUMBER
9275 032712 005316      DEC      (SP)      ;;BACKUP BY ONE
9276 032714 006316      ASL      (SP)      ;;SCALE THE TEST NUMBER AS AN INDEX
9277 032716 062716 033122      ADD      #$$SW08TBL,(SP)      ;;FORM THE ADDRESS OF TEST POINTER
9278 032722 013637 001106      MOV      @((SP)+,$LPADR)      ;;SET LOOP ADDRESS TO DESIRED TEST
```

9279	032726	000466				BR	\$OVER	::GO LOOP ON THE TEST
9280	032730	005726			8\$:	TST	(SP)+	::CLEAN THE BAD TEST NUMBER OFF OF THE STACK
9281	032732	105737	001103		2\$:	TSTB	\$ERFLG	::HAS AN ERROR OCCURRED?
9282	032736	001421				BEQ	3\$::BR IF NO
9283	032740	123737	001115	001103		CMPB	\$ERMAX,\$ERFLG	::MAX. ERRORS FOR THIS TEST OCCURRED?
9284	032746	101015				BHI	3\$::BR IF NO
9285	032750	032777	001000	146162		BIT	#BIT09,@SWR	::LOOP ON ERROR?
9286	032756	001404				BEQ	4\$::BR IF NO
9287	032760	013737	001110	001106	7\$:	MOV	\$LPERR,\$LPADR	::SET LOOP ADDRESS TO LAST SCOPE
9288	032766	000446				BR	\$OVER	
9289	032770	105037	001103		4\$:	CLRB	\$ERFLG	::ZERO THE ERROR FLAG
9290	032774	005037	001262			CLR	\$TIMES	::CLEAR THE NUMBER OF ITERATIONS TO MAKE
9291	033000	000415				BR	1\$::ESCAPE TO THE NEXT TEST
9292	033002	032777	004000	146130	3\$:	BIT	#BIT11,@SWR	::INHIBIT ITERATIONS?
9293	033010	001011				BNE	1\$::BR IF YES
9294	033012	005737	001304			TST	\$PASS	::IF FIRST PASS OF PROGRAM
9295	033016	001406				BEQ	1\$::INHIBIT ITERATIONS
9296	033020	005237	001104			INC	\$ICNT	::INCREMENT ITERATION COUNT
9297	033024	023737	001262	001104		CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE
9298	033032	002024				BGE	\$OVER	::BR IF MORE ITERATION REQUIRED
9299	033034	012737	000001	001104	1\$:	MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
9300	033042	013737	033120	001262		MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
9301	033050	105237	001102		\$SVLAD:	INCB	\$TSTNM	::COUNT TEST NUMBERS
9302	033054	113737	001102	001302		MOVB	\$TSTNM,\$TESTN	::SET TEST NUMBER IN APT MAILBOX
9303	033062	011637	001106			MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
9304	033066	011637	001110			MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
9305	033072	005037	001264			CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
9306	033076	112737	000001	001115		MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9307	033104	013777	001102	146030	\$OVER:	MOV	\$TSTNM,@DISPLAY	::DISPLAY TEST NUMBER
9308	033112	013716	001106			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
9309	033116	000002				RTI		::FIXES PS
9310	033120	003720			\$MXCNT:	2000.		::MAX. NUMBER OF ITERATIONS
9311	033122				\$SW08TBL:			
9312	033122	003164			.WORD	TST1+2		::STARTING ADDRESS OF TEST 1
9313	033124	003302			.WORD	TST2+2		::STARTING ADDRESS OF TEST 2
9314	033126	003370			.WORD	TST3+2		::STARTING ADDRESS OF TEST 3
9315	033130	004620			.WORD	TST4+2		::STARTING ADDRESS OF TEST 4
9316	033132	004726			.WORD	TST5+2		::STARTING ADDRESS OF TEST 5
9317	033134	005104			.WORD	TST6+2		::STARTING ADDRESS OF TEST 6
9318	033136	005232			.WORD	TST7+2		::STARTING ADDRESS OF TEST 7
9319	033140	005336			.WORD	TST10+2		::STARTING ADDRESS OF TEST 10
9320	033142	006262			.WORD	TST11+2		::STARTING ADDRESS OF TEST 11
9321	033144	006400			.WORD	TST12+2		::STARTING ADDRESS OF TEST 12
9322	033146	006516			.WORD	TST13+2		::STARTING ADDRESS OF TEST 13
9323	033150	006760			.WORD	TST14+2		::STARTING ADDRESS OF TEST 14
9324	033152	007112			.WORD	TST15+2		::STARTING ADDRESS OF TEST 15
9325	033154	007322			.WORD	TST16+2		::STARTING ADDRESS OF TEST 16
9326	033156	007512			.WORD	TST17+2		::STARTING ADDRESS OF TEST 17
9327	033160	007646			.WORD	TST20+2		::STARTING ADDRESS OF TEST 20
9328	033162	010514			.WORD	TST21+2		::STARTING ADDRESS OF TEST 21
9329	033164	011112			.WORD	TST22+2		::STARTING ADDRESS OF TEST 22
9330	033166	011324			.WORD	TST23+2		::STARTING ADDRESS OF TEST 23
9331	033170	011542			.WORD	TST24+2		::STARTING ADDRESS OF TEST 24
9332	033172	011666			.WORD	TST25+2		::STARTING ADDRESS OF TEST 25
9333	033174	012020			.WORD	TST26+2		::STARTING ADDRESS OF TEST 26
9334	033176	012176			.WORD	TST27+2		::STARTING ADDRESS OF TEST 27

```
9335 033200 012330 .WORD TST30+2 ;; STARTING ADDRESS OF TEST 30
9336 033202 012624 .WORD TST31+2 ;; STARTING ADDRESS OF TEST 31
9337 033204 013020 .WORD TST32+2 ;; STARTING ADDRESS OF TEST 32
9338 033206 013254 .WORD TST33+2 ;; STARTING ADDRESS OF TEST 33
9339 033210 013442 .WORD TST34+2 ;; STARTING ADDRESS OF TEST 34
9340 033212 013674 .WORD TST35+2 ;; STARTING ADDRESS OF TEST 35
9341 033214 014062 .WORD TST36+2 ;; STARTING ADDRESS OF TEST 36
9342 033216 014304 .WORD TST37+2 ;; STARTING ADDRESS OF TEST 37
9343 033220 014472 .WORD TST40+2 ;; STARTING ADDRESS OF TEST 40
9344 033222 014664 .WORD TST41+2 ;; STARTING ADDRESS OF TEST 41
9345 033224 015060 .WORD TST42+2 ;; STARTING ADDRESS OF TEST 42
9346 033226 015264 .WORD TST43+2 ;; STARTING ADDRESS OF TEST 43
9347 033230 015470 .WORD TST44+2 ;; STARTING ADDRESS OF TEST 44
9348 033232 016330 .WORD TST45+2 ;; STARTING ADDRESS OF TEST 45
9349 033234 016534 .WORD TST46+2 ;; STARTING ADDRESS OF TEST 46
9350 033236 016740 .WORD TST47+2 ;; STARTING ADDRESS OF TEST 47
9351 033240 017156 .WORD TST50+2 ;; STARTING ADDRESS OF TEST 50
9352 033242 017362 .WORD TST51+2 ;; STARTING ADDRESS OF TEST 51
9353 033244 017566 .WORD TST52+2 ;; STARTING ADDRESS OF TEST 52
9354 033246 017712 .WORD TST53+2 ;; STARTING ADDRESS OF TEST 53
9355 033250 020036 .WORD TST54+2 ;; STARTING ADDRESS OF TEST 54
9356 033252 020162 .WORD TST55+2 ;; STARTING ADDRESS OF TEST 55
9357 033254 020306 .WORD TST56+2 ;; STARTING ADDRESS OF TEST 56
9358 033256 020432 .WORD TST57+2 ;; STARTING ADDRESS OF TEST 57
9359 033260 020504 .WORD TST60+2 ;; STARTING ADDRESS OF TEST 60
9360 033262 020564 .WORD TST61+2 ;; STARTING ADDRESS OF TEST 61
9361 033264 021050 .WORD TST62+2 ;; STARTING ADDRESS OF TEST 62
9362 033266 021232 .WORD TST63+2 ;; STARTING ADDRESS OF TEST 63
9363 033270 021420 .WORD TST64+2 ;; STARTING ADDRESS OF TEST 64
9364 033272 022124 .WORD TST65+2 ;; STARTING ADDRESS OF TEST 65
9365 033274 022342 .WORD TST66+2 ;; STARTING ADDRESS OF TEST 66
9366 033276 022446 .WORD TST67+2 ;; STARTING ADDRESS OF TEST 67
9367 033300 023070 .WORD TST70+2 ;; STARTING ADDRESS OF TEST 70
9368 033302 023512 .WORD TST71+2 ;; STARTING ADDRESS OF TEST 71
9369 033304 023622 .WORD TST72+2 ;; STARTING ADDRESS OF TEST 72
9370 033306 024470 .WORD TST73+2 ;; STARTING ADDRESS OF TEST 73
9371 033310 025162 .WORD TST74+2 ;; STARTING ADDRESS OF TEST 74
9372 033312 025404 .WORD TST75+2 ;; STARTING ADDRESS OF TEST 75
9373 033314 025662 .WORD TST76+2 ;; STARTING ADDRESS OF TEST 76
9374 033316 025742 .WORD TST77+2 ;; STARTING ADDRESS OF TEST 77
9375 033320 026134 .WORD TST100+2 ;; STARTING ADDRESS OF TEST 100
9376 033322 026360 .WORD TST101+2 ;; STARTING ADDRESS OF TEST 101
9377 033324 026506 .WORD TST102+2 ;; STARTING ADDRESS OF TEST 102
9378 033326 027030 .WORD TST103+2 ;; STARTING ADDRESS OF TEST 103
9379 033330 027562 .WORD TST104+2 ;; STARTING ADDRESS OF TEST 104
9380 033332 030024 .WORD TST105+2 ;; STARTING ADDRESS OF TEST 105
9381 033334 030250 .WORD TST106+2 ;; STARTING ADDRESS OF TEST 106
9382 033336 030470 .WORD TST107+2 ;; STARTING ADDRESS OF TEST 107
9383 033340 031176 .WORD TST110+2 ;; STARTING ADDRESS OF TEST 110
9384 033342 031774 .WORD $EOP+2 ; ADDRESS OF END OF PASS
9385 033344 044172 .WORD ABTFAIL+2 ; ADDRESS OF ABORT FAILURE HANDLER
9386
9387 .SBTTL APT COMMUNICATIONS ROUTINE
9388
9389 ;*****
9390 033346 112737 000001 033612 $ATY1: MOVB #1,$FFLG ;; TO REPORT FATAL ERROR
```

```

9391 033354 112737 000001 033610 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
9392 033362 000403 BR $ATYC
9393 033364 112737 000001 033612 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
9394 033372 $ATYC:
9395 033372 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
9396 033374 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
9397 033376 105737 033610 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
9398 033402 001450 BEQ 5$ ;;IF NOT: BR
9399 033404 122737 000001 001316 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
9400 033412 001031 BNE 3$ ;;IF NOT: BR
9401 033414 132737 000100 001317 BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
9402 033422 001425 BEQ 3$ ;;IF NOT: BR
9403 033424 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
9404 033430 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
9405 033436 005737 001276 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
9406 033442 001375 BNE 1$ ;;IF NOT: WAIT
9407 033444 010037 001312 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
9408 033450 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
9409 033452 001376 BNE 2$
9410 033454 163700 001312 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
9411 033460 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
9412 033462 010037 001314 MOV R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
9413 033466 012737 000004 001276 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
9414 033474 000413 BR 5$
9415 033476 017637 000004 033522 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
9416 033504 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
9417 033512 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
9418 033516 004737 044202 JSR PC,$TYPE ;;CALL TYPE MACRO
9419 033522 000000 4$: .WORD 0
9420 033524 5$:
9421 033524 105737 033612 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
9422 033530 001416 BEQ 12$ ;;IF NOT: BR
9423 033532 005737 001316 TST $ENV ;;RUNNING UNDER APT?
9424 033536 001413 BEQ 12$ ;;IF NOT: BR
9425 033540 005737 001276 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
9426 033544 001375 BNE 11$ ;;IF NOT: WAIT
9427 033546 017637 000004 001300 MOV @4(SP),$FATAL ;;GET ERROR #
9428 033554 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
9429 033562 005237 001276 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
9430 033566 105037 033612 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
9431 033572 105037 033611 CLRB $LFLG ;;CLEAR LOG FLAG
9432 033576 105037 033610 CLRB $MFLG ;;CLEAR MESSAGE FLAG
9433 033602 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
9434 033604 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
9435 033606 000207 RTS PC ;;RETURN
9436 033610 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
9437 033611 000 $LFLG: .BYTE 0 ;;LOG FLAG
9438 033612 000 $FFLG: .BYTE 0 ;;FATAL FLAG
9439 033614 .EVEN
9440 000200 APTSIZE=200
9441 000001 APTENV=001
9442 000100 APTPOOL=100
9443 000040 APTCSUP=040
9444 .SBTTL ERROR HANDLER ROUTINE
9445
9446

```

```
9447      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,  
9448      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL  
9449      ;*AND GO TO TYPERR ON ERROR  
9450      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
9451      ;*SW15=1      HALT ON ERROR  
9452      ;*SW13=1      INHIBIT ERROR TYPEOUTS  
9453      ;*SW10=1      BELL ON ERROR  
9454      ;*SW09=1      LOOP ON ERROR  
9455      ;*CALL  
9456      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER  
9457  
9458      033614      $ERROR:  
9459      033614      104407      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR  
9460      033616      105237      001103      7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG  
9461      033622      001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO  
9462      033624      013777      001102      145310      MOV      $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER AND ERROR FLAG  
9463      033632      032777      002000      145300      BIT      #BIT10,@SWR      ;;BELL ON ERROR?  
9464      033640      001402      BEQ      1$      ;;NO - SKIP  
9465      033642      104401      001266      TYPE      $BELL      ;;RING BELL  
9466      033646      005237      001112      1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS  
9467      033652      011637      001116      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION  
9468      033656      162737      000002      001116      SUB      #2,$ERRPC  
9469      033664      117737      145226      001114      MOV      @ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE  
9470      033672      032777      020000      145240      BIT      #BIT13,@SWR      ;;SKIP TYPEOUT IF SET  
9471      033700      001055      BNE      20$      ;;SKIP TYPEOUTS  
9472      033702      021627      001002      CMP      (SP),#1002      ;;IF RETURN PC LESS THAN 1002  
9473      033706      101046      BHI      12$      ;;ERROR IS ILLEGAL TRAP  
9474      ;;PROCESS UNEXPECTED TRAP OR INTERRUPT  
9475      033710      016637      000004      001116      MOV      4(SP),$ERRPC      ;;GET PC AT TIME OF FALSE TRAP  
9476      033716      162737      000002      001116      SUB      #2,$ERRPC      ;;ADJUST PC  
9477      033724      104401      033770      TYPE      ,10$      ;;TYPE HEADER  
9478      033730      013746      001116      MOV      $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT  
9479      033734      104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)  
9480      033736      104401      033776      TYPE      ,11$  
9481      033742      162716      000004      SUB      #4,(SP)      ;;GET FALSE TRAP VECTOR ADDR  
9482      033746      011637      001116      MOV      (SP),$ERRPC  
9483      033752      013746      001116      MOV      $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT  
9484      033756      104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)  
9485      033760      104401      001273      TYPE      , $CRLF  
9486      033764      022626      CMP      (SP)+,(SP)+      ;;POP FALSE TRAP VECTOR PC&ADDR  
9487      033766      000422      BR      20$  
9488      033770      050200      036503      000040      10$:      .ASCIZ      <200>'PC= '  
9489      033776      020040      047125      054105      11$:      .ASCIZ      ' UNEXPECTED TRAP TO '  
9490      034004      042520      052103      042105  
9491      034012      052040      040522      020120  
9492      034020      047524      000040  
9493      .EVEN  
9494      034024      12$:  
9495      034024      004737      034136      JSR      PC,TYPERR      ;;GO TO USER ERROR ROUTINE  
9496      034030      104401      001273      TYPE      , $CRLF  
9497      034034      20$:  
9498      034034      122737      000001      001316      CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE  
9499      034042      001007      BNE      2$      ;;NO,SKIP APT ERROR REPORT  
9500      034044      113737      001114      034056      MOV      $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER  
9501      034052      004737      033364      JSR      PC,$ATY4      ;;REPORT FATAL ERROR TO APT  
9502      034056      000      21$:      .BYTE      0
```

```
9503 034057 000  
9504 034060 000777  
9505 034062 005777 145052  
9506 034066 100002  
9507 034070 000000  
9508 034072 104407  
9509 034074 032777 001000 145036  
9510 034102 001402  
9511 034104 013716 001110  
9512 034110 005737 001264  
9513 034114 001402  
9514 034116 013716 001264  
9515 034122  
9516 034122 022737 032162 000042  
9517 034130 001001  
9518 034132 000000  
9519 034134  
9520 034134 000002  
9521  
9522  
9523  
9524  
9525  
9526  
9527  
9528  
9529  
9530  
9531 034136 104413  
9532 034140 113737 001102 001302  
9533 034146 042737 177400 001302  
9534 034154 113700 001114  
9535 034160 042700 177400  
9536 034164 005300  
9537 034166 006300  
9538 034170 006300  
9539 034172 006300  
9540 034174 062700 001360  
9541 034200 012037 034214  
9542 034204 001404  
9543 034206 104401 001273  
9544 034212 104401  
9545 034214 000000  
9546 034216 012037 034232  
9547 034222 001404  
9548 034224 104401 001273  
9549 034230 104401  
9550 034232 000000  
9551 034234 012001  
9552 034236 001445  
9553 034240 005004  
9554 034242 012000  
9555 034244 012002  
9556 034246 104401 001273  
9557 034252 112003  
9558 034254 105720
```

```
                .BYTE 0  
22$: BR 22$ ;:APT ERROR LOOP  
2$: TST @SWR ;:HALT ON ERROR  
    BPL 3$ ;:SKIP IF CONTINUE  
    HALT ;:HALT ON ERROR!  
    CKSWR ;:TEST FOR CHANGE IN SOFT-SWR  
3$: BIT #BIT09,@SWR ;:LOOP ON ERROR SWITCH SET?  
    BEQ 4$ ;:BR IF NO  
    MOV $LPERR,(SP) ;:FUDGE RETURN FOR LOOPING  
4$: TST $ESCAPE ;:CHECK FOR AN ESCAPE ADDRESS  
    BEQ 5$ ;:BR IF NONE  
    MOV $ESCAPE,(SP) ;:FUDGE RETURN ADDRESS FOR ESCAPE  
5$:  
    CMP #SENDAD,@#42 ;:ACT-11 AUTO-ACCEPT?  
    BNE 6$ ;:BRANCH IF NO  
    HALT ;:YES  
6$:  
    RTI ;:RETURN  
;:*****  
;:SBTTL TYPE ERROR ROUTINE  
;:*ENTRY JSR PC,TYPERR  
;:*RETURN RTS PC  
;:*  
;:*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH  
;:*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)  
;:*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING  
;:*THE ERROR.  
;:*****  
TYPERR: SAVREG  
    MOVB $STNM,$TESTN ;:GET TEST NUMBER FOR REPORT  
    BIC #177400,$TESTN ;:CLEAR UNUSED BITS  
    MOVB $ITEMB,R0 ;:ENTER ERROR NUMBER  
    BIC #177400,R0 ;:CLEAR UNUSED BITS  
    DEC R0 ;:FORM INDEX FOR ERROR TABLE  
    ASL R0  
    ASL R0  
    ASL R0  
1$: ADD #ERRTB,R0 ;:FORM ADDRESS OF ERROR ENTRY  
    MOV (R0)+,2$ ;:GET EM POINTER  
    BEQ 3$ ;:BRANCH IF THERE ISN'T ONE  
    TYPE ,%CRLF ;:TYPE CARRIAGE RETURN LINE FEED  
    TYPE ;:TYPE ERROR MESSAGE (EM)  
2$: .WORD 0 ;:EM POINTER GOES HERE  
3$: MOV (R0)+,4$ ;:GET DH POINTER  
    BEQ 5$ ;:BRANCH IF THERE ISN'T ONE  
    TYPE ,%CRLF ;:TYPE CR-LF  
    TYPE ;:TYPE DATA HEADER  
4$: .WORD 0 ;:DH POINTER GOES HERE  
5$: MOV (R0)+,R1 ;:GET DT POINTER  
    BEQ 20$ ;:BRANCH IF THERE ARE NONE  
    CLR R4 ;:SET INDENT SWITCH  
    MOV (R0)+,R0 ;:GET DF POINTER  
    MOV (R0)+,R2 ;:STORE NUMBER OF DH'S  
    TYPE ,%CRLF  
10$: MOVB (R0)+,R3 ;:GET & STORE NUMBER OF DATA WORDS  
    TSTB (R0)+ ;:BUMP PAST FORMAT WORD
```

```

9559 034256 005703          TST      R3          ;TEST IF ANY DATA FOR THIS HEADER
9560 034260 001416          BEQ      14$         ;NO - SKIP DATA PRINT
9561 034262 005704          TST      R4          ;CHECK FOR INDENT
9562 034264 001004          BNE      12$         ;YES, GO INDENT
9563 034266 013146          11$:    MOV      @ (R1)+, -(SP) ;PUT FIRST DATA WORD ON STACK
9564 034270 104402          TYPOC                     ;TYPE IT
9565 034272 005303          DEC      R3          ;MORE DATA WORDS
9566 034274 001403          BEQ      13$         ;NO-BRANCH
9567 034276 104401 050763    TYPE     ,SPACE2      ;TYPE SEPARATORS
9568 034302 000771          BR       11$         ;LOOP
9569 034304 104401 001273    13$:    TYPE     ,$CRLF    ;TYPE <CR><LF>
9570 034310 005710          TST      (R0)        ;CHECK IF NEXT HEADER AVAILABLE
9571 034312 001401          BEQ      14$         ;NO, DO NOT CHANGE INDENT
9572 034314 005104          COM      R4          ;CHANGE INDENT
9573 034316 005302          14$:    DEC      R2          ;MORE DH'S?
9574 034320 003414          BLE      20$         ;NO-BRANCH
9575 034322 012037 034342    15$:    MOV      (R0)+, 18$  ;GET NEXT DH POINTER
9576 034326 001751          BEQ      10$         ;IF 0 GET DATA
9577 034330 005704          TST      R4          ;INDENT?
9578 034332 001402          BEQ      17$         ;NO, BRANCH
9579 034334 104401 050763    TYPE     ,SPACE2      ;TYPE INDENT
9580 034340 104401          17$:    TYPE     DH          ;TYPE DH
9581 034342 000000          18$:    .WORD    0          ;DH POINTER GOES HERE
9582 034344 104401 001273    TYPE     , $CRLF      ;
9583 034350 000740          BR       10$         ;GET DATA
9584 034352 104414          20$:    RESREG                     ;
9585 034354 005237 001632    INC      ERRCNT      ;INCREMENT THE ERROR COUNT
9586 034360 032777 010000 144552    BIT      #SW12, @SWR  ;CHECK IF SWITCH 12 SET
9587 034366 001421          BEQ      25$         ;NO, RETURN
9588 034370 022737 000024 001632    CMP      #20, ERRCNT ;CHECK IF ERROR THRESHOLD EXCEEDED
9589 034376 103015          BHIS    25$         ;NO, RETURN
9590 034400 104401 053227    TYPE     ,ABORT      ;TYPE 'PROGRAM ABORTED BECAUSE
9591                                ;ERROR THRESHOLD EXCEEDED'
9592 034404 005737 000042    TST      42          ;CHECK IF CHAIN MODE
9593 034410 001407          BEQ      22$         ;NO, HALT PROCESSOR
9594 034412 012706 001100    MOV      #STACK, SP  ;INITIALIZE STACK
9595 034416 012737 000001 032020    MOV      #1, $EOPCT  ;FORCE END OF PROGRAM
9596 034424 000137 031772    JMP      $EOP
9597 034430 000000          22$:    HALT
9598 034432 000207          25$:    RTS      PC
9599                                ;SBTTL NON-EXISTANT MEMORY AND INTERRUPT CHECK HANDLER
9600                                ;* THIS ROUTINE SETS THE INTERRUPT FLAG AND DOES AN RTI.
9601                                ;* THIS IS THE INDICATION TO THE ROUTINE CHECKING
9602                                ;* NON-EXISTANT MEMORY THAT AN INTERRUPT DID OCCUR.
9603
9604 034434 005237 001662    NEXINT: INC      INTSET ;BUMP THE INTERRUPT COUNTER
9605 034440 000002          RTI
9606
9607                                ;SBTTL RK611 INTERRUPT HANDLER
9608                                ;* MOST INTERRUPTS FROM THE RK611 ARE HANDLED BY THIS ROUTINE. ACTUAL
9609                                ;* PROCESSING AS A RESULT OF THE INTERRUPT IS LEFT TO THE MAIN
9610                                ;* PROGRAM. THE HANDLER JUST SETS A FLAG TO INDICATE THE
9611                                ;* INTERRUPT OCCURRED.
9612
9613 034442 000240          INTHLR: NOP
9614 034444 005237 001662    INC      INTSET      ;BUMP THE INTERRUPT FLAG

```



```

9615 034450 000002          RTI          ;RETURN.
9616
9617          .SBTTL  MEMORY PARITY ERROR TRAP HANDLER
9618          ;*      MEMORY PARITY TRAPS WILL BE REPORTED BY THIS ROUTINE. THE REPORT
9619          ;*      WILL INCLUDE THE PC AT TIME OF FAILURE AND ABORT THE PROGRAM.
9620
9621 034452 032777 020000 144460 PERHLR: BIT    #BIT13,@SWR    ;TEST IF INHIBIT REPORT
9622 034460 001003          BNE     1$          ;YES - SKIP
9623 034462 104401 053456          TYPE   ,EM3        ;TYPE PARITY ERROR MESSAGE
9624 034466 104402          TYPOC          ;AND PC VALUE
9625 034470 012737 000001 032020 1$:  MOV    #1,$EOPCT    ;FORCE END OF PROGRAM
9626 034476 012706 001100          MOV    #STACK,SP   ;CLEAN OFF STACK
9627 034502 000137 031772          JMP    $EOP
9628
9629          .SBTTL  LINE CLOCK INTERRUPT HANDLER
9630          ;*      THE LINE CLOCK INTERRUPT HANDLER WILL INCREMENT THE LCLKTK
9631          ;*      (LINE CLOCK TICK COUNTER) EACH TIME AN INTERRUPT OCCURS.
9632
9633 034506 005237 001674 145170 LCKHLR: INC    LCLKTK    ;INCREMENT CLOCK TICK COUNTER
9634 034512 042777 000200          BIC    #BIT7,@KWLADD ;CLEAR MONITOR BIT
9635 034520 000002          RTI
9636
9637          ;*****
9638          .SBTTL  OPTION PRESENT TEST AND SETUP
9639          ;*      THIS ROUTINE CHECKS IF THE MEMORY PARITY OPTION AND THE
9640          ;*      LINE CLOCK ARE ON THE SYSTEM. FLAGS ARE SET IF PRESENT; CLEARED
9641          ;*      OTHERWISE, AND THE APPROPRIATE INTERRUPT VECTORS ARE SET UP.
9642 034522 104413          OPTTST: SAVREG
9643 034524 013746 000004          MOV    ERRVEC,-(SP) ;STORE OLD NEM CONTENTS
9644 034530 013746 000006          MOV    ERRVEC+2,-(SP)
9645 034534 012737 034754 000004          MOV    #20$,ERRVEC ;DET INTERRUPT FOR NEM
9646 034542 012737 000340 000006          MOV    #PR7,ERRVEC+2 ;SET PRIORITY
9647 034550 005037 001670          CLR    MEMPAR      ;CLEAR PARITY WORD FLAGS
9648 034554 042737 000100 001664          BIC    #PARPRE,OPTFLG ;CLEAR PARITY PRESENT FLAG
9649 034562 005737 170200          TST    170200      ;TEST IF EITHER 11/70 OR 11/44
9650 034566 000240          NOP
9651 034570 000240          NOP
9652 034572 005737 177770          TST    177770      ;NOW SEE IF IT'S 11/70
9653 034576 000240          NOP
9654 034600 000240          NOP
9655 034602 012737 177750 001672          MOV    #177750,CSRPTR ;SET POINTER FOR 11/70
9656 034610 052737 002000 001664          BIS    #CP1170,OPTFLG ;SET 11/70 FLAG
9657 034616 052737 000100 001664          BIS    #PARPRE,OPTFLG ;SET PARITY PRESENT FLAG
9658 034624 000507          BR     35$        ;GO TO VECTOR SETUP
9659
9660 034626 013703 001714          3$:  MOV    MEMBAS,R3   ;SET UP POINTER TO FIRST PARITY CSR
9661 034632 012704 000001          MOV    #1,R4       ;INIT MASK
9662 034636 012713 000001          6$:  MOV    #1,(R3)     ;SET PARITY DETECT IN THAT CSR
9663 034642 005713          TST    (R3)
9664 034644 052713 000002          BIS    #BIT1,(R3)  ;IS THIS ECC MEMORY?
9665 034650 032713 000002          BIT    #BIT1,(R3)
9666 034654 001061          BNE    47$         ;YES - EXIT
9667 034656 050437 001670          BIS    R4,MEMPAR   ;SET PARITY PRESENT BIT
9668          BIT    #PARPRE,OPTFLG ;WAS PARITY PRESENT SET BEFORE
9669          BNE    10$         ;YES - SKIP
9670 034662 013700 001716          MOV    MMVECA,R0  ;SET UP FOR PARITY TRAPS

```

9671	034666	012720	034766			MOV	#40\$, (R0)+	:	TO 40\$
9672	034672	012710	000340			MOV	#PR7, (R0)		
9673	034676	012700	072640			MOV	#OBUFF+224, R0	:	SET POINTER TO BUFFER WHERE BAD PARITY
9674								:	IS USED IN THE TESTS
9675	034702	012713	000004			MOV	#BIT2, (R3)	:	SET TO WRITE WRONG PARITY
9676	034706	012710	177777			MOV	#-1, (R0)	:	WRITE WORD BAD
9677	034712	012713	000001			MOV	#1, (R3)	:	SET TO DETECT PARITY ERROR
9678	034716	005710				TST	(R0)	:	READ BAD WORD
9679	034720	042737	000100	001664		BIC	#PARPRE, OPTFLG	:	REV 005
9680	034726	012713	000002			MOV	#2, (R3)	:	TRUN OFF CZR ERROR ENABLE
9681	034732	012710	000000			MOV	#0, (R0)	:	CLEAR BAD PARITY
9682						BIC	#7, (R3)	:	REV 006
9683						MOV	#0, (R0)	:	REV 006
9684	034736	000442				BR	35\$:	REV 005
9685	034740	062703	000002		10\$:	ADD	#2, R3	:	BUMP TO NEXT CSR
9686	034744	000241				CLC			
9687	034746	006104				ROL	R4	:	SHIFT MASK
9688	034750	001332				BNE	6\$:	TEST IF ALL DONE
9689	034752	000434				BR	35\$:	YES - SKIP
9690									
9691	034754	022626			20\$:	CMP	(SP)+, (SP)+	:	CLEAR STACK
9692	034756	012737	035040	000004		MOV	#30\$, ERRVEC	:	SET NEW NEM TRAP ADDRESS
9693	034764	000720				BR	3\$:	GO TO CSR CHECKS
9694									
9695	034766	022626			40\$:	CMP	(SP)+, (SP)+	:	CLEAR STACK
9696	034770	010337	001672			MOV	R3, CSRPTR	:	SET CSR POINTER FOR CSR TO BE USED
9697	034774	052737	000100	001664		BIS	#PARPRE, OPTFLG	:	SET PARITY PRESENT FLAG
9698	035002	012713	000000			MOV	#0, (R3)	:	TURN OFF CSR ERROR ENABLE
9699	035006	012710	000000			MOV	#0, (R0)	:	WRITE GOOD PARITY
9700	035012	012713	000001			MOV	#1, (R3)	:	SET TO DETECT PARITY ERRORS
9701	035016	000412				BR	35\$:	EXIT
9702									
9703	035020	012713	000000		47\$:	MOV	#0, (R3)	:	CLEAR CSR
9704	035024	042737	000100	001664		BIC	#PARPRE, OPTFLG	:	FORGET THEM ALL
9705	035032	005037	001670			CLR	MEMPAR	:	FORGET THEM ALL
9706	035036	000402				BR	35\$:	EXIT
9707	035040	022626			30\$:	CMP	(SP)+, (SP)+	:	CLEAR STACK
9708	035042	000736				BR	10\$:	GO CHECK NEXT CSR
9709	035044	013700	001716		35\$:	MOV	MMVECA, R0	:	SET UP POINTER TO PARITY VECTOR
9710	035050	005737	001670			TST	MEMPAR	:	TEST IF ANY PARITY PRESENT
9711	035054	001004				BNE	37\$:	YES - SKIP
9712	035056	032737	002000	001664		BIT	#CP1170, OPTFLG	:	TEST IF 11/70
9713	035064	001405				BEQ	39\$:	NO - SKIP
9714	035066	012720	034452		37\$:	MOV	#PERHLR, (R0)+	:	SET UP PARITY VECTOR
9715	035072	012710	000340			MOV	#PR7, (R0)	:	SET PRIORITY
9716	035076	000403				BR	38\$		
9717	035100	012720	000116		39\$:	MOV	#116, (R0)+	:	ELSE RESTORE TRAP CATCHER
9718	035104	005010				CLR	(R0)		
9719	035106	012737	034434	000004	38\$:	MOV	#NEXINT, ERRVEC	:	SET UP NEM VECTOR FOR LINE CLOCK TEST
9720	035114	005037	001662			CLR	INTSET	:	CLEAR INTERRUPT COUNTER
9721	035120	013700	001712			MOV	KWLVEC, R0	:	SET POINTER TO VECTOR
9722	035124	012720	034506			MOV	#LCKHLR, (R0)+	:	INSERT ADDRESS OF INTERRUPT HDLR
9723	035130	012710	000340			MOV	#PR7, (R0)	:	INSERT PRIORITY
9724	035134	012777	000100	144546		MOV	#BIT6, @KWLADD	:	LOAD KW11-L FOR INTERRUPT ENABLE
9725	035142	000240				NOP			
9726	035144	000240				NOP			

```
9727 035146 000240          NOP
9728 035150 005737 001662    TST      INTSET      ;TEST IN NEM ON KW11-P REFERENCE
9729 035154 001003          BNE      4$          ;THIS BRANCH WILL BYPASS SET UP OF
9730                                ;CLOCK OPTION
9731 035156 052737 100000 001664 4$:  BIS      #LCLKPR,OPTFLG ;SET CLOCK PRESENT FLAG
9732 035164 012701 000006          MOV      #6,R1      ;RESTORE OLD VECTOR
9733 035170 005037 001662          CLR      INTSET      ;CLEAR INT FLAG
9734 035174 012637 000006          MOV      (SP)+,ERRVEC+2
9735 035200 012637 000004          MOV      (SP)+,ERRVEC
9736 035204 012746 000000          MOV      #PRO,-(SP) ;RESTORE PRIORITY TO 0
9737 035210 012746 035216          MOV      #12$,-(SP)
9738 035214 000002          RTI
9739 035216          12$:
9740 035216 104414          RESREG
9741 035220 000207          RTS      PC
9742
9743          ;:*****
9744          .SBTTL LOOP ON INTERNAL ERROR
9745          ;* THIS ROUTINE IS USED TO PROVIDE TIGHT SCOPE LOOPS. THE CALLER
9746          ;* IS EXPECTED TO SET $LPERR TO THE START OF THE SCOPE LOOP
9747          ;* TO BE EXECUTED ON ERROR.
9748
9749 035222 032777 001000 143710 SCOP1$: BIT      #SW9,@SWR      ;CHECK IF LOOP ON ERROR
9750 035230 001405          BEQ      5$          ;NO, CONTINUE
9751 035232 105737 001103          TSTB     $ERFLG      ;CHECK IF ERROR OCCURRED
9752 035236 001402          BEQ      5$
9753 035240 013716 001110          MOV      $LPERR,(SP) ;LOAD ERROR RETURN
9754 035244 000002          5$: RTI              ;RETURN
9755          .SBTTL LINE CLOCK CALIBRATE
9756          ;* WAITS FOR A LINE CLOCK INTERRUPT TO CALIBRATE THE INTERRUPTS
9757          ;* TO A MEANINGFUL TIME VALUE. IN ADDITION IT PRESETS
9758          ;* THE TIMCNT IF THERE IS NO LINE CLOCK. TIMCNT IS USED IN THE
9759          ;* LINE CLOCK SIMULATOR.
9760
9761 035246 005037 001674          CLKCAL: CLR      LCLKTK      ;CLEAR TICK COUNTER
9762 035252 032737 100000 001664  BIT      #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
9763 035260 001004          BNE      1$          ;YES - SKIP
9764 035262 012737 000020 001654  MOV      #16.,TIMCNT ;ELSE PRESET TIMCNT
9765 035270 000410          BR       2$          ;AND EXIT
9766 035272 005737 001662          1$: TST      INTSET      ;TEST IF INTERRUPT HAS OCCURRED
9767 035276 001005          BNE      2$          ;YES- ABORT CALIBRATION
9768 035300 005737 001674          TST      LCLKTK      ;WAIT FOR CLOCK TICK
9769 035304 001772          BEQ      1$          ;NOT YET - LOOP
9770 035306 005037 001674          CLR      LCLKTK      ;CLEAR TICK COUNT
9771 035312 000207          2$: RTS      PC      ;RETURN
9772
9773          .SBTTL LINE CLOCK SIMULATION ROUTINE
9774          ;* THIS ROUTINE IS USED TO SIMULATE THE LINE CLOCK. TO
9775          ;* DO THIS THE VALUE STORED IN MILCNT IS USED AS THE
9776          ;* BASE AND REPRESENTS THE NUMBER OF TIMES A DECREMENT
9777          ;* AND BRANCH LOOP CAN BE DONE IN 1 MILLISECOND. THE
9778          ;* TIMCNT VALUE IS DECREMENTED AND IF IT REACHED 0 THE
9779          ;* LINE CLOCK TICK COUNTER IS BUMPED. THEN THE TIMCNT
9780          ;* IS RESET TO 16 (REPRESENTS 16 MILLISECONDS PER LINE CLOCK
9781          ;* TICK).
9782          ;*
```

```
9783          : *      THUS THE ROUTINE RETURNS TO THE CALLER AFTER 1 MILLISECOND
9784          : *      AND BUMPS THE LINE CLOCK TICK COUNTER FOR EACH 16 CALLS.
9785          : *
9786
9787 035314 010046          MYTIME: MOV      RO,-(SP)          ;SAVE RO
9788 035316 013700 001652      MOV      MILCNT,RO          ;SET COUNT
9789 035322 005737 001662 1$:  TST      INTSET          ;TEST IF INTERRUPT SET
9790 035326 001012          BNE      2$              ;YES - SKIP
9791 035330 005300          DEC      RO              ;DECREMENT COUNT TO ZERO
9792 035332 001373          BNE      1$
9793 035334 005337 001654      DEC      TIMCNT          ;DECREMENT TIMCNT
9794 035340 001005          BNE      2$              ;IF NOT ZERO - EXIT
9795 035342 005237 001674      INC      LCLKTK          ;ELSE BUMP TICK COUNTER
9796 035346 012737 000020 001654  MOV      #16.,TIMCNT      ;RESET TIME COUNT
9797 035354 012600          2$:  MOV      (SP)+,RO          ;RESTORE RO
9798 035356 000207          RTS      PC              ;RETURN
9799
9800          .SBTTL  WAIT FOR INTERRUPT ROUTINE
9801          : *      THE ROUTINE IS ENTERED BY ONE OF FOURTEEN TRAP CALLS. THE CALL
9802          : *      SPECIFIES HOW MANY TICKS OF THE LINE CLOCK ARE TO ELAPSE
9803          : *      WHILE WAITING FOR INTERRUPT. IF INTERRUPT DOES NOT OCCUR
9804          : *      IN THAT PERIOD OF TIME, AN ERROR MESSAGE IS PREPARED
9805          : *      (BUT NOT PRINTED IN THE ROUTINE)AND THEN RETURNS TO THE
9806          : *      LOCATION FOLLOWING THE CALL. IF INTERRUPT OCCURS THE
9807          : *      RETURN IS BUMPED BY 2. NORMALLY AN ERROR CALL WILL
9808          : *      BE IN THE LOCATION AFTER THE CALL TO INTERRUPT WAIT.
9809
9810 035360 104413          IWAT8S: SAVREG          ;ENTRY FOR 8 SECOND WAIT
9811 035362 012700 000764      MOV      #500.,RO
9812 035366 000463          BR      WATSRT
9813 035370 104413          IWAT1M: SAVREG          ;ENTRY FOR 1 MIN WAIT
9814 035372 012700 007246      MOV      #3750.,RO
9815 035376 000457          BR      WATSRT
9816 035400 104413          IWAT2S: SAVREG          ;ENTRY FOR 2 SECOND WAIT
9817 035402 012700 000200      MOV      #128.,RO
9818 035406 000453          BR      WATSRT
9819 035410 104413          IWAT1S: SAVREG          ;ENTRY FOR 1 SECOND WAIT
9820 035412 012700 000077      MOV      #63.,RO
9821 035416 000447          BR      WATSRT
9822 035420 104413          IWAT159: SAVREG          ;ENTRY FOR 160 MS WAIT
9823 035422 012700 000012      MOV      #10.,RO
9824 035426 000443          BR      WATSRT
9825 035430 104413          IWAT144: SAVREG          ;ENTRY FOR 144 MS WAIT
9826 035432 012700 000011      MOV      #9.,RO
9827 035436 000437          BR      WATSRT
9828 035440 104413          IWAT128: SAVREG          ;ENTRY FOR 128 MS WAIT
9829 035442 012700 000010      MOV      #8.,RO
9830 035446 000433          BR      WATSRT
9831 035450 104413          IWAT112: SAVREG          ;ENTRY FOR 112 MS WAIT
9832 035452 012700 000007      MOV      #7.,RO
9833 035456 000427          BR      WATSRT
9834 035460 104413          IWAT96: SAVREG          ;ENTRY FOR 96 MS WAIT
9835 035462 012700 000006      MOV      #6.,RO
9836 035466 000423          BR      WATSRT
9837 035470 104413          IWAT80: SAVREG          ;ENTRY FOR 80 MS WAIT
9838 035472 012700 000005      MOV      #5.,RO
```

```
9830 035476 000417  
9840 035500 104413  
9841 035502 012700 000004  
9842 035506 000413  
9843 035510 104413  
9844 035512 012700 000003  
9845 035516 000407  
9846 035520 104413  
9847 035522 012700 000002  
9848 035526 000403  
9849 035530 104413  
9850 035532 012700 000001  
9851 035536 012746 000000  
9852 035542 012746 035550  
9853 035546 000002  
9854  
9855 035550 012737 000020 001654 5$: MOV #16.,TIMCNT ;PRESET TIME COUNTER  
9856 035556 004737 035246 JSR PC,CLKCAL ;GO CALIBRATE THE CLOCK  
9857 035562 005737 001662 1$: TST INTSET ;TEST IF INTERRUPT OCCURRED  
9858 035566 001036 BNE 3$ ;YES - EXIT  
9859 035570 032737 100000 001664 BIT #LCLKPR,OPTFLG ;TEST IF KW11-L AVAILABLE  
9860 035576 001002 BNE 2$ ;YES - SKIP  
9861 035600 004737 035314 JSR PC,MYTIME ;ELSE CALL SIMULATOR  
9862 035604 023700 001674 2$: CMP LCLKTK,R0 ;TEST IF ENOUGH TICKS COUNTED  
9863 035610 103764 BLO 1$ ;NO - LOOP  
9864 035612 104420 TGETRK ;ELSE GET '611 REGS  
9865 035614 013701 001540 MOV T,CS1,R1 ;PUT CS1 IN R1- STRIP ALL BUT  
9866 035620 042701 177741 BIC #177741,R1 ;COMMAND CODE; INDEX INTO TABLE  
9867 035624 016137 050120 001372 MOV CMNDLB(R1),DH2N ;AND SELECT HEADER TO IDENTIFY OPERATION  
9868 035632 012737 053526 001370 MOV #EM4,EM2N ;MESSAGE (NO INTERRUPT OR INTERRUPT LATE)  
9869 035640 013700 001302 MOV $TESTN,R0 ;GET NUMBER OF PRESENT TEST  
9870 035644 006300 ASL R0 ;SHIFT FOR INDEX  
9871 035646 016037 033122 001264 MOV $$SW08TB(R0),$ESCAPE ;LOAD ESCAPE TO ABORT TESTS  
9872 035654 162737 000002 001264 SUB #2,$ESCAPE ;BUT GO TO NEXT SCOPE CALL  
9873 035662 000402 BR 4$  
9874 035664 062716 000002 3$: ADD #2,(SP) ;BUMP RETURN AROUND ERROR  
9875 035670 104414 4$: RESREG ;RESTORE REGS  
9876 035672 000002 RTI ;RETURN  
9877  
9878 .SBTTL 'L' REGISTER LOADING ROUTINE  
9879 :* THE PARAMETERS FOLLOWING THE CALL ARE TRANSFERRED INTO  
9880 :* THE 'L' REGISTERS L.CS1-L.DCYL. L.MR1 IS NOT  
9881 :* LOADED IN THIS MANNER SINCE IT IS NOT COMMONLY LOADED  
9882 :* FOR AN OPERATION. L.CS2 IS LOADED FROM DRVNUM.  
9883 :*  
9884 :* CALL: JSR R4,LRLOAD  
9885 :* COMMAND  
9886 :* WORD COUNT  
9887 :* BUS ADDRESS  
9888 :* .BYTE SECTOR ADDRESS  
9889 :* .BYTE TRACK ADDRESS  
9890 :* CYLINDER ADDRESS  
9891  
9892 035674 LRLOAD:  
9893 035674 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK  
9894 035676 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
```

```

9895 035700 012701 001600      MOV      #L.CS1,R1      ;GET ADDRESS OF L REGS
9896 035704 012700 000004      MOV      #4,R0         ;PRESET COUNT
9897 035710 012421              1$:      MOV      (R4)+,(R1)+   ;MOVE FIRST FOUR WORDS INTO 'L' REGS
9898 035712 005300              DEC      R0            ;CS1, WC, BA, DA
9899 035714 001375              BNE     1$
9900 035716 013721 001626      MOV      DRVNUM,(R1)+  ;LOAD DRIVE NUMBER
9901 035722 005721              TST     (R1)+         ;BUMP PAST ASOF
9902 035724 012411              MOV     (R4)+,(R1)    ;LOAD DCYL
9903 035726 012601              MOV     (SP)+,R1      ;;POP STACK INTO R1
9904 035730 012600              MOV     (SP)+,R0      ;;POP STACK INTO R0
9905 035732 000204              RTS      R4
9906
9907
9908      .SBTTL  LOAD RK611 FOR OPERATION
9909      ;*    THE REGISTER SETUP STORAGE IS TRANSFERRED TO THE RK611 REGISTER.
9910      ;*    THIS IS A STRAIGHT TRANSFER WITH NO CHECKING OR MANIPULATION
9911      ;*    OF THE REGISTER CONTENTS.  L.CS1 IS TRANSFERRED LAST AS IT
9912      ;*    SHOULD BE IF THE GO BIT IS SET.
9912 035734 005037 001662      LOADRK: CLR     INTSET      ;CLEAR INTERRUPT FLAG.
9913 035740 010046              MOV     R0,-(SP)      ;STORE REGISTER
9914 035742 010146              MOV     R1,-(SP)
9915 035744 012700 001602      MOV     #L.WC,R0     ;GET ADDRESS OF SETUP STORAGE WC
9916 035750 010201              MOV     R2,R1        ;GET BASE ADDRESS
9917 035752 062701 000002      ADD     #2,R1        ;PUT R1 PAST RKCS1
9918 035756 012021              MOV     (R0)+,(R1)+  ;LOAD WORD COUNT
9919 035760 012021              MOV     (R0)+,(R1)+  ;LOAD BUS ADDRESS
9920 035762 012021              MOV     (R0)+,(R1)+  ;LOAD DISK ADDRESS
9921 035764 012011              MOV     (R0)+,(R1)   ;LOAD CS2
9922 035766 062701 000006      ADD     #6,R1        ;BUMP R1 TO ASOF
9923 035772 012021              MOV     (R0)+,(R1)+  ;LOAD OFFSET
9924 035774 012021              MOV     (R0)+,(R1)+  ;LOAD CYLINDER
9925 035776 062701 000004      ADD     #4,R1        ;BUMP R1 TO MR1
9926 036002 011011              MOV     (R0),(R1)    ;LOAD MR1
9927 036004 123727 001102 000007      CMPB   $TSTNM,#7    ;SEE IF TEST 7
9928 036012 001403              BEQ     1$           ;BR IF YES
9929 036014 053737 001720 001600      BIS     DTYPE,L.CS1  ;ELSE SET CORRECT DRIVE TYPE
9930 036022 013712 001600      1$:      MOV     L.CS1,(R2)   ;LOAD CS1
9931 036026 012601              MOV     (SP)+,R1     ;RESTORE REGISTER
9932 036030 012600              MOV     (SP)+,R0
9933 036032 000002              RTI
9934
9935
9936      ;THIS ROUTINE LOADS CONSTANTS EITHER FOR THR RK06 OR THE RK07
9937
9938 036034 005737 001720      LDCON: TST     DTYPE   ;SEE IF RK06
9939 036040 001414              BEQ     1$           ;BR IF YES
9940 036042 012737 001456 036120      MOV     #1456,LSTCYL ;ELSE LOAD RK07 CONSTANTS
9941 036050 012737 001000 036122      MOV     #1000,TSTCYL ;MAY NOT BE NEEDED
9942 036056 005037 036130              CLR     HOLD2
9943 036062 012737 000040 036132      MOV     #DTYE,HOLD3
9944 036070 000207              RTS      PC
9945
9946 036072 012737 000632 036120 1$:      MOV     #632,LSTCYL  ;RK06 CONSTANTS
9947 036100 005037 036122              CLR     TSTCYL      ;MAY NOT BE NEEDED
9948 036104 012737 002000 036130      MOV     #CDT,HOLD2
9949 036112 005037 036132              CLR     HOLD3
9950 036116 000207              RTS      PC

```

```
9951
9952 036120 000000          LSTCYL: 0
9953 036122 000000          TSTCYL: 0
9954 036124 000000          HOLD: 0
9955 036126 000000          HOLD1: 0
9956 036130 000000          HOLD2: 0
9957 036132 000000          HOLD3: 0
9958
9959
9960
9961
9962 036134 006300          ; THESE ROUTINES LOAD THE CORRECT DRIVE TYPE INTO DTYPE
9963 036136 016037 001722 001720 CTYP0: ASL      R0
9964 036144 006200          MOV      TYPTBL(R0),DTYPE
9965 036146 000207          ASR      R0
9966
9967 036150 006301          GTP1: ASL      R1
9968 036152 016137 001722 001720 MOV      TYPTBL(R1),DTYPE
9969 036160 006201          ASR      R1
9970 036162 000207          RTS      PC
9971
9972
9973
9974
9975
9976
9977
9978 036164 010046          .SBTTL STORE RK611 REGISTERS
9979 036166 010146          ;* ALL THE RK611 REGISTERS ARE STORED IN THE TEST TABLE T
9980 036170 010346          ;* WITH THE EXCEPTION OF THE DATA BUFFER WHICH IS NOT STORED IN
9981 036172 012700 001540          ;* THIS ROUTINE.
9982 036176 010201          GETRK: MOV      R0,-(SP)          ;STORE REGISTERS TO BE USED
9983 036200 012703 000012          MOV      R1,-(SP)
9984 036204 012120          MOV      R3,-(SP)
9985 036206 005303          MOV      #T.CS1,R0          ;SET POINTER TO TEST TABLE
9986 036210 001375          MOV      R2,R1              ;SET POINTER TO RK611 BASE
9987 036212 062701 000002          MOV      #10.,R3           ;SET COUNT FOR 1ST 10 REGS
9988 036216 005720          1$: MOV      (R1)+,(R0)+      ;STORE RKCS1 THROUGH RKSPAR
9989 036220 012703 000004          DEC      R3
9990 036224 012120          BNE      1$
9991 036226 005303          ADD      #2,R1              ;BUMP POINTER PAST RKDB
9992 036230 001375          TST      (R0)+             ;BUMP POINTER PAST T.RKDB
9993 036232 012603          MOV      #4,R3              ;SET COUNT FOR LAST 5 REGS
9994 036234 012601          2$: MOV      (R1)+,(R0)+      ;STORE RKMR1 THROUGH RKMR3
9995 036236 012600          DEC      R3
9996 036240 000002          BNE      2$
9997
9998
9999
10000
10001
10002 036242 016637 000002 001256 BITCNT: MOV      2(SP),STMP16      ;GET WORD WHOSE BITS ARE TO BE COUNTED
10003 036250 010346          MOV      R3,-(SP)          ;STORE R3
10004 036252 005037 001260          CLR      STMP17            ;CLEAR STMP16 FOR COUNTING
10005 036256 012703 000021          MOV      #17.,R3          ;SET A SHIFT COUNTER
10006 036262 000241          CLC                       ;CLEAR CARRY
```

```
10007 036264 006037 001256 1$: ROR $TMP16 ;ROTATE WORD.
10008 036270 103407 BCS 3$ ;WAS BIT SHIFTED OUT A 1?
10009 036272 005303 2$: DEC R3 ;NO - DEC COUNT
10010 036274 001373 BNE 1$ ;LOOP IF NOT ZERO
10011 036276 012603 MOV (SP)+,R3 ;RESTORE R3
10012 036300 013766 001260 000002 MOV $TMP17,2(SP) ;PUT COUNT OF BITS ON STACK
10013 036306 000204 RTS R4 ;RETURN
10014 036310 005237 001260 3$: INC $TMP17 ;BUMP COUNT
10015 036314 000766 BR 2$ ;LOOP
10016
10017 .SBTTL MAINTENANCE CLOCK ROUTINE
10018 ;* THE PARAMETERS PASSED TO THIS ROUTINE ARE LOCATED IN THE
10019 ;* ADDRESS OFTER THE CALL. THE FIRST BYTE CANTAINS THE NUMBER
10020 ;* OF PHASES ADDRESSES THE CALLING ROUTINE WANTS THE CONTROLLER
10021 ;* CLOCKED THROUGH AND THE SECOND BYTE CONTAINS THE NUMBER OF
10022 ;* CLOCK TRANSITIONS(PARTIAL PHASES) TO BE DONE.
10023
10024 036316 MCLOCK:
10025 036316 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
10026 036320 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
10027 036322 112400 MOV#B (R4)+,R0 ;GET NUMBER OF CONTROLLER PHASE ADDRESSES
10028 036324 112401 MOV#B (R4)+,R1 ;GET PARTIAL PHASE ADDRESS COUNT
10029
10030 036326 006300 ASL R0 ;MULTIPLY PHASE ADDRESS COUNT BY 4
10031 036330 006300 ASL R0
10032 036332 060100 ADD R1,R0 ;ADD IN PARTIALS
10033 036334 052762 000400 000026 1$: BIS #MCLK,RKMR1(R2) ;SET CLOCK
10034 036342 042762 000400 000026 BIC #MCLK,RKMR1(R2) ;CLEAR MCLK
10035 036350 005300 DEC R0 ;DECREMENT COUNT
10036 036352 001370 BNE 1$ ;LOOP IF NOT ZERO
10037 036354 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
10038 036356 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
10039 036360 000204 RTS R4
10040 .SBTTL READ AND SORT HEADERS
10041 ;* THE HEADERS IN THE CYLINDER AND TRACK SPECIFIED BY
10042 ;* THE FIELDS IN THE 'L' REGISTERS ARE READ AND STORED IN
10043 ;* ASSCENDING ORDER. CONTROLLER ERRORS ARE CHECKED IN THE
10044 ;* READ HEADER OPERATION AND DATA LATE IS CHECKED AFTER
10045 ;* EACH READ OF THE DATA BUFFER.
10046 ;*
10047 ;* CALL: JSR R4,RDSTHD
10048 ;* TCHKOP ;RETURN POINT IF CERR IN READ HDR
10049 ;* ERROR 4 ;OR 5, 6, 7
10050 ;* ERROR 13 ;RETURN IF DATA LATE IN DB UNLOAD
10051 ;* ERROR 2 ;RETURN IF TO SLOW OR
10052 ;* ;IF HDR 0 NOT FOUND
10053
10054 036362 104413 RDSTHD: SAVREG
10055 036364 032737 100000 001664 BIT #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
10056 036372 001402 BEQ 20$ ;NO - SKIP
10057 036374 005077 143310 CLR @KWLADD ;RESET INTERRUPT
10058 036400 012700 000026 20$: MOV #26,R0 ;PRESET FOR 26 SECTOR FORMAT
10059 036404 032737 010000 001600 BIT #CFMT,L.CS1 ;IS 24 SECTOR MODE SET?
10060 036412 001402 BEQ 1$ ;NO - SKIP
10061 036414 012700 000024 MOV #24,R0 ;ELSE CHANGE TO 24 SECTOR MODE
10062 036420 012701 070414 1$: MOV #IBUFF,R1 ;SET POINTER TO INPUT BUFFER
```


10063	036424	010005				MOV	R0,R5	;SAVE NUMBER OF SECTORS
10064	036426	010104				MOV	R1,R4	;SAVE Ibuff ADDRESS
10065	036430	010203				MOV	R2,R3	;SET UP POINTER TO RKDB
10066	036432	062703	000024			ADD	#RKDB,R3	
10067	036436	013762	001626	000010		MOV	DRVNUM,RKCS2(R2)	;LOAD DRIVE N'JM
10068	036444	013762	001614	000020		MOV	L.DCYL,RKDCYL(R2)	;LOAD CYLINDER NUM
10069	036452	013762	001606	000006		MOV	L.DA,RKDA(R2)	;LOAD TRACK AND SECTOR
10070								
10071	036460	012737	000020	001654	2\$:	MOV	#16.,TIMCNT	;SET TIME COUNTER
10072	036466	005037	001662			CLR	INTSET	;CLEAR INTERRUPT FLAG
10073	036472	005037	001674			CLR	LCLKTK	;CLEAR TICK COUNTER
10074	036476	053737	001720	001600		BIS	DTYPE,L.CS1	;SET DRV TYP
10075	036504	013762	001600	000000		MOV	L.CS1,RKCS1(R2)	;LOAD COMMAND
10076								
10077	036512	005737	001662		3\$:	TST	INTSET	;TEST IF INT OCCURRED
10078	036516	001020				BNE	4\$;YES - SKIP
10079	036520	004737	035314			JSR	PC,MYTIME	;WAIT 1 MS
10080	036524	005737	001674			TST	LCLKTK	;HAVE WE WAITED 16 MS?
10081	036530	001770				BEQ	3\$;NO - LOOP ON WAIT
10082								
10083	036532	062766	000006	000006		ADD	#6,6(SP)	;SET RETURN FOR TO SLOW
10084	036540	104420				TGETRK		;GET RK REGS
10085	036542	012737	053526	001370		MOV	#EM4,EM2N	;LOAD MESSAGE "TO SLOW/NOT COMPLETE"
10086	036550	012737	050324	001372		MOV	#OPER24,DH2N	;LOAD COMMAND "READ HEADER" FOR REPORT
10087	036556	000466				BR	10\$;SKIP
10088								
10089	036560	005762	000000		4\$:	TST	RKCS1(R2)	;TEST FOR CONTROLLER ERROR
10090	036564	100474				BMI	11\$;YES - SKIP
10091								
10092	036566	011324				MOV	(R3),(R4)+	;STORE HEADERS
10093	036570	011324				MOV	(R3),(R4)+	
10094	036572	011324				MOV	(R3),(R4)+	
10095								
10096	036574	005762	000010			TST	RKCS2(R2)	;TEST IF DATA LATE
10097	036600	100443				BMI	8\$;YES - SKIP
10098								
10099	036602	005300				DEC	R0	;DEC SECTOR COUNT
10100	036604	001325				BNE	2\$;IF NOT ZERO - LOOP
10101								
10102	036606	032737	100000	001664		BIT	#LCLKPR,OPTFLG	;TEST IF CLOCK PRESENT
10103	036614	001403				BEQ	5\$;NO - SKIP
10104	036616	012777	000100	143064		MOV	#BIT6,@KWLADD	;SET INTERRUPT ENABLE
10105	036624	032761	000037	000002	5\$:	BIT	#37,2(R1)	;HEADER AT TOP OF BUFF=HEAD 0?
10106	036632	001413				BEQ	6\$;YES - SKIP
10107	036634	012124				MOV	(R1)+,(R4)+	;ELSE MOV THIS HEADER TO BOTTOM
10108	036636	012124				MOV	(R1)+,(R4)+	
10109	036640	012124				MOV	(R1)+,(R4)+	
10110								
10111	036642	005305				DEC	R5	;TEST FO INSURE HEAD 0 IS FOUND
10112	036644	001367				BNE	5\$;IF ALL HEADERS NOT CHECKED - LOOP
10113	036646	012737	056276	001370		MOV	#EM56,EM2N	;ELSE "HEADER 0 NOT FOUND" MESSAGE
10114	036654	005037	001372			CLR	DH2N	
10115	036660	000421				BR	9\$;SKIP
10116								
10117	036662	012700	070414		6\$:	MOV	#IBUFF,R0	;GET TOP OF Ibuff
10118	036666	012120			7\$:	MOV	(R1)+,(R0)+	;MOVE HEADERS SO THEY START AT TOP OF Ibuff

```

10119 036670 012120      MOV      (R1)+,(R0)+
10120 036672 012120      MOV      (R1)+,(R0)+
10121 036674 020004      CMP      R0,R4          ;ALL HEADERS MOVED?
10122 036676 001373      BNE      7$            ;NO - LOOP
10123
10124 036700 062766 000010 000006      ADD      #10,6(SP)      ;SET UP FOR GOOD RETURN
10125 036706 000423      BR       11$
10126
10127 036710 012737 054541 001500 8$:      MOV      #EMDLT,EM13N   ;'DATA LATE SET RESULT OF READ DB'
10128 036716 012737 056707 061022      MOV      #EMRDB,DF011A
10129 036724 062766 000004 000006 9$:      ADD      #4,6(SP)      ;SET ERROR RETURN
10130 036732 104420      TGETRK   ;GET RK REGS
10131 036734 013700 001302 10$:     MOV      $TESTN,R0     ;GET TEST NUMBER
10132 036740 006300      ASL      R0            ;SHIFT FOR INDEX
10133 036742 016037 033122 001264      MOV      $$W08TB(R0),$ESCAPE ;SET ESCAPE
10134 036750 162737 000002 001264      SUB      #2,$ESCAPE    ;TO NEXT SCOPE CALL
10135
10136 036756 104414      11$:     RESREG
10137 036760 000204      RTS      R4
10138
10139      .SBTTL  GET DRIVE STATUS
10140      :*      THIS ROUTINE GETS ALL THE DRIVE STATUS AND PLACES IT IN $REG10
10141      :*      THROUGH $REG17.  THESE REGISTORS ARE FIRST CLEARED TO ALL ONES AND
10142      :*      THEN IF ERROR OCCURS WHILE GETTING STATUS, THE 1'S ARE LEFT
10143      :*      IN THE REGISTERS.
10144      :*
10145      :*CALL: JSR      R4,GETDRS
10146      :*      BR       ERROR PROCESSING      ERROR RETURN
10147      :*      BR       NO EROR PROCESSING    GOOD RETURN
10148
10149 036762 104413      GETDRS: SAVREG
10150 036764 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR ANY OLD ERRORS LAYING AROUND
10151 036772 012700 001202      MOV      #$REG10,R0     ;PRESET ALL STATUS STORAGE TO
10152 036776 012701 000010      MOV      #8.,R1         ;ALL ONES
10153 037002 012720 177777      1$:      MOV      #177777,(R0)+
10154 037006 005301      DEC      R1
10155 037010 001374      BNE      1$
10156 037012 012700 001206      MOV      #$REG12,R0     ;SET POINTER TO REG12 FOR A01 & B01
10157 037016 012701 000001      MOV      #1,R1          ;PRESET FOR PAIR ONE.
10158 037022 005037 001230      CLR      $TMP3          ;CLEAR ERROR SWITCH
10159 037026 013762 001610 000010 2$:     MOV      L.CS2,RKCS2(R2) ;LOAD DRIVE #
10160 037034 010162 000026      MOV      R1,RKMR1(R2)   ;LOAD MR1
10161 037040 012762 000001 000000      MOV      #BIT0,RKCS1(R2) ;DO SELECT
10162 037046 012703 000050      MOV      #40.,R3        ;WAIT FOR A FEW MICRO RECORDS TO
10163 037052 005303      3$:      DEC      R3             ;BIT SELECT FINISH
10164 037054 001376      BNE      3$
10165 037056 032762 100000 000000      BIT      #CERR,RKCS1(R2) ;ANY ERROR SET AS A RESULT OF SELECT?
10166 037064 001415      BEQ      4$            ;NO - SKIP
10167 037066 032762 024000 000000      BIT      #CTO!SPAR,RKCS1(R2) ;CHECK IF TIMEOUT OR PARITY ERROR
10168 037074 001004      BNE      8$            ;YES - SKIP
10169 037076 032762 037400 000010      BIT      #37400,RKCS2(R2) ;TEST FOR ERRORS:
10170      ;      NED!UPE!MDS!UFE!NEM!PGE
10171 037104 001405      BEQ      4$            ;NO - SKIP
10172 037106 012737 000001 001230 8$:     MOV      #1,$TMP3       ;SET ERROR FLAG
10173 037114 022020      CMP      (R0)+,(R0)+    ;BUMP TO LET THAT PAIR STAY ALL 1'S.
10174 037116 000404      BR       5$            ;SKIP

```

```

10175 037120 016220 000034      4$:  MOV    RKMR2(R2),(R0)+  ;STORE A WORD
10176 037124 016220 000036      MOV    RKMR3(R2),(R0)+  ;STORE B WORD
10177 037130 012762 100000 000000 5$:  MOV    #CCLR,RKCS1(R2) ;CLEAR ANY OLD ERROR IN CONTROLLER
10178 037136 005701              TST    R1                ;IS R1 A 0 (LAST TRANSFER, PAIR 0)
10179 037140 001410              BEQ    6$                ;YES - SKIP
10180 037142 005201              INC    R1                ;ELSE BUMP TO NEXT PAIR
10181 037144 022701 000004      CMP    #4,R1            ;PAIR 3 JUST STORED?
10182 037150 001326              BNE    2$                ;NO - SKIP
10183 037152 005001              CLR    R1                ;ELSE SET TO PAIR 0
10184 037154 012700 001202      MOV    #SREG10,R0       ;PRESET POINTER FOR PAIR 0
10185 037160 000722              BR     2$                ;GO GET THEM
10186 037162 104414      6$:  RESREG ;EXIT HERE
10187 037164 005737 001230      TST    $TMP3            ;ANY ERROR IN STATUS GETTING
10188 037170 001001              BNE    7$                ;YES - SKIP
10189 037172 005724              TST    (R4)+            ;ELSE BUMP PART ERROR
10190 037174 000204      7$:  RTS     R4                ;RETURN
10191
10192      .SBTTL  SUBSYSTEM INITIALIZE AND INITIALIZE STATE TEST
10193      :*      THE SUBSYSTEM IS INITIALIZED WITH A SUBSYSTEM CLEAR
10194      :*      COMMAND. CERR AND DI ARE MONITORED FOR A SHORT
10195      :*      PERIOD OF TIME DURING WHICH THEY SHOULD BOTH RESET.
10196      :*
10197      :*      IF THEY DO RESET, READY IS TESTED TO INSURE IF SETS.
10198      :*
10199      :*      IF ANY OF THESE THREE CONDITIONS ARE NOT MET AN APPROPRIATE
10200      :*      ERROR MESSAGE IS PREPARED AND REPORTED WHEN THE ROUTINE
10201      :*      RETURN TO THE CALL. IF EVERY THING IS GOOD, THE RETURN
10202      :*      SKIPS OVER THE ERROR CALL AND TEST ABORT.
10203      :*
10204      :*      THE USUAL CALL TO THIS ROUTINE WILL BE FOLLOWED BY
10205      :*      AN ERROR MESSAGE AND BRANCH TO END OF TEST. THIS
10206      :*      IS DONE BECAUSE FAILURE TO INITIALIZE CORRECTLY IS FATAL TO
10207      :*      THE TEST.
10208
10209 037176      SSINIT:
10210 037176 010046      MOV    R0,-(SP)         ;;PUSH R0 ON STACK
10211 037200 010146      MOV    R1,-(SP)         ;;PUSH R1 ON STACK
10212 037202 012701 000007      MOV    #7,R1           ;SET CLEAR COUNT
10213 037206 012700 001600      MOV    #L.CS1,R0       ;GET ADDRESS OF 'L' REGS
10214 037212 012720 000100      MOV    #100,(R0)+      ;PRESET CS1 INTR ENABLE
10215 037216 005020      7$:  CLR    (R0)+            ;CLEAR THE NEXT
10216 037220 005301              DEC    R1                ;COUNT 0?
10217 037222 001375              BNE    7$                ;NO - LOOP
10218 037224 012762 000040 000010      MOV    #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
10219 037232 012737 000012 001222      MOV    #10,$TMP0       ;SET A COUNTER
10220 037240 016237 000000 001540 1$:  MOV    RKCS1(R2),T.CS1 ;GET CS1
10221 037246 032737 140000 001540      BIT    #CERR!DI,T.CS1 ;TEST IF ERROR OR DI SET
10222 037254 001433              BEQ    2$                ;NO - SKIP TO READY TEST
10223 037256 005337 001222      DEC    $TMP0           ;ELSE DECREMENT COUNTER
10224 037262 001366              BNE    1$                ;AND LOOP
10225 037264 032737 100000 001540      BIT    #CERR,T.CS1    ;TEST - IS IT CERR STILL SET
10226 037272 001404              BEQ    3$                ;NO - SKIP TO DI MESSAGE
10227 037274 012737 053621 001400      MOV    #EM5,EM3N       ;MESSAGE (SUBSYS CLR NOT RESET ERROR)
10228 037302 000403              BR     4$
10229 037304 012737 053665 001400 3$:  MOV    #EM6,EM3N       ;MESSAGE (SUBSYS CLEAR NOT RESET DI)
10230 037312 104420      4$:  TGETRK

```

```

10231 037314 013700 001302      MOV    $TESTN,R0      ;GET PRESENT TEST NUMBER
10232 037320 006300              ASL    R0              ;SHIFT FOR INDEX
10233 037322 016037 033122 001264  MOV    $$W08TBL(R0), $ESCAPE ;LOAD ESCAPE TO ABORT TEST
10234 037330 162737 000002 001264  SUB    #2,$ESCAPE     ;SET TO NEXT SCOPE CALL
10235 037336 012601              MOV    (SP)+,R1       ;;POP STACK INTO R1
10236 037340 012600              MOV    (SP)+,R0       ;;POP STACK INTO R0
10237 037342 000414              BR     6$             ;DSKIP TO EXIT
10238 037344 032737 000200 001540 2$:   BIT    #RDY,T.CS1     ;TEST READY SET
10239 037352 001004              BNE    5$             ;YES - GOOD EXIT
10240 037354 012737 053744 001400      MOV    #EM7,EM3N     ;MESSAGE (SUBSYS CLR NOT SET READY)
10241 037362 000753              BR     4$
10242 037364 012601 5$:     MOV    (SP)+,R1       ;RESTORE REGS
10243 037366 012600              MOV    (SP)+,R0
10244 037370 062716 000002      ADD    #2,(SP)       ;GOOD RETURN
10245 037374 000002 6$:     RTI
10246
10247      .SBTTL WORD COUNT AT END OF OPERATION CHECK
10248      ;* THIS ROUTINE COMPARES THE CONTENTS OF THE TEST STORAGE FOR
10249      ;* THE WORD COUNT AGAINST THE SUPPLIED VALUE. IF UNEQUAL, THE
10250      ;* ERROR FLAG (WCERR) IS SET IN GROUP 4 ERROR FLAGS (GRP4ER)
10251      ;*
10252      ;*CALL: JSR    R4,CHKWC
10253      ;*      .WORD      ;EXPECTED WC VALUE
10254
10255 037376 012437 047744      CHKWC: MOV    (R4)+,EXPWC ;STORE EXPECTED VALUE
10256
10257 037402 023737 047744 001542      CMP    EXPWC,T.WC    ;COMPARE
10258 037410 001406              BEQ    1$            ;EQUAL - SKIP
10259 037412 052737 000001 050002      BIS    #WCERR,GRP4ER ;SET ERROR FLAG
10260 037420 013737 001542 047760      MOV    T.WC,REALWC  ;STORE REAL WORD COUNT
10261 037426 000204 1$:     RTS    R4            ;RETURN.
10262
10263      .SBTTL BUS ADDRESS AT END OF OPERATION CHECK
10264      ;* THIS ROUTINE COMPUTES THE EXPECTED BUS ADDRESS AT THE END OF
10265      ;* A TRANSFER BY USING THE INITIAL BUS ADDRESS, ADDING IN THE
10266      ;* INITIAL WORD COUNT, AND SUBTRACTING ANY RESIDUAL WORD COUNT.
10267      ;* IF THIS COMPUTED BA DOES NOT EQUAL THE CONTENTS OF RKBA
10268      ;* AN ERROR FLAG (BAERR) IS SET IN GROUP 4 ERROR FIELD (GRP4ER)
10269      ;*
10270      ;* IF BUS ADDRESS INCREMENT INHIBIT WAS SET, THE EXPECTED BUS
10271      ;* ADDRESS IS THE STARTING BUS ADDRESS.
10272      ;*CALL: JSR    R4,CHKBA
10273
10274 037430 010046      CHKBA: MOV    R0,-(SP)
10275 037432 010146              MOV    R1,-(SP)
10276 037434 010346              MOV    R3,-(SP)
10277 037436 032737 000020 001610      BIT    #BAI,L.CS2   ;TEST IF BAI SET
10278 037444 001404              BEQ    4$            ;NO - SKIP
10279 037446 013737 001604 047750      MOV    L.BA,EXPBA   ;STORE EXPECTED BA (SAME AS STARTING BA)
10280 037454 000441              BR     3$
10281 037456 013700 001602 4$:     MOV    L.WC,R0      ;GET INITIAL WORD COUNT
10282 037462 005400              NEG    R0
10283 037464 113703 001601      MOVB   L.CS1+1,R3   ;GET BA16 & BA17
10284 037470 042703 177774      BIC    #177774,R3   ;CLEAR UNWANTED BITS
10285
10286 037474 005700              TST    R0            ;TEST IF INITIAL WORD COUNT 0
  
```

10287	037476	001003		BNE	6\$:NO - SKIP
10288	037500	062703	000002	ADD	#2,R3	:ADD 2 TO BA16,17 (65K WORD XFER)
10289	037504	000407		BR	9\$:SKIP
10290	037506	005700		6\$: TST	R0	:TEST IF INITIAL WC BIT 15 SET
10291	037510	100001		BPL	5\$:NO - SKIP
10292	037512	005203		INC	R3	:BUMP BA16,17 (32K WORD XFER)
10293	037514	006300		5\$: ASL	R0	:SHIFT WORD COUNT TO MAKE MEM ADD CNT
10294	037516	063700	001604	ADD	L.BA,R0	:ADD IN START BUFFER ADD
10295	037522	005503		ADC	R3	:IF CARRY - ADD INTO BA16,17
10296	037524	013701	001542	9\$: MOV	T.WC,R1	:GET END OF OPERATION WORD COUNT
10297	037530	001411		BEQ	1\$:BRANCH IF ZERO
10298	037532	005401		NEG	R1	
10299	037534	005701		TST	R1	:TEST END OPERATION WC BIT 15 SET
10300	037536	100001		BPL	7\$:NO - SKIP
10301	037540	005303		DEC	R3	:DEC BA 16,17 (32K WC LEFT)
10302	037542	006301		7\$: ASL	R1	:SHIFT WC TO MAKE MEM ADD CNT
10303	037544	160100		SUB	R1,R0	:SUB FROM COMPUTED BUS ADDRESS
10304	037546	005603		SBC	R3	:SUB CARRY FROM BA16,17
10305	037550	010337	047746	MOV	R3,EXPUBA	:STORE EXPECTED UPPER BA BITS
10306	037554	010037	047750	1\$: MOV	R0,EXPBA	
10307	037560	020037	001544	3\$: CMP	R0,T.BA	:EQUAL TO COMPUTED?
10308	037564	001406		BEQ	2\$:YES - SKIP
10309	037566	052737	000004 050002	BIS	#BAERR,GRP4ER	:ELSE SET BAERR FLAG
10310	037574	013737	001544 047764	MOV	T.BA,REALBA	:STORE REAL BUS ADDRESS
10311	037602	113703	001541	2\$: MOVB	T.CS1+1,R3	:GET REAL UPPER BA
10312	037606	042703	177774	BIC	#177774,R3	:CLEAR UNWANTED BITS
10313	037612	020337	047746	CMP	R3,EXPUBA	:CHECK IF EQUAL
10314	037616	001405		BEQ	8\$:YES - SKIP
10315	037620	052737	000002 050002	BIS	#UBAERR,GRP4ER	:ELSE SET UBA ERROR
10316	037626	010337	047762	MOV	R3,REALUB	:STORE REAL UPPER BA
10317	037632	012603		8\$: MOV	(SP)+,R3	
10318	037634	012601		MOV	(SP)+,R1	
10319	037636	012600		MOV	(SP)+,R0	
10320	037640	000204		RTS	R4	

```

10321
10322 .SBTTL CYLINDER,TRACK,SECTOR TEST AT END OF OPERATION
10323 :* THIS ROUTINE CHECKS THAT THE CONTENTS OF THE RKDCYL AND RKDA
10324 :* ARE CORRECT FOR ANY SIZE DATA TRANSFER AT THE END OF THE
10325 :* OPERATION. THE CONTENTS OF THE LOAD REGISTER STORAGE ARE
10326 :* COUNTED ON TO HAVE THE INITIAL VALUES TO MAKE THE
10327 :* NECESSARY CALCULATION.
10328 :*
10329 :* ALL THREE VALUES ARE GENERATED AND STORED IN EXPECTED VALUES
10330 :* STORAGE EXPCYL, EXPTRK, EXPSEC. ALL 3 ARE CHECKED AND
10331 :* IF ONE OR MORE ARE WRONG, THE CORRESPONDING BIT IN THE
10332 :* ERROR FLAGS FIELD (GRP4ER) IS SET.
10333 :*
10334 :*CALL: JSR R4,CHKCTS
10335

```

10336	037642	104413		CHKCTS: SAVREG		
10337	037644	013700	001602	MOV	L.WC,R0	:GET SPECIFIED WORD COUNT
10338	037650	005400		NEG	R0	:NEGATE IT
10339	037652	013701	001542	MOV	T.WC,R1	:GET END OF OPERATION WORD COUNT
10340	037656	001401		BEQ	10\$:IF ZERO - SKIP
10341	037660	005401		NEG	R1	:NEGATE IT
10342	037662	160100		10\$: SUB	R1,R0	:COMPUTE ACTUAL WORDS TRANSFERRED

```

10343 037664 005001          CLR      R1          ;CLEAR R1 FOR COUNTING
10344          :      THE FOLLOWING CODE DETERMINES HOW MAY SECTORS OF DATA HAS BEEN
10345          :      TRANSFERRED IN THE OPERATION. ONCE IT HAS COMPUTED THAT, THE
10346          :      END OF OPERATION VALUES FOR THE CYLINDER, TRACK, AND SECTOR
10347          :      IS CALCULATED.
10348 037666 022700 000400 1$:  CMP      #400,R0
10349 037672 003004          BGT      2$
10350 037674 005201          INC      R1
10351 037676 162700 000400  SUB      #400,R0
10352 037702 000771          BR       1$
10353 037704 005700          2$:  TST      R0
10354 037706 001401          BEQ      3$
10355 037710 005201          INC      R1
10356          :      AT THIS POINT R1 HAS A COUNT OF THE
10357          :      NUMBER OF FULL SECTOR TRANSFER + 1 IF A
10358          :      PARTIAL SECTOR WAS TRANSFERRED.
10359 037712 012703 000026 3$:  MOV      #26,R3
10360 037716 032737 010000 001600 BIT      #CFMT,L.CS1
10361 037724 001402          BEQ      4$
10362 037726 012703 000024  MOV      #24,R3
10363          :      R3 HAS BEEN SET UP WTH THE NUMBER
10364          :      OF SECTORS IN A TRACK FOR THE FORMAT USED
10365 037732 013737 001614 047752 4$:  MOV      L.DCYL,EXPCYL ;GET STARTING VALUES FOR CYLINDER
10366 037740 113704 001607          MOVB     L.DT,R4      ;TRACK
10367 037744 042704 177400          BIC      #177400,R4
10368 037750 113705 001606          MOVB     L.DS,R5      ;SECTOR
10369 037754 042705 177400          BIC      #177400,R5
10370 037760 005301          DEC      R1          ;ADJUST COUNT FOR ZERO DETECT
10371 037762 005205          5$:  INC      R5          ;BUMP SECTOR COUNT
10372 037764 020503          CMP      R5,R3      ;DID THIS MAKE SECTOR COUTN > 1 TRACK?
10373 037766 001010          BNE     6$          ;NO - SKIP
10374 037770 005005          CLR      R5          ;ELSE CLEAR SECTOR COUNT
10375 037772 005204          INC      R4          ;BUMP TRACK COUNT
10376 037774 022704 000003          CMP      #3,R4      ;DID THIS MAKE TRK COUNT > 1 CYLINDER?
10377 040000 001003          BNE     6$          ;NO - SKIP
10378 040002 005004          CLR      R4          ;ELSE CLEAR TRACK COUNT
10379 040004 005237 047752          INC      EXPCYL     ;BUMP CYLINDER COUNT
10380 040010 005301          6$:  DEC      R1          ;DEC COUNT
10381 040012 001363          BNE     5$          ;IF ZERO - EXIT
10382 040014 010437 047756          MOV      R4,EXPTRK  ;STORE EXPECTED TRACK
10383 040020 010537 047754          MOV      R5,EXPSEC  ;STORE EXPECTED SECTOR (CYL ALREADY SLOW)
10384 040024 023737 001560 047752  CMP      T.DCYL,EXPCYL ;TEST IF CYLINDER OK
10385 040032 001403          BEQ      7$          ;YES - SKIP
10386 040034 052737 000010 050002  BIS      #CYLERR,GRP4ER ;NO - SET ERROR FLAG
10387 040042 120437 001547          7$:  CMPB     R4,T.DA+1   ;TEST TRACK OK
10388 040046 001403          BEQ      8$          ;YES - SKIP
10389 040050 052737 000020 050002  BIS      #TRKERR,GRP4ER ;NO - SET ERROR FLAG
10390 040056 120537 001546          8$:  CMPB     R5,T.DA      ;TEST SECTOR COUNT OK
10391 040062 001403          BEQ      9$          ;YES - SKIP
10392 040064 052737 000040 050002  BIS      #SECERR,GRP4ER ;USE SET ERROR FLAG
10393 040072 012700 047760          9$:  MOV      #REALWC,R0
10394 040076 013720 001542          MOV      T.WC,(R0)+ ;STORE REAL WORD COUNT
10395 040102 013720 001544          MOV      T.BA,(R0)+ ;STORE REAL BUS ADDRESS
10396 040106 013720 001560          MOV      T.DCYL,(R0)+ ;STORE REAL CYLINDER ADDRESS
10397 040112 113710 001547          MOVB     T.DA+1,(R0) ;STORE REAL TRACK ADDRESS
10398 040116 005720          TST      (R0)+

```

10399 040120 113710 001546
10400 040124 104414
10401 040126 000204
10402
10403

MOV B T.DA,(R0) ;STORE REAL SECTOR ADDRESS
RESREG
RTS R4

10404
10405
10406
10407
10408
10409
10410
10411
10412
10413
10414
10415
10416

.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 (CHKOP). 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.

10417 040130 104413
10418 040132 011600
10419 040134 012037 001242
10420 040140 012037 001244
10421 040144 012037 001246
10422 040150 010016
10423 040152 012737 177777 001250
10424 040160 000403
10425

CHKWE: SAVREG
MOV (SP),R0 ;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 R0,(SP) ;STORE RETURN
MOV #-1,\$TMP13 ;SET FLAG - EXPECTED ERROR
BR CHKST

10426 040162 104413
10427 040164 005037 001250
10428

CHKOP: SAVREG
CLR \$TMP13 ;RESET EXPECTED ERROR FLAG

10429 040170 104420
10430 040172 005037 047774
10431 040176 005037 047776
10432 040202 005037 050000
10433 040206 005037 050002
10434 040212 005037 050116
10435 040216 032737 024000 001540
10436 040224 001111
10437 040226 032737 177400 001550
10438 040234 001105
10439 040236 032737 000070 001552
10440 040244 001101
10441 040246 005737 001554
10442 040252 001076
10443 040254 032737 100000 001540
10444 040262 001405
10445 040264 052737 100000 047774
10446 040272 000137 041022
10447

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
BIT #DSERBIT,T.DS ;TEST IF ERROR SET IN DS
BNE 4\$;YES - SKIP
TST T.ER ;TEST IF ERROR SET IN ER
BNE 4\$;YES - SKIP
BIT #CERR,T.CS1 ;COMBINED ERROR SET?
BEQ 9\$;NO - SKIP
BIS #CERNER,GRP1ER ;SET ERROR FLAG IN GROUP 1
JMP 25\$;SKIP

10448
10449
10450
10451
10452
10453
10454

: CODE TO CHECK WORD COUNT, BUFFER ADDRESS, CYLINDER, TRACK,
: AND SECTOR AT THE END OF THE OPERATION. THIS IS DONE ONLY
: IF CERR NOT SET BY THE OPERATION.

: ALL OF THE ABOVE CONDITIONS ARE CHECKED AND A BIT SET FOR
: EACH CHECK THAT FAILS. HOWEVER, ONLY ONE ERROR IS REPORTED.
: THE ORDER OF PRIORITY FOR REPORTING THE ERROR IS THE ORDER

```
10455 ; LISTED ABOVE.
10456
10457 040276 005737 001250 9$: TST $TMP13 ;TEST IF ERROR EXPECTED
10458 040302 001402 BEQ 62$ ;NO - SKIP
10459 040304 000137 041022 JMP 25$ ;YES - JUMP
10460 040310 013700 001540 62$: MOV T.CS1,R0 ;GET CS1
10461 040314 042700 177741 BIC #177741,R0 ;CHECK IF OPERATION IS READ DATA,
10462 040320 022700 000020 CMP #20,R0 ;WRITE DATA, OR WRITE CHECK. IF
10463 040324 002445 BLT 3$ ;NOT, SKIP ALL CHECKING IN GROUP
10464 040326 022700 000030 CMP #30,R0 ;FOUR
10465 040332 003042 BGT 3$
10466 040334 004437 037376 JSR R4,CHKWC ;CHECK WORD COUNT
10467 040340 000000 .WORD 0 ;EXPECTED WORD COUNT
10468 040342 004437 037430 JSR R4,CHKBA ;CHECK BUS ADDRESS
10469 040346 004437 037642 JSR R4,CHKCTS ;CHECK CYL, TRACK, & SECTOR
10470 040352 005737 050002 TST GRP4ER ;ANY GROUP 4 ERRORS?
10471 040356 001430 BEQ 3$ ;NO - SKIP
10472 040360 016037 050120 061002 MOV CMNDLB(R0),DF010A ;LOAD ADDRESS OF COMMAND MESSAGE
10473 040366 013700 050002 MOV GRP4ER,R0 ;PUT GROUP 4 ERROR FLAG IN R0
10474 040372 005001 CLR R1 ;CLEAR R1 FOR INDEX COUNTER
10475 040374 006200 1$: ASR R0 ;SHIFT FLAGS - FIRST ONE ON RIGHT IS ERROR TO
10476 040376 103402 BCS 2$ ;BE REPORTED, REST ARE IGNORED
10477 040400 005720 TST (R0)+ ;WHEN AN ERROR BIT IS FOUND,
10478 040402 000774 BR 1$ ;GET THE ERROR MESSAGE ASSOCIATED
10479 040404 016037 050004 001450 2$: MOV GRP4MS(R0),EM10N ;WITH IT AND SET ERROR TABLE ITEM TO
10480 040412 016037 047744 001202 MOV EXPWC(R0),$REG10 ;POINT TO THE MESSAGE. LOAD REG10 & 11
10481 040420 016037 047760 001204 MOV REALWC(R0),$REG11 ;WITH EXPECTED & IS VALUES
10482 040426 104414 RESREG ;RESTORE REGISTER
10483 040430 012776 000010 000000 MOV #10,@(SP) ;MAKE THE ERROR CALL POINT TO THE
10484 040436 000002 RTI ;RIGHT TABLE ENTRY & RETURN.
10485 040440 3$:
10486 040440 104414 RESREG
10487 040442 062716 000002 ADD #2,(SP) ;BUMP RETURN PAST ERROR
10488 040446 000002 RTI ;RETURN
10489
10490 ; THE FOLLOWING CODE BUILDS THE GROUP 1,2, & 3 ERROR WORDS.
10491
10492 040450 012700 047774 4$: MOV #GRP1ER,R0 ;SET UP GENERAL REGISTER AS POINTER
10493 040454 012701 001540 MOV #T.CS1,R1 ;CS1
10494 040460 012703 001550 MOV #T.CS2,R3 ;CS2
10495 040464 012704 001552 MOV #T.DS,R4 ;DS
10496 040470 012705 001554 MOV #T.ER,R5 ;AND ER
10497
10498 040474 051510 BIS (R5),(R0) ;SET ALL BITS IN GRP1ER THAT
10499 ;CORRESPOND TO ERROR BITS IN R5ER
10500 040476 042710 120701 BIC #ILF!ECH!BSE!HVR!OPI!DCK,(R0) ; CLEAR ALL THAT DON'T BELONG GRP1
10501
10502 040502 032711 020000 BIT #SPAR,(R1) ;TEST IF SPAR SET
10503 040506 001402 BEQ 5$ ;NO - SKIP
10504 040510 052710 000001 BIS #SPARERR,(R0) ;SET SPAR ERROR FLAG
10505
10506 040514 032714 000010 5$: BIT #ACLO,(R4) ;TEST ACLO SET
10507 040520 001402 BEQ 6$ ;NO - SKIP
10508 040522 052710 000100 BIS #ACLOERR,(R0) ;SET ACLO ERROR FLAG
10509
10510 040526 032714 000020 6$: BIT #SPDLSS,(R4) ;TEST SPEED LOSS SET
```


10511	040532	001402		BEQ	7\$;NO - SKIP
10512	040534	052710	000200	BIS	#SPDERR,(R0)	;SET SPEED LOSS ERROR FLAG
10513						
10514	040540	032714	000040	7\$: BIT	#DROT,(R4)	;TEST IF DROT SET
10515	040544	001402		BEQ	8\$;NO - SKIP
10516	040546	052710	000400	BIS	#DROTERR,(R0)	;SET DROT ERROR FLAG
10517						
10518	040552	032711	100000	8\$: BIT	#CERR,(R1)	;TEST CERR ITSELF SET
10519	040556	001002		BNE	10\$;YES - SKIP
10520	040560	032710	020000	BIT	#NCERWE,(R0)	;SET NO CERR WITH ERROR FLAG
10521						
10522	040564	012700	047776	10\$: MOV	#GRP2ER,R0	;SET POINTER TO GROUP 2 ERROR FLAGS
10523						
10524	040570	032715	000100	BIT	#ECH,(R5)	;TEST IF ECH SET
10525	040574	001402		BEQ	11\$;NO - SKIP
10526	040576	052710	000001	BIS	#ECHERR,(R0)	;SET ECH FLAG
10527						
10528	040602	032715	100000	11\$: BIT	#DCK,(R5)	;TEST DCK SET
10529	040606	001402		BEQ	12\$;NO - SKIP
10530	040610	052710	000002	BIS	#DCKERR,(R0)	;SET DCK ERROR FLAG.
10531						
10532	040614	032713	040000	12\$: BIT	#WCE,(R3)	;TEST WRITE CHECK ERROR
10533	040620	001402		BEQ	120\$;NO - SKIP
10534	040622	052710	000004	BIS	#WCKERR,(R0)	;SET WCE BIT
10535	040626	032713	100000	120\$: BIT	#DLT,(R3)	;TEST DATA LATE
10536	040632	001402		BEQ	13\$;NO - SKIP
10537	040634	052710	000010	BIS	#DLTERR,(R0)	;SET DLT ERROR FLAG
10538						
10539	040640	032715	020000	13\$: BIT	#OPI,(R5)	;TEST OPI SET
10540	040644	001402		BEQ	14\$;NO - SKIP
10541	040646	052710	000020	BIS	#OPIERR,(R0)	;SET OPI ERROR FLAG
10542						
10543	040652	032715	000400	14\$: BIT	#HVRC,(R5)	;TEST HVRC SET
10544	040656	001402		BEQ	16\$;NO - SKIP
10545	040660	052710	000040	BIS	#HVRCERR,(R0)	;SET HVRC FLAG
10546						
10547	040664	032715	000200	16\$: BIT	#BSE,(R5)	;TEST BSE ERROR FLAG
10548	040670	001402		BEQ	17\$;NO - SKIP
10549	040672	052710	000100	BIS	#BSERR,(R0)	;SET BSE FLAG
10550						
10551	040676	012700	050000	17\$: MOV	#GRP3ER,R0	;SET POINTER TO GROUP 3 FLAGS
10552						
10553	040702	032713	010000	BIT	#NED,(R3)	;TEST NED SET
10554	040706	001402		BEQ	18\$;NO - SKIP
10555	040710	052710	000001	BIS	#NEDERR,(R0)	;SET NED FLAG
10556						
10557	040714	032711	004000	18\$: BIT	#CTO,(R1)	;TEST CTO SET
10558	040720	001402		BEQ	19\$;NO - SKIP
10559	040722	052710	000002	BIS	#CTOERR,(R0)	;SET CTO FLAG
10560						
10561	040726	032713	000400	19\$: BIT	#UFE,(R3)	;TEST UFE SET
10562	040732	001402		BEQ	20\$;NO - SKIP
10563	040734	052710	000004	BIS	#UFERR,(R0)	;SET UFE FLAG
10564						
10565	040740	032713	001000	20\$: BIT	#MDS,(R3)	;TEST MDS SET
10566	040744	001402		BEQ	21\$;NO - SKIP

```
10567 040746 052710 000010      BIS      #MDSERR,(R0)      ;SET MDE FLAG
10568
10569 040752 032713 002000      21$:    BIT      #PGE,(R3)      ;TEST PGE SET
10570 040756 001402                BEQ      22$              ;NO - SKIP
10571 040760 052710 000020      BIS      #PGERR,(R0)     ;SET PGE FLAG
10572
10573 040764 032713 004000      22$:    BIT      #NEM,(R3)     ;TEST NEM SET
10574 040770 001402                BEQ      23$              ;NO - SKIP
10575 040772 052710 000040      BIS      #NEMERR,(R0)    ;SET NEM FLAG
10576
10577 040776 032713 020000      23$:    BIT      #UPE,(R3)     ;TEST UPE SET
10578 041002 001402                BEQ      24$              ;NO - SKIP
10579 041004 052710 000100      BIS      #UPERR,(R0)     ;SET UPE FLAG
10580
10581 041010 032715 000001      24$:    BIT      #ILF,(R5)     ;TEST ILF SET
10582 041014 001402                BEQ      25$              ;NO - SKIP
10583 041016 052710 000200      BIS      #ILFERR,(R0)    ;SET ILF FLAG.
```

```
10584
10585      ;      THE FOLLOWING CODE IS EXECUTED ONLY IF ERRORS WERE EXPECTED.
10586      ;      THE FLAG IN $TMP13 INDICATES IF
10587      ;      AN ERROR WAS EXPECTED AND THE CONTENTS OF TMP10,
10588      ;      TEMP11, & TEMP12 SPECIFY WHICH ERRORS. THESE ARE COMPARED AGAINST
10589      ;      THE ERRORS FOUND AND STORED IN GRP1ER, GRP2ER, AND GRP3ER.
10590      ;      THE CONTENTS OF GRP1,2, & 3 ARE MODIFIED TO INDICATE ERRORS THAT
10591      ;      OCCURRED BUT WERE NOT EXPECTED. THE CONTENTS OF $TMP10,11,
10592      ;      & 12 ARE MODIFIED TO INDICATE EXPECTED ERRORS THAT DID NOT
10593      ;      OCCUR. BOTH CONDITIONS CAN EXIST AT THE SAME TIME AND MUST
10594      ;      BE REPORTED.
```

```
10595
10596 041022 005737 001250      25$:    TST      $TMP13          ;CHECK IF AN ERROR WAS EXPECTED
10597 041026 001423                BEQ      110$             ;NO - SKIP
10598 041030 012704 047774      MOV      #GRP1ER,R4       ;GET ADDRESS OF ERROR
10599 041034 012705 001242      MOV      #$TMP10,R5      ;GET ADDRESS OF EXPECTED ERRORS
10600
10601 041040 011500      26$:    MOV      (R5),R0       ;GET EXPECTED ERROR
10602 041042 011401      MOV      (R4),R1         ;GET GROUP ERROR FLAGS
10603 041044 020001      CMP      R0,R1            ;ARE THEY EQUAL?
10604 041046 001003      BNE      27$              ;NO - SKIP
10605 041050 005000      CLR      R0               ;CLEAR EXPECTED ED
10606 041052 005001      CLR      R1               ;CLEAR OCCURED
10607 041054 000403      BR       28$
10608
10609 041056 010003      27$:    MOV      R0,R3         ;STORE EXPECTED ERRORS
10610 041060 040100      BIC      R1,R0            ;RESET EXPECTED THAT OCCURRED
10611 041062 040301      BIC      R3,R1            ;RESET OCCURRED THAT EXPECTED
10612 041064 010025      28$:    MOV      R0,(R5)+       ;STORE EXPECTED THAT DID NOT OCCUR
10613 041066 010124      MOV      R1,(R4)+       ;STORE OCCURRED THAT WERE NOT EXPECTED
10614 041070 022705 001250      CMP      #$TMP13,R5      ;ALL GROUPS CHECKED.
10615 041074 001361      BNE      26$              ;NO - LOOP
```

```
10616
10617      ;      THE FOLLOWING CODE:
10618      ;
10619      ;      A. DETERMINES WHICH FORMAT IS TO BE USED
10620      ;      B. LOADS THE ADDRESSES OF THE ASCIZ TEXT INTO THE SELECTED
10621      ;      ERROR TABLE ITEM AND FORMAT FIELD
10622      ;      C. COUNTS THE NUMBER OF ERRORS THAT MUST BE REPORTED
```

```
10623 : D. GETS DRIVE STATUS IF GROUP 1 ERROR.
10624 :
10625 : THE DECISION OF WHICH ERROR IS TO BE USED
10626 : IS BASED ON THE ERROR GROUP (OR GROUPS) THAT HAVE
10627 : FLAGS SET. IF ANY BIT IS SET IN GROUP 1,2, OR 3, GROUP 1 FORMAT (ERROR 4 OR 5)
10628 : WILL BE USED; ANY SET IN GROUP 2 OR 3, GROUP 2 (ERROR 6) WILL BE USED; AND A
10629 : FLAG SET IN GROUP 3 ONLY, GROUP 3 (ERROR 7) IS USED.
10630 :
10631 : THE FORMAT TO BE USED IN THE CONTROLLING FACTOR IN HOW THE
10632 : ERROR TRAP IS CHANGED IN THE MAIN CALL. IF GROUP 1 FORMAT IS
10633 : USED THE ERROR TRAP WILL BE CHANGED TO ERROR 4 OR 5 (DEPENDING ON
10634 : AVAILABILITY OF DRIVE STATUS), GROUP 2 FORMAT WILL BE ERROR
10635 : 6, AND GROUP 3 WILL BE ERROR 7. ONLY THE LOW ORDER BYTE
10636 : OF THE ERROR TRAP WILL BE ALTERED. THE SP WILL BE
10637 : POINTING TO THE LOCATION THAT CONTAINS THE ERROR CALL TRAP.
10638 :
10639 : IF THE STATUS IS READ FROM THE DRIVE WITH NO PROBLEM,
10640 : ERROR 4 IS USED. IF ANY ERROR IS ENCOUNTERED READING
10641 : STATUS, ERROR 5 IS USED. ERROR 5 INCLUDES A WARNING MESSAGE.
10642 :
10643 041076 005004 110$: CLR R4 ;CLEAR COUNTERS
10644 041100 005005 CLR R5
10645 041102 012700 001224 MOV #STMP1,R0 ;LOAD POINTERS FOR TEMPORARY STORAGE OF ADDRESS
10646 041106 012701 001226 MOV #STMP2,R1 ;WHERE ASCIZ ADDRESSES GO
10647 041112 012703 050116 MOV #GPSUMF,R3 ;POINTERS TO GROUP SUMMARY FLAGS
10648 041116 012710 060762 MOV #DF007A,(R0) ;PRESET FOR GRP3 ERR MESSAGE BUILD
10649 041122 012711 001442 MOV #DH7N,(R1)
10650 041126 013746 050000 MOV GRP3ER,-(SP) ;GET GROUP 3 ERRORS, PUT ON STACK
10651 041132 004437 036242 JSR R4,BITCNT ;GO COUNT NUMBER AT ERRORS
10652 041136 005716 TST (SP) ;ANY ERRORS?
10653 041140 001403 BEQ 29$ ;NO - SKIP
10654 041142 061605 ADD (SP),R5 ;ADD IN ERROR TOTAL
10655 041144 052713 000004 BIS #GRP3ST,(R3) ;SET BIT TO INDICATE GROUP 3 ERROR
10656
10657 041150 005726 29$: TST (SP)+ ;CLEAR OFF STACK
10658 041152 005737 001250 TST STMP13 ;ERROR EXPECTED
10659 041156 001412 BEQ 31$ ;NO - SKIP
10660 041160 013746 001246 MOV STMP12,-(SP) ;PUT GROUP 3 NOT RECEIVED ERRORS ON STACK
10661 041164 004437 036242 JSR R4,BITCNT ;COUNT NUMBER OF ERRORS.
10662 041170 005716 TST (SP) ;WERE THERE ANY
10663 041172 001403 BEQ 30$ ;NO - SKIP
10664 041174 052713 000040 BIS #GP3NR,(R3) ;SET GROUP 3 NOT RECEIVED ERROR FLAG
10665 041200 061604 ADD (SP),R4 ;ADD COUNT TO TOTAL THESE
10666
10667 041202 005726 30$: TST (SP)+ ;CLEAR OFF STACK
10668 041204 013746 047776 31$: MOV GRP2ER,-(SP) ;GET GROUP 2 ERRORS FOR COUNTING
10669 041210 004437 036242 JSR R4,BITCNT ;COUNT BITS
10670 041214 005716 TST (SP) ;ANY SET?
10671 041216 001407 BEQ 32$ ;NO - SKIP
10672 041220 052713 000002 BIS #GRP2ST,(R3) ;SET FLAG FOR GROUP 2 ERRORS
10673 041224 061605 ADD (SP),R5 ;ADD INTO TOTAL
10674 041226 012710 060736 MOV #DF006A,(R0) ;STORE ADDRESS FOR BUILDING REPORT
10675 041232 012711 001432 MOV #DH6N,(R1)
10676
10677 041236 005726 32$: TST (SP)+ ;CLEAR OFF STACK
10678 041240 005737 001250 TST STMP13 ;ANY EXPECTED ERRORS
```

```
10679 041244 001416 BEQ 34$ ;NO - SKIP
10680 041246 013746 001244 MOV $TMP11,-(SP) ;GET GROUP 2 NOT RECEIVED ERRORS
10681 041252 004437 036242 JSR R4,BITCNT ;COUNT NUMBER OF BITS
10682 041256 005716 TST (SP) ;ANY SET?
10683 041260 001407 BEQ 33$ ;NO - SKIP
10684 041262 052713 000020 BIS #GP2NR,(R3) ;SET FLAG FOR GROUP 2 NOT RECEIVED
10685 041266 061604 ADD (SP),R4 ;ADD INTO TOTAL
10686 041270 012710 060736 MOV #DF006A,(R0) ;STORE ADDRESS FOR BUILDING REPORT
10687 041274 012711 001432 MOV #DH6N,(R1)
10688
10689 041300 005726 33$: TST (SP)+ ;CLEAR OFF STACK
10690 041302 013746 047774 34$: MOV GRP1R,-(SP) ;GET GROUP 1 ERROR FLAGS
10691 041306 004437 036242 JSR R4,BITCNT ;COUNT THE NUMBER OF BITS
10692 041312 005716 TST (SP) ;ANY SET?
10693 041314 001407 BEQ 35$ ;NO - SKIP
10694 041316 052713 000001 BIS #GRP1ST,(R3) ;SET FLAG FOR GROUP 1 ERRORS SET
10695 041322 061605 ADD (SP),R5 ;ADD INTO TOTAL
10696 041324 012710 060652 MOV #DF004A,(R0) ;LOAD ADDRESS FOR BUILDING REPORT
10697 041330 012711 001412 MOV #DH4N,(R1)
10698 041334 005726 35$: TST (SP)+ ;CLEAR OFF STACK
10699 041336 005737 001250 TST $TMP13 ;ANY EXPECTED ERRORS?
10700 041342 001416 BEQ 60$ ;NO - SKIP
10701 041344 013746 001242 MOV $TMP10,-(SP) ;GET GROUP 1 NO RECEIVED ERROR
10702 041350 004437 036242 JSR R4,BITCNT ;COUNT # OF BITS
10703 041354 005716 TST (SP) ;ANY SET?
10704 041356 001407 BEQ 36$ ;NO - SKIP
10705 041360 052713 000010 BIS #GP1NR,(R3) ;SET FLAG FOR GROUP 1 NOT RECEIVED
10706 041364 061604 ADD (SP),R4 ;ADD INTO TOTAL
10707 041366 012710 060652 MOV #DF004A,(R0) ;LOAD ADDRESS FOR BUILDING REPORT
10708 041372 012711 001412 MOV #DH4N,(R1)
10709 041376 005726 36$: TST (SP)+ ;CLEAR OFF STACK.
10710 041400 032713 000011 60$: BIT #GRP1ST!GP1NR,(R3) ;ANY GROUP 1 ERROR
10711 041404 001414 BEQ 52$ ;NO - SKIP
10712 041406 042713 040000 BIC #DRSTER,(R3)
10713 041412 004437 036762 JSR R4,GETDRS
10714 041416 000401 BR 51$ ;ERROR RETURN
10715 041420 000406 BR 52$ ;NO ERROR RETURN
10716 041422 012710 060702 51$: MOV #DF005A,(R0) ;CHANGE TO FORMAT 5 - STORE ADDRESS
10717 041426 012711 001422 MOV #DH5N,(R1) ;FOR BUILDING REPORT
10718 041432 052713 040000 BIS #DRSTER,(R3) ;SET DRIVE STATUS ERROR
10719 041436 52$:
10720
10721
10722
10723
10724
10725
10726
10727
10728
10729
10730
10731
10732
10733
10734
```

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

```
10735 : $TMP11 CONTAINS GROUP 2 ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR
10736 : $TMP12 CONTAINS GROUP 3 ERRORS THAT WERE EXPECTED BUT DID NOT OCCUR
10737 : GRP1ER CONTAINS GROUP 1 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECT
10738 : GRP2ER CONTAINS GROUP 2 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECT
10739 : GRP3ER CONTAINS GROUP 3 ERRORS THAT OCCURRED OR OCCURRED AND WERE NOT EXPECT
10740 : (R1)=#$TMP2 CONTAINS THE ADDRESS OF THE HEADER MESSAGE ADDRESS IN
10741 : DF THAT MUST BE ALTERED TO IDENTIFY THE OPERATION
10742 : (R0)=#$TMP1 CONTAINS THE ADDRESS OF THE HEADER MESSAGE ADDRESS IN
10743 : THE ERROR ITEM TABLE THAT MUST BE ALTERED TO PROVIDE A
10744 : PROPER MESSAGE TO REPORT.
10745 : (R3)=#GRSUMF CONTAIN FLAGS TO INDICATE WHICH OF THE GROUP
10746 : ERROR FLAG FIELDS HAVE ERROR BITS STORED.
10747 041436 032777 020000 137474 BIT #SW13,@SWR ;IS REPORT INHIBITED?
10748 041444 001402 BEQ 37$ ;NO - SKIP
10749 041446 000137 042034 JMP 49$ ;ELSE EXIT
10750 041452 005737 001250 37$: TST $TMP13 ;WERE ERRORS EXPECTED?
10751 041456 001004 BNE 38$ ;YES - SKIP
10752 :
10753 : IF NO ERRORS WERE EXPECTED, $TMP10,11, &12 ARE NOT MEANINGFUL
10754 :
10755 041460 012771 057655 000000 MOV #DH007,@(R1) ;HEADER = ERROR IN OPERATION
10756 041466 000411 BR 39$
10757 041470 012771 057476 000000 38$: MOV #DH005,@(R1) ;PRESET HDRMSG = EXPECTED NOT SET
10758 041476 032713 000070 BIT #GP1NR!GP2NR!GP3NR,(R3) ;ANY NOT RECEIVED ERRORS?
10759 041502 001003 BNE 39$ ;YES - SKIP
10760 041504 012771 057573 000000 MOV #DH006,@(R1) ;SET MESSAGE TO UNEXPECTED ERROR SET
10761 041512 013701 001540 39$: MOV T.CS1,R1 ;GET CS1
10762 041516 042701 177741 BIC #177741,R1 ;CLEAR ALL BUT COMMAND
10763 :
10764 041522 016170 050120 000000 MOV CMNDLB(R1),@(R0) ;MOVE ADDRESS OF COMMAND MESSAGE
10765 : ;INTO REPORT
10766 041530 032713 000007 BIT #GRP1ST!GRP2ST!GRP3ST,(R3) ;ANY GPR ERRORS?
10767 041534 001462 BEQ 46$ ;NO - SKIP GPR REPORT
10768 :
10769 : PRINT ALL THE ERRORS CONTAINED IN THE GPR1,2,3ER(UNEXPECTED ERRORS)
10770 :
10771 041536 013701 050000 MOV GRP3ER,R1 ;GET GROUP 3 ERROR FLAGS
10772 041542 012700 050020 MOV #GRP3MS,R0 ;SET POINTER TO GRP3 ERROR MESSAGES
10773 041546 005037 001252 140$: CLR $TMP14 ;CLEAR GROUP PRINTING INDICATOR
10774 041552 012737 000021 001254 40$: MOV #17.,$TMP15 ;PRESET SHIFT COUNT
10775 041560 000241 CLC ;CLEAR CARRY
10776 041562 006001 41$: ROR R1 ;ROTATE ERROR FLAGS
10777 041564 103406 BCS 42$ ;WAS BIT SHIFTED OUT SET?
10778 041566 062700 000002 141$: ADD #2,R0 ;NO - BUMP POINTER
10779 041572 005337 001254 DEC $TMP15 ;DEC SHIFT COUNT
10780 041576 001371 BNE 41$ ;LOOP IF SHIFT NOT ZERO
10781 041600 000411 BR 44$
10782 041602 011037 041614 42$: MOV (R0),43$ ;GET ERROR MESSAGE ADDRESS FROM TABLE
10783 041606 104401 001273 TYPE ,SCLF ;TYPE CRLF
10784 041612 104401 TYPE ;TYPE ERROR MESSAGE
10785 041614 000000 43$: .WORD ;MESSAGE ADDRESS GOES HERE
10786 041616 005305 DEC R5 ;DECEREMENT TOTAL ERROR COUNT.
10787 041620 001362 BNE 141$ ;LOOP IF ZERO
10788 041622 000427 BR 46$ ;ELSE EXIT GPR ERROR PRINT LOOP
10789 :
10790 041624 005713 44$: TST (R3) ;TEST GPSUMF FLAG FOR PRINTING ERROR NOT RECEIVED
```

```
10791 041626 100455 BMI 47$ ;YES - SKIP
10792 041630 005737 001252 TST $TMP14 ;PRINTING GROUP 3?
10793 041634 001007 BNE 45$ ;NO -SKIP
10794 041636 013701 047776 MOV GRP2ER,R1 ;ELSE SET TO GROUP 2, GET GRP2ER
10795 041642 012700 050040 MOV #GRP2MS,RO ;& SET POINTER TO GROUP 2 ERROR MESSAGE TABLE
10796 041646 005237 001252 INC $TMP14 ;BUMP TO INDICATE PRINTING GROUP 2
10797 041652 000737 BR 40$ ;GO RESTART PRINT LOOP
10798 041654 022737 000002 001252 45$: CMP #2,$TMP14 ;PRINTING GROUP 1?
10799 041662 001407 BEQ 46$ ;YES - EXIT GPR ERROR PRINT LOOP.
10800 041664 013701 047774 MOV GRP1ER,R1 ;ELSE SET TO GROUP 1, GET GROUP 1 ERROR
10801 041670 012700 050056 MOV #GRP1MS,RO ;SET POINTER TO GROUP 1 ERROR MESSAGE TABLE
10802 041674 005237 001252 INC $TMP14 ;BUMP TO INDICATE PRINTING GROUP 1
10803 041700 000724 BR 40$ ;RESTART PRINT LOOP.
10804
10805 041702 005737 001250 46$: TST $TMP13 ;EXPECTING ERRORS?
10806 041706 001452 BEQ 49$ ;NO - SKIP
10807
10808 ; PRINT ALL ERRORS CONTAINED IN $TMP10, 11, 12(NOT RECEIVED ERRORS)
10809
10810 041710 005713 TST (R3) ;TEST IF PRINTING NOT RECEIVED ERRORS
10811 041712 100423 BMI 47$ ;YES - SKIP
10812 041714 032713 000070 BIT #GP1NR!GP2NR!GP3NR,(R3) ;ANY NOT RECEIVED ERRORS
10813 041720 001445 BEQ 49$ ;NO - SKIP
10814 041722 032713 000007 BIT #GRP1ST!GRP2ST!GRP3ST,(R3) ;ANY NOT RECEIVED ERRORS?
10815 041726 001404 BEQ 146$ ;NO - SKIP LABEL FOR UNEXPECTED ERRORS
10816 041730 104401 001273 TYPE ,$CRLF ;TYPE CRLF
10817 041734 104401 057573 TYPE ,DH006 ;TYPE HEADER FOR PREVIOUS ERRORS
10818 041740 052737 100000 050116 146$: BIS #REP NR,GPSUMF ;SET PRINTING NOT RECEIVED ERRORS SWITCH
10819 041746 010405 MOV R4,R5 ;MOVE TOTAL ERRORS TO R5
10820 041750 013701 001246 MOV $TMP12,R1 ;GET GRP3 NOT RECEIVED ERRORS
10821 041754 012700 050020 MOV #GRP3MS,RO ;SET POINTER TO GROUP 3 MESSAGES
10822 041760 000672 BR 140$ ;GO START PRINT LOOP
10823 041762 005737 001252 47$: TST $TMP14 ;PRINTING GROUP 3?
10824 041766 001007 BNE 48$ ;NO - SKIP
10825 041770 013701 001244 MOV $TMP11,R1 ;ELSE SETUP TO PRINT GROUP 2 - GET ERRORS
10826 041774 012700 050040 MOV #GRP2MS,RO ;& SET POINTER TO GROUP 2 MESSAGE TABLE
10827 042000 005237 001252 INC $TMP14 ;BUMP TO INDICATE GROUP 2 PRINTING
10828 042004 000662 BR 40$ ;GO START PRINT LOOP
10829 042006 022737 000002 001252 48$: CMP #2,$TMP14 ;PRINTING GROUP 1?
10830 042014 001407 BEQ 49$ ;YES - EXIT LOOP
10831 042016 013701 001242 MOV $TMP10,R1 ;SET POINTER TO GROUP 1 MESSAGE
10832 042022 012700 050056 MOV #GRP1MS,RO ;TABLE AND GET GROUP 1 ERRORS.
10833 042026 005237 001252 INC $TMP14 ;BUMP TO INDICATE GROUP 1 PRINTING
10834 042032 000647 BR 40$ ;START LOOP AGAIN.
10835
10836 042034 032713 000077 49$: BIT #77,(R3) ;TEST IF ANY ERRORS TO BE REPORTED
10837 ; GRP1ST!GRP2ST!GRP3ST
10838 ; GP1NR!GP2NR!GP3NR
10839 042040 001004 BNE 61$ ;YES - SKIP
10840 042042 104414 RESREG ;ELSE EXIT
10841 042044 062716 000002 ADD #2,(SP) ;BUMP FOR GOOD RETURN
10842 042050 000002 RTI
10843
10844 042052 112776 000007 000000 61$: MOVB #7,@(SP) ;PRESET FOR GROUP 3 ERROR RETURN.
10845 042060 032713 000022 BIT #GRP2ST!GP2NR,(R3) ;ANY GROUP 2 ERRORS?
10846 042064 001403 BEQ 50$ ;NO - SKIP
```

```

10847 042066 112776 000006 000000      MOVB    #6,(SP)          ;ELSE SET FOR GROUP 2 ERROR RETURN
10848
10849 042074 032713 000011      50$:    BIT    #GRP1ST!GP1NR,(R3) ;ANY GROUP 1 ERRORS?
10850 042100 001411      BEQ     53$             ;NO - SKIP
10851 042102 112776 000004 000000      MOVB    #4,(SP)          ;ELSE SET FOR GROUP 1 ERROR RETURN.
10852 042110 032713 040000      BIT     #DRSTER,(R3)    ;CHECK IF ERROR GETTING DRIVE STATUS
10853 042114 001403      BEQ     53$             ;NO - SKIP
10854 042116 112776 000005 000000      MOVB    #5,(SP)          ;ELSE CHANGE RETURN FORM GROUP 1
10855
10856 042124 005737 001264      53$:    TST     $ESCAPE      ;CHECK IF ESCAPE ALREADY SET
10857 042130 001011      BNE     54$             ;YES - SKIP
10858 042132 013700 001302      MOV     $TESTN,RO      ;SET UP $ESCAPE TO FORCE
10859 042136 006300      ASL     RO              ;ABORT TO PRESENT TEST AFTER
10860 042140 016037 033122 001264      MOV     $$W08TB(RO),$ESCAPE ;ERROR IS REPORTED
10861 042146 162737 000002 001264      SUB     #2,$ESCAPE     ;BUT GO TO NEXT SCOPE STATEMENT
10862 042154 104414      54$:    RESREG
10863 042156 000002      RTI                    ;RETURN
10864
10865
10866
10867
10868
10869
10870
10871
10872
10873
10874
10875
10876
10877
10878
10879
10880 042160 012637 001236      BDSRCK: MOV     (SP)+,$TMP6      ;STORE OLD R4 CONTENTS
10881 042164 010437 001240      MOV     R4,$TMP7        ;GET RETURN ADDRESS
10882 042170 011404      MOV     (R4),R4         ;GET POINTER TO FIELD TO BE CHECKED
10883 042172 005037 001234      CLR     $TMP5           ;CLEAR A COUNTER
10884 042176 005714      1$:    TST     (R4)         ;TEST IF FIELD HAS NO (OR NO MORE) ENTRIES
10885 042200 100417      BMI     4$              ;YES - EXIT
10886 042202 023724 001614      CMP     L.DCYL,(R4)+    ;IS THIS ENTRY FOR THIS CYLINDER?
10887 042206 001012      BNE     3$              ;NO - SKIP
10888 042210 005204      INC     R4              ;BUMP TO TRACK
10889 042212 123714 001607      CMPB   L.DT,(R4)       ;IS ENTRY FOR THIS TRACK?
10890 042216 001005      BNE     2$              ;NO - SKIP
10891 042220 005046      CLR     -(SP)           ;CLEAR STACK LOCATION
10892 042222 114416      MOVB   -(R4),(SP)      ;PUT SECTOR NUMBER ON STACK
10893 042224 005237 001234      INC     $TMP5           ;BUMP COUNTER
10894 042230 000401      BR     3$              ;BRANCH
10895
10896 042232 005304      2$:    DEC     R4           ;DECREMENT POINTER TO WORD ALIGN
10897 042234 005724      3$:    TST     (R4)+       ;BUMP TO NEXT ENTRY
10898 042236 000757      BR     1$              ;TEST NEXT ENTRY
10899
10900 042240 013746 001234      4$:    MOV     $TMP5,-(SP)   ;PUT COUNT ON STACK
10901 042244 013746 001236      MOV     $TMP6,-(SP)   ;PUT OLD R4 CONTENTS BACK ON STACK
10902 042250 013704 001240      MOV     $TMP7,R4      ;SET UP RETURN
  
```

```

*****
.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>
*****
  
```

10903 042254 005724
10904 042256 000204
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
10939
10940
10941
10942
10943
10944
10945
10946
10947
10948
10949
10950
10951
10952
10953
10954
10955
10956
10957
10958

TST (R4)+ ;BUMP PAST PARAMETER
RTS R4 ;RETURN

.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).
* 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:

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 :
10989 :
10990 :
10991 :

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.

10992 042260
10993 042260 010046
10994 042262 010146
10995 042264 010346
10996 042266 010546
10997 042270 012400
10998 042272 012737 056222 001520
10999 042300 032700 000200
11000 042304 001005
11001 042306 012737 056251 001520
11002 042314 000137 043046
11003 042320
11004 042320 010446
11005 042322 032700 001400
11006 042326 001002
11007 042330 000137 042630
11008 042334 113701 001607
11009 042340 013703 001614
11010 042344 012705 000005
11011
11012 042350 006301
11013 042352 005305
11014 042354 001375

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
BIT #BIT7,R0 ;;HEADER OPERATION SPECIFIED?
BNE 18\$;;YES - SKIP
MOV #EM55,EM15N ;;CHANGE FOR DATA COMPARE ERROR
JMP 17\$;;ELSE JUMP TO DATA ROUTINE
18\$: MOV R4,-(SP) ;;PUSH R4 ON STACK
BIT #BIT8!BIT9,R0 ;;MUST HEADERS BE BUILT?
BNE 19\$;;YES - SKIP
JMP 11\$;;ELSE JUMP TO HEADER COMPARE
19\$: MOV L.DT,R1 ;;START HEADER BUILD ROUTINE
MOV L.DCYL,R3 ;;GET TRACK AND CYL
MOV #5,R5 ;;SET COUNT TO SHIFT TRACK FOR HDR WORD
1\$: ASL R1 ;;SHIFT TRACK
DEC R5 ;;DECREMENT TRACK
BNE 1\$;;LOOP UNTIL COUNT 0

```
11015  
11016 042356 012704 000026      MOV #26,R4      ;PRESET FOR 26 SECTOR MODE  
11017 042362 032737 010000 001600  BIT #CFMT,L.CS1 ;IS IT 24 SECTOR MODE?  
11018 042370 001404          BEQ 2$         ;NO - SKIP  
11019 042372 012704 000024      MOV #24,R4      ;CHANGE COUNT FOR 24 SECTOR MODE  
11020 042376 052701 001000      BIS #BIT9,R1    ;SET 24 SECTOR MODE BIT IN WRD 2 OF HDR  
11021  
11022 042402 052701 140000      2$: BIS #BIT15!BIT14,R1 ;SET BS BITS TO INDICATE GOOD SECTOR  
11023 042406 012705 072414      MOV #OBUFF,R5  ;SET POINTER TO ADDRESS WHERE HEADERS GO  
11024 042412 010325      3$: MOV R3,(R5)+ ;INSERT CYLINDER  
11025 042414 010125      MOV R1,(R5)+  ;INSERT TRACK AND SECTOR  
11026 042416 010337 001224      MOV R3,$TMP1  ;CALCULATE HVRC WORD  
11027 042422 010115      MOV R1,(R5)  
11028 042424 040137 001224      BIC R1,$TMP1  
11029 042430 040315      BIC R3,(R5)  
11030 042432 053725 001224      BIS $TMP1,(R5)+ ;COMPLETE HVRC WORD INSERTION  
11031  
11032 042436 005304          DEC R4         ;DECREMENT HEADER COUNT  
11033 042440 001402          BEQ 4$         ;DONE? - YES, SKIP  
11034 042442 005201          INC R1         ;BUMP SECTOR  
11035 042444 000762          BR 3$         ;LOOP  
11036  
11037 042446 032700 001000      4$: BIT #BIT9,R0 ;MUST HEADERS BE CORRECTED FOR TABLE ENTRIES?  
11038 042452 001003          BNE 5$         ;YES - SKIP  
11039 042454 005700      10$: TST R0      ;IS THIS A COMPARE OPERATION?  
11040 042456 100464          BMI 11$       ;YES-GO DO HDR COMPARE  
11041 042460 000534          BR 50$        ;ELSE GET OUT  
11042  
11043 042462 013737 001640 042520 5$: MOV BSF26P,6$ ;PRESET FOR BS FACTORY LIST  
11044 042470 012737 100000 001224  MOV #BIT15,$TMP1 ;SET BIT TO BE RESET IN BAD HEADER  
11045 042476 032737 010000 001600  BIT #CFMT,L.CS1 ;IS THIS 26 SECTOR MODE?  
11046 042504 001403          BEQ 8$         ;YES - SKIP  
11047 042506 013737 001636 042520  MOV BSF24P,6$  ;ELSE CHANGE FOR 24 SECTOR MODE  
11048  
11049 042514 004437 042160      8$: JSR R4,BDSRCK ;GO CHECK FOR BAD SECTOR THIS ADDRESS  
11050 042520 000000      6$: .WORD 0     ;POINTER TO FILE TO BE CHECKED GOES HERE  
11051 042522 012605      MOV (SP)+,R5  ;GET # OF BAD SECTORS THIS PACK ADDRESS  
11052 042524 001417          BEQ 9$         ;SKIP IF ZERO  
11053  
11054 042526 011601      7$: MOV (SP),R1  ;GET 1ST BAD SECTOR NUMBER  
11055 042530 006301          ASL R1        ;MULTIPLY SECTOR NUMBER BY 6 TO  
11056 042532 006301          ASL R1        ;LOCATE SECTOR TO BE MARKED BAD  
11057 042534 061601          ADD (SP),R1  
11058 042536 062601          ADD (SP)+,R1  
11059 042540 062701 000002          ADD #2,R1    ;ADD 2 FOR 2ND WORD THAT SECTOR  
11060 042544 043761 001224 072414  BIC $TMP1,OBUFF(R1) ; CLEAR BIT FOR BAD SECTOR IN HDR  
11061 042552 043761 001224 072416  BIC $TMP1,OBUFF+2(R1) ; CORRECT THE HVRC BIT  
11062 042560 005305          DEC R5        ;DECREMENT BAD SECTOR COUNT  
11063 042562 001361          BNE 7$        ;LOOP IF NOT ZERO  
11064  
11065 042564 032737 100000 001224 9$: BIT #BIT15,$TMP1 ;WERE WE DOING BS FACTORY LIST?  
11066 042572 001730          BEQ 10$       ;NO - GO CHECK IF COMPARE MUST BE DONE  
11067 042574 012737 040000 001224  MOV #BIT14,$TMP1 ;ELSE SET BIT TO BE RESET IN BAD HDR  
11068 042602 013737 001644 042520  MOV BSS26P,6$  ;PRESET POINTER FOR 26 SECTOR MODE  
11069 042610 032737 010000 001600  BIT #CFMT,L.CS1 ;TEST IF WE ARE DOING 26 SECTOR MODE  
11070 042616 001736          BEQ 8$         ;YES - SKIP TO START CHECK
```

```

11071 042620 013737 001642 042520      MOV      BSS24P,6$      ;CHANGE POINTER TOR 24 SECTOR MODE
11072 042626 000732                BR        8$           ;SKIP TO START CHECK
11073
11074                ; START OF COMPARE
11075 042630 012701 000102 001600 11$:  MOV      #102,R1      ;PRESET FOR 102 WORDS OF HEADER
11076 042634 032737 010000                BIT      #CFMT,L.CS1  ;CHECK IF 26 SECTOR MODE
11077 042642 001402                BEQ      12$          ;YES - SKIP
11078 042644 012701 000074                MOV      #74,R1      ;CHANGE TO 74 WORDS OF HEADER
11079
11080 042650 012704 070414                12$:  MOV      #IBUFF,R4   ;SET START OF HEADERS TO BE COMPARED
11081 042654 012705 072414                MOV      #OBUFF,R5   ;SET START OF GOOD HEADERS
11082 042660 005003                CLR      R3          ;CLEAR COUNTER
11083 042662 032700 040000                BIT      #BIT14,R0   ;IS THIS A CONTINUATION OF EARLIER COMPARE
11084 042666 001412                BEQ      13$          ;NO - SKIP
11085 042670 013705 001700                28$:  MOV      DESHLD,R5   ;GET VALUES WHERE PREVIOUS CHECK STOPPED
11086 042674 013704 001702                MOV      SRCHLD,R4   ; DESTINATION AND SOURCE
11087 042700 013703 001704                MOV      WRDNUM,R3   ; WORD NUMBER IN ERROR
11088 042704 013701 001706                MOV      WRDCNT,R1   ; WORD COUNT LEFT IN COMPARE
11089 042710 005701                TST     R1           ;TEST IF WORD COUNT LEFT = 0
11090 042712 001417                BEQ      50$          ;YES - EXIT
11091
11092 042714 032700 030000                13$:  BIT      #BIT12!BIT13,R0 ;MEM MANAGE REQUIRED?
11093 042720 001402                BEQ      25$          ;NO - SKIP
11094 042722 005237 177572                INC     @#SRO        ;TURN IT ON
11095 042726 022425                25$:  CMP      (R4)+,(R5)+ ;COMPARE THE WORDS
11096 042730 001012                BNE     14$          ;SKIP IF NOT EQUAL
11097 042732 005203                INC     R3           ;BUMP WORD NUMBER IN ERROR
11098 042734 005301                DEC     R1           ;DEC WORD COUNT LEFT IN COMPARE
11099 042736 001373                BNE     25$          ;LOOP IF NOT ZERO
11100 042740 032700 030000                BIT      #BIT12!BIT13,R0 ;MEM MANAGE IN USE?
11101 042744 001402                BEQ      50$          ;NO - SKIP
11102 042746 005337 177572                DEC     @#SRO        ;TURN IT OFF
11103 042752                50$:
11104 042752 012604                MOV     (SP)+,R4     ;;POP STACK INTO R4
11105 042754 000427                BR      16$
11106
11107                ; ERROR REPORT PREP AND PARAMETER STORAGE FOR CONTINUATION
11108
11109 042756 010537 001700                14$:  MOV     R5,DESHLD    ;STORE DESTINATION
11110 042762 010437 001702                MOV     R4,SRCHLD   ; SOURCE
11111 042766 014537 001202                MOV     -(R5),SREG10 ;LOAD GOOD WORD FOR REPORT
11112 042772 014437 001204                MOV     -(R4),SREG11 ; BAD WORD
11113 042776 010337 001206                MOV     R3,SREG12   ; WORD NUMBER
11114 043002 005301                DEC     R1           ;DEC COUNT LEFTFOR CONTINUATION
11115 043004 005203                INC     R3           ;BUMP BAD WORD NUMBER
11116 043006 010137 001706                MOV     R1,WRDCNT   ;STORE COUNT LEFT
11117 043012 010337 001704                MOV     R3,WRDNUM   ; WORD NUM IN ERROR
11118 043016 032700 030000                BIT     #BIT12!BIT13,R0 ;MEM MANAGE IS USE?
11119 043022 001402                BEQ     15$          ;NO - SKIP
11120 043024 005337 177572                DEC     @#SRO        ;TURN IT OFF
11121
11122 043030                15$:
11123 043030 012604                MOV     (SP)+,R4     ;;POP STACK INTO R4
11124 043032 005724                TST     (R4)+        ;ERROR RETURN
11125
11126 043034                16$:

```

```

11127 043034 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
11128 043036 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
11129 043040 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11130 043042 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
11131 043044 000204      RTS      R4
11132
11133      ;      DATA PATTERN PROCESSING ROUTINE
11134
11135 043046 032700 040000    17$:    BIT      #BIT14,R0      ;CONTINUE WITH COMPARE?
11136 043052 001402      BEQ      29$            ;NO - SKIP
11137 043054 010446      MOV      R4,-(SP)      ;STORE RETURN
11138 043056 000704      BR       28$            ;GO CONTINUE COMPARE
11139
11140 043060 012705 072414    29$:    MOV      #OBUFF,R5      ;GET DESTINATION
11141 043064 012703 070414      MOV      #IBUFF,R3      ;GET SOURCE
11142 043070 032700 030000      BIT      #BIT12!BIT13,R0 ;USE MEM MANAGE?
11143 043074 001412      BEQ      21$            ;NO - SKIP
11144
11145 043076 012437 172354      MOV      (R4)+,@#KIPAR6 ;LOAD PAR FOR RELOCATION
11146 043102 032700 010000      BIT      #BIT12,R0      ;RELOCATE SOURCE?
11147 043106 001403      BEQ      20$            ;NO - SKIP
11148 043110 012705 140070      MOV      #140070,R5     ;SET DESTINATION TO USE PAR6 + OFFSET
11149 043114 000402      BR       21$            ;SKIP
11150 043116 012703 140070    20$:    MOV      #140070,R3     ;SET SOURCE TO USE PAR6 + OFFSET
11151
11152 043122 012401    21$:    MOV      (R4)+,R1      ;STORE COUNT
11153 043124 010446      MOV      R4,-(SP)      ;STORE RETURN
11154 043126 010304      MOV      R3,R4         ;PUT IN IBUFF POINTER
11155 043130 005003      CLR      R3            ;CLEAR R3 FOR WORD NUMBER COUNTER
11156 043132 032700 000077      BIT      #77,R0        ;ANY DATA PATTERN SPECIFIED?
11157 043136 001666      BEQ      13$            ;NO - GO DO COMPARE
11158
11159      ;      START OF GENERATION ROUTINE
11160
11161 043140 010537 001700      MOV      R5,DESHLD     ;STORE PARAMETERS FOR COMPARE
11162 043144 010437 001702      MOV      R4,SRCHLD
11163 043150 010337 001704      MOV      R3,WRDNUM
11164 043154 010137 001706      MOV      R1,WRDCNT
11165
11166      ;      CODE TO GENERATE DATA PATTERN IN AREA POINTED TO BY R5.
11167      ;      MEMORY MANAGEMENT WILL BE TURNED ON BUT RELOCATION
11168      ;      WILL NOT OCCUR UNLESS REQUIRED BY SWITCHES
11169
11170 043160 032700 030000      BIT      #BIT12!BIT13,R0 ;MEMORY MANAGEMENT REQUIRED?
11171 043164 001402      BEQ      33$            ;NO - SKIP
11172 043166 005237 177572      INC      @#SRO         ;TURN IT ON
11173 043172 032700 002000    33$:    BIT      #BIT10,R0     ;GENERATE PATTERN IN I'UFF?
11174 043176 001403      BEQ      32$            ;NO - SKIP
11175 043200 010446      MOV      R4,-(SP)      ;ELSE SWAP R4 AND R5
11176 043202 010504      MOV      R5,R4
11177 043204 012605      MOV      (SP)+,R5
11178
11179 043206 122700 000001    32$:    CMPB    #1,R0          ;PATTERN 1 (ALL ZEROS)?
11180 043212 001004      BNE      55$            ;NO - SKIP
11181 043214 005025    30$:    CLR      (R5)+         ;CLEAR WORD IN BUFF
11182 043216 005301      DEC      R1            ;DEC WORD COUNT

```

11183	043220	001375		BNE	30\$;LOOP UNTIL WORD COUNT ZERO
11184	043222	000550		BR	22\$;EXIT BUILD
11185							
11186	043224	122700	000007	55\$:	CMPB	#7,R0	;PATTERN 7 (ALL ONES)?
11187	043230	001005			BNE	56\$;NO - SKIP
11188	043232	012725	177777	31\$:	MOV	#-1,(R5)+	;LOAD WORD IN BUFF
11189	043236	005301			DEC	R1	;DEC WORD COUNT
11190	043240	001374			BNE	31\$;LOOP UNTIL WORD COUNT ZERO
11191	043242	000540			BR	22\$;EXIT BUILD
11192							
11193	043244	122700	000002	56\$:	CMPB	#2,R0	;PATTERN 2 SET UP
11194	043250	001003			BNE	57\$	
11195	043252	012703	047144		MOV	#PAT02,R3	
11196	043256	000504			BR	70\$	
11197							
11198	043260	122700	000003	57\$:	CMPB	#3,R0	;PATTERN 3 SET UP
11199	043264	001003			BNE	58\$	
11200	043266	012703	047204		MOV	#PAT03,R3	
11201	043272	000476			BR	70\$	
11202							
11203	043274	122700	000004	58\$:	CMPB	#4,R0	;PATTERN 4 SET UP
11204	043300	001003			BNE	59\$	
11205	043302	012703	047244		MOV	#PAT04,R3	
11206	043306	000470			BR	70\$	
11207							
11208	043310	122700	000005	59\$:	CMPB	#5,R0	;PATTERN 5 SET UP
11209	043314	001003			BNE	60\$	
11210	043316	012703	047304		MOV	#PAT05,R3	
11211	043322	000462			BR	70\$	
11212							
11213	043324	122700	000006	60\$:	CMPB	#6,R0	;PATTERN 6 SET UP
11214	043330	001003			BNE	61\$	
11215	043332	012703	047344		MOV	#PAT06,R3	
11216	043336	000454			BR	70\$	
11217							
11218	043340	122700	000010	61\$:	CMPB	#10,R0	;PATTERN 10 SET UP
11219	043344	001003			BNE	62\$	
11220	043346	012703	047404		MOV	#PAT10,R3	
11221	043352	000446			BR	70\$	
11222							
11223	043354	122700	000011	62\$:	CMPB	#11,R0	;PATTERN 11 SET UP
11224	043360	001003			BNE	63\$	
11225	043362	012703	047444		MOV	#PAT11,R3	
11226	043366	000440			BR	70\$	
11227							
11228	043370	122700	000012	63\$:	CMPB	#12,R0	;PATTERN 12 SET UP
11229	043374	001003			BNE	64\$	
11230	043376	012703	047504		MOV	#PAT12,R3	
11231	043402	000432			BR	70\$	
11232							
11233	043404	122700	000013	64\$:	CMPB	#13,R0	;PATTERN 13 SET UP
11234	043410	001003			BNE	65\$	
11235	043412	012703	047544		MOV	#PAT13,R3	
11236	043416	000424			BR	70\$	
11237							
11238	043420	122700	000014	65\$:	CMPB	#14,R0	;PATTERN 14 SET UP

```
11239 043424 001003          BNE      66$
11240 043426 012703 047604    MOV      #PAT14,R3
11241 043432 000416          BR       70$
11242
11243 043434 122700 000015    66$:    CMPB     #15,R0          ;PATTERN 15 SET UP
11244 043440 001003          BNE      67$
11245 043442 012703 047644    MOV      #PAT15,R3
11246 043446 000410          BR       70$
11247
11248 043450 122700 000016    67$:    CMPB     #16,R0          ;PATTERN 16 SET UP
11249 043454 001003          BNE      68$
11250 043456 012703 047704    MOV      #PAT16,R3
11251 043462 000402          BR       70$
11252
11253 043464 012703 047704    68$:    MOV       #PAT16,R3        ;SET UP FOR 16
11254
11255 043470 032700 004000    70$:    BIT       #BIT11,R0       ;FIRST WORD REPEAT?
11256 043474 001020          BNE      73$              ;YES - SKIP
11257 043476 010446          MOV      R4,-(SP)         ;STORE R4
11258 043500 010046          MOV      R0,-(SP)         ;STORE R0
11259 043502 012700 000020    MOV      #16,R0          ;PRESET COUNT FOR PATTERN LENGTH
11260 043506 010504          MOV      R5,R4           ;STORE START OF BUFF
11261
11262 043510 012325    71$:    MOV      (R3)+,(R5)+      ;MOV WORD TO BUFF
11263 043512 005301          DEC      R1               ;DEC WORD COUNT
11264 043514 001405          BEQ      74$              ;EXIT IF ZERO
11265 043516 005300          DEC      R0               ;DEC PAT LENGTH COUNT
11266 043520 001373          BNE      71$              ;LOOP IF NOT ZERO
11267
11268 043522 012425    72$:    MOV      (R4)+,(R5)+      ;REPEAT PATTERN IN BUFFER
11269 043524 005301          DEC      R1               ;DEC WORD COUNT
11270 043526 001375          BNE      72$              ;LOOP UNTIL WORD COUNT ZERO
11271
11272 043530 012600    74$:    MOV      (SP)+,R0         ;RESTORE R0
11273 043532 012604          MOV      (SP)+,R4         ;RESTORE R4
11274 043534 000403          BR       22$              ;EXIT BUILD
11275
11276 043536 011325    73$:    MOV      (R3),(R5)+      ;MOV THE SAME WORD INTO BUFFER
11277 043540 005301          DEC      R1               ;DEC WORD COUNT
11278 043542 001375          BNE      73$              ;LOOP UNTIL ZERO
11279
11280 043544 032700 030000    22$:    BIT      #BIT12!BIT13,R0 ;MEMORY MANAGEMENT REQUIRED?
11281 043550 001402          BEQ      34$              ;NO - SKIP
11282 043552 005337 177572    DEC      @#SRO            ;TURN OFF MEM MANAGEMENT
11283 043556 005700    34$:    TST      R0               ;IS COMPARE REQUIRED?
11284 043560 100012          BPL      23$              ;NO - SKIP
11285 043562 013705 001700    MOV      DESHLD,R5        ;RESTORE COMPARE PARAMETERS
11286 043566 013704 001702    MOV      SRCHLD,R4
11287 043572 013703 001704    MOV      WRDNUM,R3
11288 043576 013701 001706    MOV      WRDCNT,R1
11289 043602 000137 042714    JMP      13$              ;GO START COMPARE
11290
11291 043606 012604    23$:    MOV      (SP)+,R4        ;:POP STACK INTO R4
11292 043610 000137 043034    JMP      16$              ;GO TO EXIT
11293
11294 ;:*****
```

11295 .SBTTL PHASE LOCK LOOP CLOCK ADJUSTMENT ROUTINE
11296 :* THIS ROUTINE IS ENTERED VIA A START AT LOCATION 220(8). THE
11297 :* PROGRAM FIRST RUNS TEST 1, 2, AND 3 TO SET UP THE INTERNAL
11298 :* PROGRAM VARIABLES AND THEN JUMPS TO THE CLOCK ADJUST ROUTINE.
11299 :* THE ROUTINE SELECTS THE FIRST AVAILABLE DRIVE AND SETS AND
11300 :* RESETS DIAGNOSTIC MODE BIT IN MR1. INSTRUCTIONS ON WHERE TO
11301 :* SCOPE AND WHAT TO ADJUST ARE TYPED ON THE CONSOLE.
11302 :*
11303 :* THIS ROUTINE WILL LOOP UNTIL THE PROCESSOR IS HALTED.
11304 :*

```
11305 043614 104401 052662 ADJCLK: TYPE ,OPR019 ;TYPE ADJUSTMENT INSTRUCTIONS
11306
11307 043620 012762 000040 000010 MOV #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
11308 043626 013762 001626 000010 MOV DRVNUM,RKCS2(R2) ;SELECT DRIVE
11309 043634 012762 000001 000000 MOV #1,RKCS1(R2)
11310 043642 032762 000200 000000 5$: BIT #RDY,RKCS1(R2) ;WAIT FOR READY
11311 043650 001774 BEQ 5$
11312 043652 032737 100000 001664 BIT #LCLKPR,OPTFLG ;TEST IF CLOCK PRESENT
11313 043660 001402 BEQ 1$ ;NO - SKIP
11314 043662 005077 136022 CLR @KWLADD ;CLEAR INTERRUPT ENABLE
11315
11316 043666 012762 000040 000026 1$: MOV #DMD,RKMR1(R2) ;SET DIAG MODE
11317 043674 012701 000014 MOV #12.,R1 ;SET A COUNT
11318 043700 005301 2$: DEC R1 ;DEC COUNT
11319 043702 001376 BNE 2$ ;LOOP UNTIL ZERO
11320 043704 012762 000000 000026 MOV #0,RKMR1(R2) ;CLEAR MR1
11321 043712 012701 000014 MOV #12.,R1 ;SET COUNT
11322 043716 005301 3$: DEC R1 ;DEC COUNT
11323 043720 001376 BNE 3$ ;LOOP UNTIL ZERO
11324 043722 000761 BR 1$ ;RESTART LOOP
11325
```

11326 .SBTTL CONTROLLED HALT SUBROUTINE
11327 :* THIS ROUTINE IS ENTERED WHEN A CONTROL C IS TYPED. THE
11328 :* SUBSYSTEM IS CLEARED, THE DRIVE IS RECALIBRATED, AND, IF
11329 :* NECESSARY, CERTAIN CYLINDERS ARE REFORMATED. THE REFORMATTING
11330 :* IS CONTROLLED BY THE LOCATION REFMT WHICH CONTAINS THE ADDRESS
11331 :* OF THE CYLINDER TO BE REFORMATTED.
11332 :*

```
11333 043724 012737 000111 001302 CTRHLT: MOV #STN,$TESTN ;SET UP FOR HALT FAIL
11334
11335 043732 104416 TSSINIT ;CLEAR SUBSYSTEM
11336 043734 104003 ERROR 3 ;BAD INIT ERROR
11337
11338 043736 113700 001102 MOVB $STNM,R0 ;GET CURRENT TEST NUMBER
11339 043742 042700 177400 BIC #177400,R0 ;CLEAR UNUSED BITS
11340 043746 022700 000003 CMP #3,R0 ;TEST IF TEST NUMBER 3
11341 043752 001464 BEQ PROGEND ;GO TO HALT PROG
11342 043754 004437 035674 JSR R4,LRLOAD ;LOAD 'L' REGS
11343 043760 000113 RECAL ;RECAL
11344 043762 000000 0 ;0 WORDS
11345 043764 000000 0 ;0 IS BUFF ADDRESS
11346 043766 000 .BYTE 0 ;SECTOR 0
11347 043767 000 .BYTE 0 ;TRACK 0
11348 043770 000000 0 ;CYLINDER 0
11349
11350 043772 104417 TLOADRK ;LOAD RK REGS
```

```

11351 043774 104423          TWAT16          ;WAIT FOR INTERRUPT
11352 043776 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
11353
11354 044000 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
11355 044002 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
11356
11357 044004 005037 001662      CLR INTSET       ;CLEAR INTERRUPT FLAG
11358 044010 104437          TWAT8S          ;WAIT FOR SECOND INTERRUPT
11359 044012 104002          ERROR 2
11360
11361 044014 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
11362 044016 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
11363
11364 044020 104416          TSSINIT        ;CLEAR SUBSYSTEM
11365 044022 104003          ERROR 3          ;BAD INIT ERROR
11366
11367 044024 005737 001676      TST REFMT       ;TEST IF REFORMAT REQUIRED
11368 044030 001435          BEQ PROGEND    ;NO - GO TO HALT
11369 044032 104401 053025      TYPE ,OPRO20   ;TYPE MESSAGE
11370
11371 044036 004437 035674      JSR R4,LRLOAD  ;LOAD 'L' REGS
11372 044042 000127          WRHEAD         ;WRHEAD
11373 044044 177676          -102          ;-102 WORDS
11374 044046 072414          OBUFF         ;OBUFF IS BUFF ADDRESS
11375 044050 000          .BYTE 0       ;SECTOR 0
11376 044051 000          .BYTE 0       ;TRACK 0
11377 044052 000312          312          ;CYLINDER 312
11378
11379 044054 005737 001676      TST REFMT       ;TEST IF CYL 0
11380 044060 100002          BPL 5$        ;NO - SKIP
11381 044062 005037 001614      CLR L.DCYL    ;ELSE LOAD FOR CYL 0
11382 044066 004437 042260      JSR R4,GENCOM ;GENERATE HEADERS
11383 044072 001200          1200
11384
11385 044074 104417          TLOADRK       ;LOAD RK REGS
11386 044076 104434          TWAT159      ;WAIT FOR INTERRUPT
11387 044100 104002          ERROR 2          ;TO SLOW/NOT COMPLETE ERROR
11388
11389 044102 104421          TCHKOP          ;CHECK OPERATION FOR ANY ERRORS
11390 044104 104004          ERROR 4 ;OR 5, 6, 7 ;REPORT ALL ERRORS
11391
11392 044106 122737 000002 001607  CMPB #2,L.DT   ;TEST IF TRACK 2 FORMATTED
11393 044114 001403          BEQ PROGEND    ;YES - SKIP
11394 044116 105237 001607      INCB L.DT     ;ELSE BUMP TRACK
11395 044122 000761          BR 5$         ;DO NEXT TRACK
11396
11397 044124 104401 053121          PROGEN: TYPE ,OPRO21 ;TYPE HALT MESSAGE
11398 044130 ^12706 001100      MOV #STACK,SP ;CLEAR STACK
11399 044134 105037 001103      CLR $ERFLG    ;CLEAR ERROR FLAG
11400 044140 005037 001264      CLR $ESCAPE   ;CLEAR ESCAPE
11401 044144 005737 000042      TST @#42     ;TEST IF MONITOR PRESENT
11402 044150 001404          BEQ 10$       ;NO - SKIP
11403 044152 005037 032020      CLR $EOPCT   ;SET FOR END OF PROGRAM
11404 044156 000137 031772      JMP $EOP     ;GO TO END OF PASS
11405
11406 044162 000000          10$: HALT    ;HALT PROGRAM

```



```
11407 044164 000137 001776          JMP      START1          ;GO TO RESTART IF CONTINUE
11408
11409
11410          .SBTTL  HALT FAIL ROUTINE
11411          ;*      THIS ROUTINE IS ENTERED IF A HARDWARE ERROR IS DETECTED WHEN
11412          ;*      THE CARTRIDGE IS BEING REFORMATTED PRIOR TO HALT.
11412 044170 000240          ABTFAIL:  NOP
11413 044172 104401 053162          TYPE    ,OPRO22        ;TYPE HALT FAIL MESSAGE
11414 044176 000137 044124          JMP      PROGEND        ;GO STOP PROGRAM
11415          .SBTTL  TYPE ROUTINE
11416
11417          ;*****
11418          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11419          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11420          ;*NOTE1:  $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11421          ;*NOTE2:  $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11422          ;*NOTE3:  $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11423          ;*
11424          ;*CALL:
11425          ;*1) USING A TRAP INSTRUCTION
11426          ;*      TYPE    ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11427          ;*OR
11428          ;*      TYPE    MESADR
11429          ;*
11430          ;*
11431
11432 044202 105737 001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
11433 044206 100002          BPL      1$                ;;BR IF YES
11434 044210 000000          HALT
11435 044212 000430          BR      3$                ;;HALT HERE IF NO TERMINAL
11436 044214 010046          1$:    MOV      RO,-(SP)        ;;LEAVE
11437 044216 017600 000002          MOV      @2(SP),RO          ;;SAVE RO
11438 044222 122737 000001 001316          CMPB    #APTENV,$ENV        ;;GET ADDRESS OF ASCIZ STRING
11439 044230 001011          BNE     62$                ;;RUNNING IN APT MODE
11440 044232 132737 000100 0C1317          BITB    #APTSPOOL,$ENVM     ;;NO,GO CHECK FOR APT CONSOLE
11441 044240 001405          BEQ     62$                ;;SPOOL MESSAGE TO APT
11442 044242 010037 044252          MOV     RO,61$             ;;NO,GO CHECK FOR CONSOLE
11443 044246 004737 033354          JSR     PC,$ATY3           ;;SETUP MESSAGE ADDRESS FOR APT
11444 044252 000000          61$:   .WORD    0              ;;SPOOL MESSAGE TO APT
11445 044254 132737 000040 001317          62$:   BITB    #APTCSUP,$ENVM  ;;MESSAGE ADDRESS
11446 044262 001003          BNE     60$                ;;APT CONSOLE SUPPRESSED
11447 044264 112046          2$:    MOVB    (RO)+,-(SP)     ;;YES,SKIP TYPE OUT
11448 044266 001005          BNE     4$                ;;PUSH CHARACTER TO BE TYPED ONTO STACK
11449 044270 005726          TST     (SP)+              ;;BR IF IT ISN'T THE TERMINATOR
11450 044272 012600          60$:   MOV     (SP)+,RO          ;;IF TERMINATOR POP IT OFF THE STACK
11451 044274 062716 000002          3$:    ADD     #2,(SP)         ;;RESTORE RO
11452 044300 000002          RTI
11453 044302 122716 000011          4$:    CMPB    #HT,(SP)        ;;ADJUST RETURN PC
11454 044306 001430          BEQ     8$                ;;RETURN
11455 044310 122716 000200          CMPB    #CRLF,(SP)         ;;BRANCH IF <HT>
11456 044314 001006          BNE     5$                ;;BRANCH IF NOT <CRLF>
11457 044316 005726          TST     (SP)+              ;;POP <CR><LF> EQUIV
11458 044320 104401          TYPE
11459 044322 001273          $CRLF
11460 044324 105037 044460          CLRB    $CHARCNT          ;;TYPE A CR AND LF
11461 044330 000755          BR      2$                ;;CLEAR CHARACTER COUNT
11462 044332 004737 044414          5$:    JSR     PC,$TYPEC        ;;GET NEXT CHARACTER
11462          ;;GO TYPE THIS CHARACTER
```

```
11463 044336 123726 001156      6$:  CMPB  $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
11464 044342 001350              BNE  2$                ;; IF NO GO GET NEXT CHAR.
11465 044344 013746 001154      MOV  $NULL,-(SP)       ;; GET # OF FILLER CHARS. NEEDED
11466                                ;; AND THE NULL CHAR.
11467 044350 105366 000001      7$:  DECB  1(SP)         ;; DOES A NULL NEED TO BE TYPED?
11468 044354 002770              BLT  6$                ;; BR IF NO--GO POP THE NULL OFF OF STACK
11469 044356 004737 044414      JSR  PC,$TYPEC         ;; GO TYPE A NULL
11470 044362 105337 044460      DECB $CHARCNT          ;; DO NOT COUNT AS A COUNT
11471 044366 000770              BR   7$                ;; LOOP
11472
11473                                ;HORIZONTAL TAB PROCESSOR
11474
11475 044370 112716 000040      8$:  MOVB  #' ,(SP)      ;; REPLACE TAB WITH SPACE
11476 044374 004737 044414      9$:  JSR  PC,$TYPEC         ;; TYPE A SPACE
11477 044400 132737 000007 044460 BITB  #7,$CHARCNT      ;; BRANCH IF NOT AT
11478 044406 001372              BNE  9$                ;; TAB STOP
11479 044410 005726              TST  (SP)+            ;; POP SPACE OFF STACK
11480 044412 000724              BR   2$                ;; GET NEXT CHARACTER
11481 044414 105777 134530      $TYPEC: TSTB @STPS      ;; WAIT UNTIL PRINTER IS READY
11482 044420 100375              BPL  $TYPEC
11483 044422 116677 000002 134522 MOVB  2(SP),@STPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
11484 044430 122766 000015 000002 CMPB  #CR,2(SP)        ;; IS CHARACTER A CARRIAGE RETURN?
11485 044436 001003              BNE  1$                ;; BRANCH IF NO
11486 044440 105037 044460      CLRB $CHARCNT         ;; YES--CLEAR CHARACTER COUNT
11487 044444 000406              BR   $TYPEX           ;; EXIT
11488 044446 122766 000012 000002 1$:  CMPB  #LF,2(SP)       ;; IS CHARACTER A LINE FEED?
11489 044454 001402              BEQ  $TYPEX           ;; BRANCH IF YES
11490 044456 105227              INCB (PC)+            ;; COUNT THE CHARACTER
11491 044460 000000      $CHARCNT: .WORD 0     ;; CHARACTER COUNT STORAGE
11492 044462 000207      $TYPEX: RTS  PC
11493
11494                                .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11495
11496                                ;*****
11497                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11498                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11499                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11500                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11501                                ;*REPLACED WITH SPACES.
11502                                ;*CALL:
11503                                ;*      MOV  NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
11504                                ;*      TYPDS                ;;GO TO THE ROUTINE
11505
11506                                $TYPDS:
11507                                MOV  R0,-(SP)      ;;PUSH R0 ON STACK
11508                                MOV  R1,-(SP)      ;;PUSH R1 ON STACK
11509                                MOV  R2,-(SP)      ;;PUSH R2 ON STACK
11510                                MOV  R3,-(SP)      ;;PUSH R3 ON STACK
11511                                MOV  R5,-(SP)      ;;PUSH R5 ON STACK
11512                                MOV  #20200,-(SP)   ;;SET BLANK SWITCH AND SIGN
11513                                MOV  20(SP),R5      ;;GET THE INPUT NUMBER
11514                                BPL  1$            ;;BR IF INPUT IS POS.
11515                                NEG  R5            ;;MAKE THE BINARY NUMBER POS.
11516                                MOVB #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
11517                                CLR  R0            ;;ZERO THE CONSTANTS INDEX
11518                                MOV  #$DBLK,R3      ;;SETUP THE OUTPUT POINTER
```

```
11519 044526 112723 000040          MOVB  #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
11520 044532 005002          CLR  R2              ;;CLEAR THE BCD NUMBER
11521 044534 016001 044670          MOV  $DTBL(R0),R1    ;;GET THE CONSTANT
11522 044540 160105          SUB  R1,R5           ;;FORM THIS BCD DIGIT
11523 044542 002402          BLT  4$              ;;BR IF DONE
11524 044544 005202          INC  R2              ;;INCREASE THE BCD DIGIT BY 1
11525 044546 000774          BR   3$
11526 044550 060105          4$: ADD  R1,R5           ;;ADD BACK THE CONSTANT
11527 044552 005702          TST  R2              ;;CHECK IF BCD DIGIT=0
11528 044554 001002          BNE  5$              ;;FALL THROUGH IF 0
11529 044556 105716          TSTB (SP)           ;;STILL DOING LEADING 0'S?
11530 044560 100407          BMI  7$              ;;BR IF YES
11531 044562 106316          5$: ASLB (SP)         ;;MSD?
11532 044564 103003          BCC  6$              ;;BR IF NO
11533 044566 116663 000001 177777 MOVB  1(SP),-1(R3)   ;;YES--SET THE SIGN
11534 044574 052702 000060          6$: BIS  #'0,R2      ;;MAKE THE BCD DIGIT ASCII
11535 044600 052702 000040          7$: BIS  #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11536 044604 110223          MOVB  R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11537 044606 005720          TST  (R0)+          ;;JUST INCREMENTING
11538 044610 020027 000010          CMP  R0,#10         ;;CHECK THE TABLE INDEX
11539 044614 002746          BLT  2$              ;;GO DO THE NEXT DIGIT
11540 044616 003002          BGT  8$              ;;GO TO EXIT
11541 044620 010502          MOV  R5,R2          ;;GET THE LSD
11542 044622 000764          BR   6$              ;;GO CHANGE TO ASCII
11543 044624 105726          8$: TSTB (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
11544 044626 100003          BPL  9$              ;;BR IF NO
11545 044630 116663 177777 177776 MOVB  -1(SP),-2(R3)  ;;YES--SET THE SIGN FOR TYPING
11546 044636 105013          9$: CLRB (R3)        ;;SET THE TERMINATOR
11547 044640 012605          MOV  (SP)+,R5       ;;POP STACK INTO R5
11548 044642 012603          MOV  (SP)+,R3       ;;POP STACK INTO R3
11549 044644 012602          MOV  (SP)+,R2       ;;POP STACK INTO R2
11550 044646 012601          MOV  (SP)+,R1       ;;POP STACK INTO R1
11551 044650 012600          MOV  (SP)+,R0       ;;POP STACK INTO R0
11552 044652 104401 044700          TYPE ,SDBLK        ;;NOW TYPE THE NUMBER
11553 044656 016666 000002 000004 MOV  2(SP),4(SP)    ;;ADJUST THE STACK
11554 044664 012616          MOV  (SP)+,(SP)
11555 044666 000002          RTI                  ;;RETURN TO USER
11556 044670 023420          $DTBL: 10000.
11557 044672 001750          1000.
11558 044674 000144          100.
11559 044676 000012          10.
11560 044700 000004          $DBLK: .BLKW 4
11561          .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
```

```
11562
11563 *****
11564 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11565 *OCTAL (ASCII) NUMBER AND TYPE IT.
11566 *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11567 *CALL:
11568 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11569 *      TYPOS    ;;CALL FOR TYPEOUT
11570 *      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11571 *      .BYTE   M              ;;M=1 OR 0
11572 *      ;;1=TYPE LEADING ZEROS
11573 *      ;;0=SUPPRESS LEADING ZEROS
11574 *
```

```

11575      ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11576      ;*$TYPOS OR $TYPOC
11577      ;*CALL:
11578      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11579      ;*      TYPON      ;;CALL FOR TYPEOUT
11580      ;*
11581      ;*$TYPOC----ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11582      ;*CALL:
11583      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11584      ;*      TYPOC      ;;CALL FOR TYPEOUT
11585
11586 044710 017646 000000      $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
11587 044714 116637 000001 045133 MOVVB     1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
11588 044722 112637 045135      MOVVB     (SP)+,$OMODE+1    ;;NUMBER OF DIGITS TO TYPE
11589 044726 062716 000002      ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
11590 044732 000406
11591 044734 112737 000001 045133 $TYPOC: MOVVB     #1,$OFILL      ;;SET THE ZERO FILL SWITCH
11592 044742 112737 000006 045135      MOVVB     #6,$OMODE+1    ;;SET FOR SIX(6) DIGITS
11593 044750 112737 000005 045132 $TYPON: MOVVB     #5,$OCNT      ;;SET THE ITERATION COUNT
11594 044756 010346      MOV      R3,-(SP)          ;;SAVE R3
11595 044760 010446      MOV      R4,-(SP)          ;;SAVE R4
11596 044762 010546      MOV      R5,-(SP)          ;;SAVE R5
11597 044764 113704 045135      MOVVB     $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
11598 044770 005404      NEG      R4
11599 044772 062704 000006      ADD      #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
11600 044776 110437 045134      MOVVB     R4,$OMODE      ;;SAVE IT FOR USE
11601 045002 113704 045133      MOVVB     $OFILL,R4      ;;GET THE ZERO FILL SWITCH
11602 045006 016605 000012      MOV      12(SP),R5        ;;PICKUP THE INPUT NUMBER
11603 045012 005003      CLR      R3              ;;CLEAR THE OUTPUT WORD
11604 045014 006105      1$: ROL     R5            ;;ROTATE MSB INTO 'C'
11605 045016 000404      BR      3$              ;;GO DO MSB
11606 045020 006105      2$: ROL     R5            ;;FORM THIS DIGIT
11607 045022 006105      ROL     R5
11608 045024 006105      ROL     R5
11609 045026 010503      MOV     R5,R3
11610 045030 006103      3$: ROL     R3            ;;GET LSB OF THIS DIGIT
11611 045032 105337 045134      DECB     $OMODE          ;;TYPE THIS DIGIT?
11612 045036 100016      BPL     7$              ;;BR IF NO
11613 045040 042703 177770      BIC     #177770,R3      ;;GET RID OF JUNK
11614 045044 001002      BNE     4$              ;;TEST FOR 0
11615 045046 005704      TST     R4              ;;SUPPRESS THIS 0?
11616 045050 001403      BEQ     5$              ;;BR IF YES
11617 045052 005204      4$: INC     R4            ;;DON'T SUPPRESS ANYMORE 0'S
11618 045054 052703 000060      BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11619 045060 052703 000040      5$: BIS     #' ,R3        ;;MAKE ASCII IF NOT ALREADY
11620 045064 110337 045130      MOVVB     R3,8$          ;;SAVE FOR TYPING
11621 045070 104401 045130      TYPE     ,8$            ;;GO TYPE THIS DIGIT
11622 045074 105337 045132      7$: DECB     $OCNT        ;;COUNT BY 1
11623 045100 003347      BGT     2$              ;;BR IF MORE TO DO
11624 045102 002402      BLT     6$              ;;BR IF DONE
11625 045104 005204      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11626 045106 000744      BR      2$              ;;GO DO THE LAST DIGIT
11627 045110 012605      6$: MOV     (SP)+,R5      ;;RESTORE R5
11628 045112 012604      MOV     (SP)+,R4      ;;RESTORE R4
11629 045114 012603      MOV     (SP)+,R3      ;;RESTORE R3
11630 045116 016666 000002 000004      MOV     2(SP),4(SP)    ;;SET THE STACK FOR RETURNING

```

```
11631 045124 012616          MOV      (SP)+,(SP)
11632 045126 000002          RTI              ;;RETURN
11633 045130 000           8$: .BYTE 0       ;;STORAGE FOR ASCII DIGIT
11634 045131 000           .BYTE 0       ;;TERMINATOR FOR TYPE ROUTINE
11635 045132 000           $OCNT: .BYTE 0  ;;OCTAL DIGIT COUNTER
11636 045133 000           $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
11637 045134 000000          $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
11638 .SBTTL TTY INPUT ROUTINE
11639
11640 ;;*****
11641 .ENABL LSB
11642 045136 000000          $TKCNT: .WORD 0  ;;NUMBER OF ITEMS IN QUEUE
11643 045140 000000          $TKQIN: .WORD 0  ;;INPUT POINTER
11644 045142 000000          $TKQOUT: .WORD 0 ;;OUTPUT POINTER
11645 045144 000001          $TKQSRT: .BLKB 1 ;;TTY KEYBOARD QUEUE
11646 045145
11647 045146
11648
11649 ;*TK INITIALIZE ROUTINE
11650 ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
11651 ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
11652
11653 ;*CALL:
11654 ;*      JSR      PC,$TKINT
11655 ;*      RETURN
11656
11657 045146 005037 045136          $TKINT: CLR      $TKCNT  ;;CLEAR COUNT OF ITEMS IN QUEUE
11658 045152 012737 045144 045140  MOV      #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
11659 045160 013737 045140 045142  MOV      $TKQIN,$TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
11660 045166 012737 045216 000060  MOV      #$TKSRV,@$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
11661 045174 012737 000200 000062  MOV      #200,@$TKVEC+2 ;;'BR' LEVEL 4
11662 045202 005777 133740          TST      @$TKB        ;;CLEAR DONE FLAG
11663 045206 012777 000100 133730  MOV      #100,@$TKS    ;;ENABLE TTY KEYBOARD INTERRUPT
11664 045214 000207          RTS      PC          ;;RETURN TO CALLER
11665
11666 ;*TK SERVICE ROUTINE
11667 ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
11668 ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
11669 ;*IT IN THE QUEUE.
11670 ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
11671 ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
11672
11673 045216 117746 133724          $TKSRV: MOVVB   @$TKB,-(SP) ;;PICKUP THE CHARACTER
11674 045222 042716 177600          BIC      #^C177,(SP)  ;;STRIP THE JUNK
11675 045226 021627 000003          CMP      (SP),#3     ;;IS IT A CONTROL C?
11676 045232 001007          BNE      1$         ;;BRANCH IF NO
11677 045234 104401 046332          TYPE    ,SCNTLC     ;;TYPE A CONTROL-C (^C)
11678 045240 004737 045146          JSR      PC,$TKINT  ;;INIT THE KEYBOARD
11679 045244 005726          TST      (SP)+      ;;CLEAN UP STACK
11680 045246 000137 043724          JMP      CTRHLT     ;;CONTROL C RESTART
11681 045252 021627 000007          1$: CMP      (SP),#7   ;;IS IT A CONTROL G?
11682 045256 001004          BNE      2$         ;;BRANCH IF NO
11683 045260 022737 000176 001140  CMP      #SWREG,SWR  ;;IS SOFT-SWR SELECTED?
11684 045266 001500          BEQ      6$         ;;GO TO SWR CHANGE
11685
11686 045270          2$:
```

11687	045270	022737	000001	045136	CMP	#1,\$TKCNT	:: IS THE QUEUE FULL?
11688	045276	001004			BNE	3\$:: BRANCH IF NO
11689	045300	104401	001266		TYPE	,\$BELL	:: RING THE TTY BELL
11690	045304	005726			TST	(SP)+	:: CLEAN CHARACTER OFF OF STACK
11691	045306	000451			BR	5\$:: EXIT
11692	045310	021627	000023	3\$:	CMP	(SP),#23	:: IS IT A CONTROL-S?
11693	045314	001021			BNE	32\$:: BRANCH IF NO
11694	045316	005077	133622		CLR	@\$TKS	:: DISABLE TTY KEYBOARD INTERRUPTS
11695	045322	005726			TST	(SP)+	:: CLEAN CHAR OFF STACK
11696	045324	105777	133614	31\$:	TSTB	@\$TKS	:: WAIT FOR A CHAR
11697	045330	100375			BPL	31\$:: LOOP UNTIL ITS THERE
11698	045332	117746	133610		MOVB	@\$TKB,-(SP)	:: GET THE CHARACTER
11699	045336	042716	177600		BIC	^C177,(SP)	:: MAKE IT 7-BIT ASCII
11700	045342	022627	000021		CMP	(SP)+,#21	:: IS IT A CONTROL-Q?
11701	045346	001366			BNE	31\$:: BRANCH IF NO
11702	045350	012777	000100	133566	MOV	#100,@\$TKS	:: REENABLE TTY KEYBOARD INTERRUPTS
11703	045356	000002			RTI		:: RETURN
11704	045360	005237	045136	32\$:	INC	\$TKCNT	:: COUNT THIS CHARACTER
11705	045364	021627	000140		CMP	(SP),#140	:: IS IT UPPER CASE?
11706	045370	002405			BLT	4\$:: BRANCH IF YES
11707	045372	021627	000175		CMP	(SP),#175	:: IS IT A SPECIAL CHAR?
11708	045376	003002			BGT	4\$:: BRANCH IF YES
11709	045400	042716	000040		BIC	#40,(SP)	:: MAKE IT UPPER CASE
11710	045404	112677	177530	4\$:	MOVB	(SP)+,@\$TKQIN	:: AND PUT IT IN QUEUE
11711	045410	005237	045140		INC	\$TKQIN	:: UPDATE THE POINTER
11712	045414	023727	045140	045145	CMP	\$TKQIN,\$\$TKQEND	:: GO OFF THE END?
11713	045422	001003			BNE	5\$:: BRANCH IF NO
11714	045424	012737	045144	045140	MOV	\$\$TKQSRT,\$\$TKQIN	:: RESET THE POINTER
11715	045432	000002			RTI		:: RETURN

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

11722	045434	022737	000176	001140	\$CKSWR: CMP	#SWREG,SWR	:: IS THE SOFT-SWR SELECTED
11723	045442	001124			BNE	15\$:: EXIT IF NOT
11724	045444	105777	133474		TSTB	@\$TKS	:: IS A CHAR WAITING?
11725	045450	100121			BPL	15\$:: IF NOT, EXIT
11726	045452	117746	133470		MOVB	@\$TKB,-(SP)	:: YES
11727	045456	042716	177600		BIC	^C177,(SP)	:: MAKE IT 7-BIT ASCII
11728	045462	021627	000007		CMP	(SP),#7	:: IS IT A CONTROL-G?
11729	045466	001300			BNE	2\$:: IF NOT, PUT IT IN THE TTY QUEUE
11730							:: AND EXIT

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

11736	045470	123727	001134	000001	6\$:	CMPB	\$AUTOB,#1	:: ARE WE RUNNING IN AUTO-MODE?
11737	045476	001674			BEQ	2\$:: BRANCH IF YES	
11738	045500	005726			TST	(SP)+	:: CLEAR CONTROL-G OFF STACK	
11739	045502	004737	045146		JSR	PC,\$TKINT	:: FLUSH THE TTY INPUT QUEUE	
11740	045506	005077	133432		CLR	@\$TKS	:: DISABLE TTY KEYBOARD INTERRUPTS	
11741	045512	112737	000001	001135	MOVB	#1,\$INTAG	:: SET INTERRUPT MODE INDICATOR	
11742								

Address	Hex	Hex	Hex	Label	Op	Op	Op	Op
11743	045520	104401	046344		TYPE	, \$CNTLG	:: ECHO THE CONTROL-G (^G)	
11744	045524	104401	046351		\$GTSWR: TYPE	, \$MSWR	:: TYPE CURRENT CONTENTS	
11745	045530	013746	000176		MOV	\$WREG, -(SP)	:: SAVE SWREG FOR TYPEOUT	
11746	045534	104402			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)	
11747	045536	104401	046362		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR	
11748	045542	005046		19\$:	CLR	-(SP)	:: CLEAR COUNTER	
11749	045544	005046			CLR	-(SP)	:: THE NEW SWR	
11750	045546	105777	133372	7\$:	TSTB	@ \$TKS	:: CHAR THERE?	
11751	045552	100375			BPL	7\$:: IF NOT TRY AGAIN	
11752								
11753	045554	117746	133366		MOVB	@ \$TKB, -(SP)	:: PICK UP CHAR	
11754	045560	042716	177600		BIC	#^C177, (SP)	:: MAKE IT 7-BIT ASCII	
11755								
11756	045564	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?	
11757	045570	001015			BNE	9\$:: BRANCH IF NOT	
11758	045572	104401	046332		TYPE	, \$CNTLC	:: YES, ECHO CONTROL-C (^C)	
11759	045576	062706	000006		ADD	#6, SP	:: CLEAN UP STACK	
11760	045602	123727	001135	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?	
11761	045610	001003			BNE	8\$:: BRANCH IF NC	
11762	045612	012777	000100	133324	MOV	#100, @ \$TKS	:: ALLOW TTY KEYBOARD INTERRUPTS	
11763	045620	000137	043724	8\$:	JMP	CTRHLT	:: CONTROL-C RESTART	
11764								
11765								
11766	045624	021627	000025	9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?	
11767	045630	001005			BNE	10\$:: BRANCH IF NOT	
11768	045632	104401	046337		TYPE	, \$CNTLU	:: YES, ECHO CONTROL-U (^U)	
11769	045636	062706	000006	20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT	
11770	045642	000737			BR	19\$:: LET'S TRY IT AGAIN	
11771								
11772								
11773	045644	021627	000015	10\$:	CMP	(SP), #15	:: IS IT A <CR>?	
11774	045650	001022			BNE	16\$:: BRANCH IF NO	
11775	045652	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?	
11776	045656	001403			BEQ	11\$:: BRANCH IF YES	
11777	045660	016677	000002	133252	MOV	2(SP), @ \$SWR	:: SAVE NEW SWR	
11778	045666	062706	000006	11\$:	ADD	#6, SP	:: CLEAN UP STACK	
11779	045672	104401	001273	14\$:	TYPE	, \$CRLF	:: ECHO <CR> AND <LF>	
11780	045676	123727	001135	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?	
11781	045704	001003			BNE	15\$:: BRANCH IF NOT	
11782	045706	012777	000100	133230	MOV	#100, @ \$TKS	:: RE-ENABLE TTY KBD INTERRUPTS	
11783	045714	000002		15\$:	RTI		:: RETURN	
11784	045716	004737	044414	16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR	
11785	045722	021627	000060		CMP	(SP), #60	:: CHAR < 0?	
11786	045726	002420			BLT	18\$:: BRANCH IF YES	
11787	045730	021627	000067		CMP	(SP), #67	:: CHAR > 7?	
11788	045734	003015			BGT	18\$:: BRANCH IF YES	
11789	045736	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII	
11790	045742	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR	
11791	045746	001403			BEQ	17\$:: BRANCH IF YES	
11792	045750	006316			ASL	(SP)	:: NO, SHIFT PRESENT	
11793	045752	006316			ASL	(SP)	:: CHAR OVER TO MAKE	
11794	045754	006316			ASL	(SP)	:: ROOM FOR NEW ONE.	
11795	045756	005266	000002	17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR	
11796	045762	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR	
11797	045766	000667			BR	7\$:: GET THE NEXT ONE	
11798	045770	104401	001272	18\$:	TYPE	, \$QUES	:: TYPE ?<CR><LF>	

```
11799 045774 000720          BR      20$          ;;SIMULATE CONTROL-U
11800          .DSABL  LSB
11801
11802
11803          ;*****
11804          ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11805          ;*CALL:
11806          ;*      RDCHR          ;;GET A CHARACTER FROM THE QUEUE
11807          ;*      RETURN HERE  ;;CHARACTER IS ON THE STACK
11808          ;*                          ;;WITH PARITY BIT STRIPPED OFF
11809          ;
11810
11811 045776 011646          $RDCHR: MOV      (SP),-(SP)      ;;PUSH DOWN THE PC AND
11812 046000 016666 000004 000002      MOV      4(SP),2(SP)      ;;THE PS
11813 046006 005066 000004          CLR      4(SP)          ;;GET READY FOR A CHARACTER
11814 046012 005046          CLR      -(SP)         ;;PUT NEW PS ON STACK
11815 046014 012746 046022      MOV      #64$,-(SP)     ;;PUT NEW PC ON STACK
11816 046020 000002          RTI                   ;;POP NEW PC AND PS
11817 046022          64$:
11818 046022 005737 045136      1$:      TST      $TKCNT      ;;WAIT ON A CHARACTER
11819 046026 001775          BEQ      1$
11820 046030 005337 045136      DEC      $TKCNT      ;;DECREMENT THE COUNTER
11821 046034 117766 177102 000004      MOV      @TKQOUT,4(SP)  ;;GET ONE CHARACTER
11822 046042 005237 045142      INC      $TKQOUT      ;;UPDATE THE POINTER
11823 046046 023727 045142 045145      CMP      $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
11824 046054 001003          BNE      2$          ;;BRANCH IF NO
11825 046056 012737 045144 045142      MOV      #$TKQSRT,$TKQOUT ;;RESET THE POINTER
11826 046064 000002          2$:      RTI                   ;;RETURN
11827          ;*****
11828          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
11829          ;*CALL:
11830          ;*      RDLIN          ;;INPUT A STRING FROM THE TTY
11831          ;*      RETURN HERE  ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
11832          ;*                          ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
11833          ;
11834 046066 010346          $RDLIN: MOV      R3,-(SP)      ;;SAVE R3
11835 046070 005046          CLR      -(SP)         ;;CLEAR THE RUBOUT KEY
11836 046072 012703 046322      1$:      MOV      #$TTYIN,R3      ;;GET ADDRESS
11837 046076 022703 046332      2$:      CMP      #$TTYIN+8.,R3    ;;BUFFER FULL?
11838 046102 101456          BLOS     4$            ;;BR IF YES
11839 046104 104410          RDCHR          ;;GO READ ONE CHARACTER FROM THE TTY
11840 046106 112613          MOV      (SP)+,(R3)     ;;GET CHARACTER
11841 046110 122713 000177      10$:     CMP      #177,(R3)      ;;IS IT A RUBOUT
11842 046114 001022          BNE      5$            ;;BR IF NO
11843 046116 005716          TST      (SP)          ;;IS THIS THE FIRST RUBOUT?
11844 046120 001007          BNE      6$            ;;BR IF NO
11845 046122 112737 000134 046320      MOV      #'\\,9$        ;;TYPE A BACK SLASH
11846 046130 104401 046320      TYPE     ,9$
11847 046134 012716 177777          MOV      #-1,(SP)      ;;SET THE RUBOUT KEY
11848 046140 005303          6$:      DEC      R3            ;;BACKUP BY ONE
11849 046142 020327 046322      CMP      R3,$$TTYIN    ;;STACK EMPTY?
11850 046146 103434          BLO     4$            ;;BR IF YES
11851 046150 111337 046320      MOV      (R3),9$       ;;SETUP TO TYPEOUT THE DELETED CHAR.
11852 046154 104401 046320      TYPE     ,9$
11853 046160 000746          BR      2$            ;;GO READ ANOTHER CHAR.
11854 046162 005716          5$:      TST      (SP)          ;;RUBOUT KEY SET?
```



```
11855 046164 001406          BEQ      7$          ;;BR IF NO
11856 046166 112737 000134 046320  MOVB    #' \,9$    ;;TYPE A BACK SLASH
11857 046174 104401 046320          TYPE    ,9$
11858 046200 005016          CLR     (SP)        ;;CLEAR THE RUBOUT KEY
11859 046202 122713 000025 7$:    CMPB    #25,(R3)   ;;IS CHARACTER A CTRL U?
11860 046206 001003          BNE     8$          ;;BR IF NO
11861 046210 104401 046337          TYPE    ,%CNTLU    ;;TYPE A CONTROL 'U'
11862 046214 000726          BR      1$          ;;GO START OVER
11863 046216 122713 000022 8$:    CMPB    #22,(R3)   ;;IS CHARACTER A 'R'?
11864 046222 001011          BNE     3$          ;;BRANCH IF NO
11865 046224 105013          CLRB   (R3)        ;;CLEAR THE CHARACTER
11866 046226 104401 001273          TYPE    ,%CRLF     ;;TYPE A 'CR' & 'LF'
11867 046232 104401 046322          TYPE    ,%TTYIN    ;;TYPE THE INPUT STRING
11868 046236 000717          BR      2$          ;;GO PICKUP ANOTHER CHACTER
11869 046240 104401 001272 4$:    TYPE    ,%QUES     ;;TYPE A '?'
11870 046244 000712          BR      1$          ;;CLEAR THE BUFFER AND LOOP
11871 046246 111337 046320 3$:    MOVB    (R3),9$    ;;ECHO THE CHARACTER
11872 046252 104401 046320          TYPE    ,9$
11873 046256 122723 000015          CMPB    #15,(R3)+  ;;CHECK FOR RETURN
11874 046262 001305          BNE     2$          ;;LOOP IF NOT RETURN
11875 046264 105063 177777          CLRB   -1(R3)     ;;CLEAR RETURN (THE 15)
11876 046270 104401 001274          TYPE    ,%LF       ;;TYPE A LINE FEED
11877 046274 005726          TST    (SP)+      ;;CLEAN RUBOUT KEY FROM THE STACK
11878 046276 012603          MOV     (SP)+,R3   ;;RESTORE R3
11879 046300 011646          MOV     (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
11880 046302 016666 000004 000002  MOV     4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
11881 046310 012766 046322 000004  MOV     #%TTYIN,4(SP)
11882 046316 000002          RTI
11883 046320 000          9$:    .BYTE  0          ;;STORAGE FOR ASCII CHAR. TO TYPE
11884 046321 000          .BYTE  0          ;;TERMINATOR
11885 046322 000010          $TTYIN: .BLKB 8.    ;;RESERVE 8 BYTES FOR TTY INPUT
11886 046332 041536 005015 000          %CNTLC: .ASCIZ /^C/<15><12> ;;CONTROL 'C'
11887 046337 0136 006525 000012 %CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
11888 046344 043536 005015 000          %CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
11889 046351 015 051412 051127 %MSWR: .ASCIZ <15><12>/SWR = /
11890 046356 036440 000040          $MNEW: .ASCIZ / NEW = /
11891 046362 020040 042516 020127
11892 046370 020075 000
11893 046374
11894 .EVEN
11895 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
11896
11897 ;;*****
11898 ;;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
11899 ;;*CHANGE IT TO BINARY.
11900 ;;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
11901 ;;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
11902 ;;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
11903 ;;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
11904 ;;*CALL:
11905 ;;* RDOCT ;;READ AN OCTAL NUMBER
11906 ;;* RETURN HERE ;;LOW ORDER BITS ARE ON TOP OF THE STACK
11907 ;;* ;;HIGH ORDER BITS ARE IN $HIOCT
11908 046374 011646          $RDOCT: MOV     (SP),-(SP) ;;PROVIDE SPACE FOR THE
11909 046376 016666 000004 000002  MOV     4(SP),2(SP) ;;INPUT NUMBER
11910 046404 010046          MOV     R0,-(SP)  ;;PUSH R0 ON STACK
```

```
11911 046406 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
11912 046410 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
11913 046412 104411      1$: RDLIN              ;;READ AN ASCII LINE
11914 046414 012600      MOV      (SP)+,R0      ;;GET ADDRESS OF 1ST CHARACTER
11915 046416 010037 046522  MOV      R0,5$         ;;AND SAVE IT
11916 046422 005001      CLR      R1            ;;CLEAR DATA WORD
11917 046424 005002      CLR      R2
11918 046426 112046      2$: MOVB      (R0)+,-(SP) ;;PICKUP THIS CHARACTER
11919 046430 001420      BEQ      3$           ;;IF ZERO GET OUT
11920 046432 122716 000060  CMPB     #'0,(SP)     ;;MAKE SURE THIS CHARACTER
11921 046436 003026      BGT      4$           ;;IS AN OCTAL DIGIT
11922 046440 122716 000067  CMPB     #'7,(SP)
11923 046444 002423      BLT      4$
11924 046446 006301      ASL      R1            ;;*2
11925 046450 006102      ROL      R2
11926 046452 006301      ASL      R1            ;;*4
11927 046454 006102      ROL      R2
11928 046456 006301      ASL      R1            ;;*8
11929 046460 006102      ROL      R2
11930 046462 042716 177770  BIC      #'C7,(SP)    ;;STRIP THE ASCII JUNK
11931 046466 062601      ADD      (SP)+,R1     ;;ADD IN THIS DIGIT
11932 046470 000756      BR       2$           ;;LOOP
11933 046472 005726      3$: TST      (SP)+     ;;CLEAN TERMINATOR FROM STACK
11934 046474 010166 000012  MOV      R1,12(SP)    ;;SAVE THE RESULT
11935 046500 010237 046532  MOV      R2,$HIOCT
11936 046504 012602      MOV      (SP)+,R2     ;;POP STACK INTO R2
11937 046506 012601      MOV      (SP)+,R1     ;;POP STACK INTO R1
11938 046510 012600      MOV      (SP)+,R0     ;;POP STACK INTO R0
11939 046512 000002      RTI
11940 046514 005726      4$: TST      (SP)+     ;;CLEAN PARTIAL FROM STACK
11941 046516 105010      CLR      (R0)         ;;SET A TERMINATOR
11942 046520 104401      TYPE
11943 046522 000000      5$: .WORD     0         ;;TYPE UP THRU THE BAD CHAR.
11944 046524 104401 001272  TYPE     $QUES        ;;'"' 'CR' & 'LF'
11945 046530 000730      BR       1$           ;;TRY AGAIN
11946 046532 000000  $HIOCT: .WORD     0    ;;HIGH ORDER BITS GO HERE
11947      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11948
11949      ;*****
11950      ;*SAVE R0-R5
11951      ;*CALL:
11952      ;* SAVREG
11953      ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11954      ;*
11955      ;*TOP---(+16)
11956      ;* +2---(+18)
11957      ;* +4---R5
11958      ;* +6---R4
11959      ;* +8---R3
11960      ;*+10---R2
11961      ;*+12---R1
11962      ;*+14---R0
11963
11964 046534      $SAVREG:
11965 046534 010046      MOV      R0,-(SP)     ;;PUSH R0 ON STACK
11966 046536 010146      MOV      R1,-(SP)     ;;PUSH R1 ON STACK
```

```
11967 046540 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
11968 046542 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
11969 046544 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
11970 046546 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
11971 046550 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
11972 046554 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
11973 046560 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF CALL
11974 046564 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF CALL
11975 046570 000002      RTI
11976
11977      ;*RESTORE R0-R5
11978      ;*CALL:
11979      ;* RESREG
11980      $RESREG:
11981 046572 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF CALL
11982 046576 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF CALL
11983 046602 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
11984 046606 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
11985 046612 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
11986 046614 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
11987 046616 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
11988 046620 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
11989 046622 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11990 046624 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
11991 046626 000002      RTI
11992      .SBTTL POWER DOWN AND UP ROUTINES
11993
11994      ;*****
11995      ;POWER DOWN ROUTINE
11996 046630 012737 046770 000024 $PWRDN: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
11997 046636 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
11998 046644 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
11999 046646 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
12000 046650 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
12001 046652 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
12002 046654 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
12003 046656 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
12004 046660 017746 132254      MOV      @SWR,-(SP)    ;;PUSH @SWR ON STACK
12005 046664 010637 046774      MOV      SP,$SAVR6    ;;SAVE SP
12006 046670 012737 046702 000024      MOV      #SPWRUP,@#PWRVEC ;;SET UP VECTOR
12007 046676 000000      HALT
12008 046700 000776      BR      -2            ;;HANG UP
12009
12010      ;*****
12011      ;POWER UP ROUTINE
12012 046702 012737 046770 000024 $PWRUP: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12013 046710 013706 046774      MOV      $SAVR6,SP    ;;GET SP
12014 046714 005037 046774      CLR      $SAVR6      ;;WAIT LOOP FOR THE TTY
12015 046720 005237 046774      1$: INC      $SAVR6    ;;WAIT FOR THE INC
12016 046724 001375      BNE     1$           ;;OF WORD
12017 046726 012677 132206      MOV      (SP)+,@SWR   ;;POP STACK INTO @SWR
12018 046732 012605      MOV      (SP)+,R5     ;;POP STACK INTO R5
12019 046734 012604      MOV      (SP)+,R4     ;;POP STACK INTO R4
12020 046736 012603      MOV      (SP)+,R3     ;;POP STACK INTO R3
12021 046740 012602      MOV      (SP)+,R2     ;;POP STACK INTO R2
12022 046742 012601      MOV      (SP)+,R1     ;;POP STACK INTO R1
```

```

12023 046744 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
12024 046746 012737 046630 000024  MOV      #SPWRDN,@PWRVEC  ;;SET UP THE POWER DOWN VECTOR
12025 046754 012737 000340 000026  MOV      #340,@PWRVEC+2  ;;PRIO:7
12026 046762 104401          TYPE                      ;;REPORT THE POWER FAILURE
12027 046764 046776          SPWRMG: .WORD $POWER      ;;POWER FAIL MESSAGE POINTER
12028 046766 000002          RTI
12029 046770 000000          $ILLUP: HALT              ;;THE POWER UP SEQUENCE WAS STARTED
12030 046772 000776          BR      -2                ;; BEFORE THE POWER DOWN WAS COMPLETE
12031 046774 000000          $$AVR6: 0                 ;;PUT THE SP HERE
12032 046776 005015 047520 042527  $POWER: .ASCIZ <15><12>'POWER'
12033 047004 000122
12034          .EVEN
12035          .SBTTL TRAP DECODER
12036
12037          ;;*****
12038          ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12039          ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12040          ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12041          ;;*GO TO THAT ROUTINE.
12042
12043 047006 010046          $TRAP: MOV      R0,-(SP)    ;;SAVE R0
12044 047010 016600 000002  MOV      2(SP),R0          ;;GET TRAP ADDRESS
12045 047014 005740          TST      -(R0)            ;;BACKUP BY 2
12046 047016 111000          MOVVB   (R0),R0          ;;GET RIGHT BYTE OF TRAP
12047 047020 006300          ASL     R0                ;;POSITION FOR INDEXING
12048 047022 016000 047042  MOV      $TRPAD(R0),R0    ;;INDEX TO TABLE
12049 047026 000200          RTS      R0              ;;GO TO ROUTINE
12050
12051          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12052
12053
12054 047030 011646          $TRAP2: MOV     (SP),-(SP)  ;;MOVE THE PC DOWN
12055 047032 016666 000004 000002  MOV     4(SP),2(SP)       ;;MOVE THE PSW DOWN
12056 047040 000002          RTI                      ;;RESTORE THE PSW
12057
12058          .SBTTL TRAP TABLE
12059
12060          ;;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12061          ;;*BY THE "TRAP" INSTRUCTION.
12062
12063          :      ROUTINE
12064          :      -----
12065 047042 047030          $TRPAD: .WORD  $TRAP2
12066 047044 044202          $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
12067 047046 044734          $TYPOC ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12068 047050 044710          $TYPOS ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
12069 047052 044750          $TYPON ;;CALL=TYPON     TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
12070 047054 044464          $TYPDS ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
12071
12072 047056 045524          $GTSWR ;;CALL=GTSWR     TRAP+6(104406) GET SOFT-SWR SETTING
12073
12074 047060 045434          $CKSWR ;;CALL=CKSWR     TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
12075 047062 045776          $RDCHR ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
12076 047064 046066          $RDLIN ;;CALL=RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
12077 047066 046374          $RDOCT ;;CALL=RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
12078 047070 046534          $SAVREG ;;CALL=SAVREG    TRAP+13(104413) SAVE R0-R5 ROUTINE

```

12079	047072	046572	\$RESREG	::CALL=RESREG	TRAP+14(104414)	RESTORE R0-R5 ROUTINE
12080	047074	035222	SCOP1\$::CALL=SCOP1	TRAP+15(104415)	INTERNAL LOOP ON ERROR
12081	047076	037176	SSINIT	::CALL=TSSINIT	TRAP+16(104416)	INITIALIZE SUBSYSTEM
12082	047100	035734	LOADRK	::CALL=TLOADRK	TRAP+17(104417)	LOAD RK611 FOR OPERATION
12083	047102	036164	GETRK	::CALL=TGETRK	TRAP+20(104420)	GET RK611 REGISTERS
12084	047104	040162	CHKOP	::CALL=TCHKOP	TRAP+21(104421)	CHECK OPERATION FOR ANY ERRORS
12085	047106	040130	CHKWE	::CALL=TCHKWE	TRAP+22(104422)	CHECK OPERATION FOR EXPECTED ERRORS
12086	047110	035530	IWAT16	::CALL=TWAT16	TRAP+23(104423)	WAIT 16 MS
12087	047112	035520	IWAT32	::CALL=TWAT32	TRAP+24(104424)	WAIT 32 MS
12088	047114	035510	IWAT48	::CALL=TWAT48	TRAP+25(104425)	WAIT 48 MS
12089	047116	035500	IWAT64	::CALL=TWAT64	TRAP+26(104426)	WAIT 64 MS
12090	047120	035470	IWAT80	::CALL=TWAT80	TRAP+27(104427)	WAIT 80 MS
12091	047122	035460	IWAT96	::CALL=TWAT96	TRAP+30(104430)	WAIT 96 MS
12092	047124	035450	IWAT112	::CALL=TWAT112	TRAP+31(104431)	WAIT 112 MS
12093	047126	035440	IWAT128	::CALL=TWAT128	TRAP+32(104432)	WAIT 128 MS
12094	047130	035430	IWAT144	::CALL=TWAT144	TRAP+33(104433)	WAIT 144 MS
12095	047132	035420	IWAT159	::CALL=TWAT159	TRAP+34(104434)	WAIT 160 MS
12096	047134	035410	IWAT1S	::CALL=TWAT1S	TRAP+35(104435)	WAIT FOR 1 SECOND
12097	047136	035400	IWAT2S	::CALL=TWAT2S	TRAP+36(104436)	WAIT FOR 2 SECONDS
12098	047140	035360	IWAT8S	::CALL=TWAT8S	TRAP+37(104437)	WAIT FOR 8 SECONDS
12099	047142	035370	IWAT1M	::CALL=TWAT1M	TRAP+40(104440)	WAIT FOR 1 MIN
12100	000102					

\$TERM=-.\$TRPAD

```
12101 .SBTTL DATA PATTERNS
12102 :DATA PATTERN 1
12103 : PATTERN IS ALL ZEROS
12104
12105 :DATA PATTERN 2
12106 : HI-LO FREQ. MIX
12107 PAT02:
12108 047144 177777 177777
12109 047146 177777 177777
12110 047150 177777 177777
12111 047152 052525 052525
12112 047154 052525 052525
12113 047156 052525 052525
12114 047160 177777 177777
12115 047162 177777 177777
12116 047164 052525 052525
12117 047166 052525 052525
12118 047170 177777 177777
12119 047172 052525 052525
12120 047174 177252 177252
12121 047176 177252 177252
12122 047200 172765 172765
12123 047202 172765 172765
12124
12125 :DATA PATTERN 3
12126 : HI FREQ. PHASE MIX
12127 PAT03:
12128 047204 000000 000000
12129 047206 000000 000000
12130 047210 000000 000000
12131 047212 177777 177777
12132 047214 177777 177777
12133 047216 177777 177777
12134 047220 000000 000000
12135 047222 000000 000000
12136 047224 177777 177777
12137 047226 177777 177777
12138 047230 000000 000000
12139 047232 177777 177777
12140 047234 000000 000000
12141 047236 177777 177777
12142 047240 000000 000000
12143 047242 177777 177777
12144
12145 :DATA PATTERN 4
12146 : LO FREQ. PHASE MIX
12147 PAT04:
12148 047244 052525 052525
12149 047246 052525 052525
12150 047250 052525 052525
12151 047252 125252 125252
12152 047254 125252 125252
12153 047256 125252 125252
12154 047260 052525 052525
12155 047262 052525 052525
12156 047264 125252 125252
```

12157 047266 125252
12158 047270 052525
12159 047272 125252
12160 047274 052525
12161 047276 125252
12162 047300 052525
12163 047302 125252

125252
052525
125252
052525
125252
052525
125252

:DATA PATTERN 5
: MAX PRECOMP. PHASE MIX
PAT05:

12164
12165
12166
12167 047304
12168 047304 133333
12169 047306 066666
12170 047310 155555
12171 047312 155555
12172 047314 133333
12173 047316 066666
12174 047320 066666
12175 047322 155555
12176 047324 155555
12177 047326 133333
12178 047330 133333
12179 047332 133333
12180 047334 133333
12181 047336 133333
12182 047340 133333
12183 047342 133333

133333
066666
155555
155555
133333
066666
066666
155555
155555
133333
133333
133333
133333
133333
133333
133333

:DATA PATTERN 6
: ROTATING BOUNDARY PULSE PRECOMP.
PAT06:

12184
12185
12186
12187 047344
12188 047344 121105
12189 047346 150442
12190 047350 064221
12191 047352 132110
12192 047354 055044
12193 047356 026422
12194 047360 013211
12195 047362 105504
12196 047364 042642
12197 047366 021321
12198 047370 110550
12199 047372 044264
12200 047374 022132
12201 047376 011055
12202 047400 104426
12203 047402 042213

121105
150442
064221
132110
055044
026422
013211
105504
042642
021321
110550
044264
022132
011055
104426
042213

:DATA PATTERN 7
: FIELD OF ALL ONES

:DATA PATTERN 10
: ROTATING CELL PULSE PRECOMP.
PAT10:

12204
12205
12206
12207
12208
12209
12210 047404
12211 047404 026455
12212 047406 113226

113226
026455

12213	047410	045513	045513
12214	047412	122645	122645
12215	047414	151322	151322
12216	047416	064551	064551
12217	047420	132264	132264
12218	047422	055132	055132
12219	047424	026455	026455
12220	047426	113226	113226
12221	047430	045513	045513
12222	047432	122645	122645
12223	047434	151322	151322
12224	047436	064551	064551
12225	047440	132264	132264
12226	047442	055132	055132

;DATA PATTERN 11
; SHIFTED 1 IN A FIELD OF ZEROS
PAT11:

12230	047444		
12231	047444	000001	000001
12232	047446	000002	000002
12233	047450	000004	000004
12234	047452	000010	000010
12235	047454	000020	000020
12236	047456	000040	000040
12237	047460	000100	000100
12238	047462	000200	000200
12239	047464	000400	000400
12240	047466	001000	001000
12241	047470	002000	002000
12242	047472	004000	004000
12243	047474	010000	010000
12244	047476	020000	020000
12245	047500	040000	040000
12246	047502	100000	100000

;DATA PATTERN 12
; SHIFTED 0 IN A FIELD OF ONES
PAT12:

12250	047504		
12251	047504	177776	177776
12252	047506	177775	177775
12253	047510	177773	177773
12254	047512	177767	177767
12255	047514	177757	177757
12256	047516	177737	177737
12257	047520	177677	177677
12258	047522	177577	177577
12259	047524	177377	177377
12260	047526	176777	176777
12261	047530	175777	175777
12262	047532	173777	173777
12263	047534	167777	167777
12264	047536	157777	157777
12265	047540	137777	137777
12266	047542	077777	077777

;DATA PATTERN 13

12267
12268

12269			: PAT13: ALTERNATING 0-1
12270	047544		052525
12271	047544	052525	052525
12272	047546	052525	052525
12273	047550	052525	052525
12274	047552	052525	052525
12275	047554	052525	052525
12276	047556	052525	052525
12277	047560	052525	052525
12278	047562	052525	052525
12279	047564	052525	052525
12280	047566	052525	052525
12281	047570	052525	052525
12282	047572	052525	052525
12283	047574	052525	052525
12284	047576	052525	052525
12285	047600	052525	052525
12286	047602	052525	052525

12287			
12288			: DATA PATTERN 14
12289			: ALTERNATING 1-0
12290	047604		: PAT14:
12291	047604	125252	125252
12292	047606	125252	125252
12293	047610	125252	125252
12294	047612	125252	125252
12295	047614	125252	125252
12296	047616	125252	125252
12297	047620	125252	125252
12298	047622	125252	125252
12299	047624	125252	125252
12300	047626	125252	125252
12301	047630	125252	125252
12302	047632	125252	125252
12303	047634	125252	125252
12304	047636	125252	125252
12305	047640	125252	125252
12306	047642	125252	125252

12307			
12308			: DATA PATTERN 15
12309			: SHIFTING ZEROS AND ONES
12310	047644		: PAT15:
12311	047644	000001	000001
12312	047646	000003	000003
12313	047650	000007	000007
12314	047652	000017	000017
12315	047654	000037	000037
12316	047656	000077	000077
12317	047660	000177	000177
12318	047662	000377	000377
12319	047664	000777	000777
12320	047666	001777	001777
12321	047670	003777	003777
12322	047672	007777	007777
12323	047674	017777	017777
12324	047676	037777	037777

12325	047700	077777	077777
12326	047702	177777	177777
12327			
12328			
12329			
12330	047704		
12331	047704	072307	072307
12332	047706	135143	135143
12333	047710	156461	156461
12334	047712	167230	167230
12335	047714	073514	073514
12336	047716	035646	035646
12337	047720	016723	016723
12338	047722	107351	107351
12339	047724	143564	143564
12340	047726	061672	061672
12341	047730	030735	030735
12342	047732	114356	114356
12343	047734	046167	046167
12344	047736	123073	123073
12345	047740	151453	151453
12346	047742	164616	164616
12347			

:DATA PATTERN 16
: COMPOSITE ROTATING
PAT16:

```
12348          .SBTTL FIELDS AND VARIABLES FOR OPERATION CHECKING
12349          CS1ERBIT=24000          ;CS1 ERROR BITS SPAR & CTO
12350          177400                  ;CS2 ERROR BITS
12351          ;DLT,WCE,UPE,NED,NEM
12352          ;PGE,MOS,UFE
12353          000070                  DSERBIT=70
12354          ;DS ERROR BITS
12355          ;SPDLSS,DROT,ACLO
12356 047744 000000 EXPWC: .WORD 0          ;EXPECTED WORD COUNT (GIVEN)
12357 047746 000000 EXPUBA: .WORD 0        ;EXPECTED UPPER BA (COMPUTED)
12358 047750 000000 EXPBA: .WORD 0        ;EXPECTED BUS ADDRESS (COMPUTED)
12359 047752 000000 EXPCYL: .WORD 0       ;EXPECTED CYLINDER (COMPUTED)
12360 047754 000000 EXPSEC: .WORD 0       ;EXPECTED SECTOR (COMPUTED)
12361 047756 000000 EXPTRK: .WORD 0      ;EXPECTED TRACK (COMPUTED)
12362
12363 047760 000000 REALWC: .WORD 0        ;WORD COUNT AT END OF OPERATION
12364 047762 000000 REALUB: .WORD 0        ;REAL UPPER BA
12365 047764 000000 REALBA: .WORD 0        ;BUS ADDRESS
12366 047766 000000 REALCY: .WORD 0       ;CYLINDER
12367 047770 000000 REALTRK: .WORD 0      ;TRACK
12368 047772 000000 REALSEC: .WORD 0     ;SECTOR
12369
12370 047774 000000 GRP1ER: .WORD 0        ;GROUP 1 ERROR FIELDS
12371          000001 SPARERR=BIT0      ;CONTROLLER DETECTED DRIVE BUS PARITY ERR
12372          000002 SKIERR= BIT1       ;SEEK INCOMPLETE
12373          000004 NXFERR= BIT2       ;NON-EXECUTABLE DRIVE FUNCTION
12374          000010 DRPARERR=BIT3     ;DRIVE DETECTED DRIVE BUS PARITY ERROR
12375          000020 FMTERR= BIT4       ;FORMAT ERROR
12376          000040 DTYERR= BIT5       ;DRIVE TYPE ERROR
12377          000100 ACLOERR=BIT6      ;AC LOW ERROR
12378          000200 SPDERR= BIT7       ;SPEED LOSS ERROR
12379          000400 DROTERR=BIT8     ;DRIVE OFF TRACK ERROR
12380          001000 COERR= BIT9        ;CYLINDER OVER FLOW ERROR
12381          002000 IDAERR= BIT10     ;ILLEGAL DISK ADDRESS ERROR
12382          004000 WLERR= BIT11      ;WRITE LOCK ERROR
12383          010000 DTERR= BIT12     ;DRIVE TIMING ERROR
12384          020000 NCERWE= BIT13     ;NO CERR WITH ERROR SET ERROR
12385          040000 UNSERR= BIT14     ;DRIVE UNSAFE ERROR
12386          100000 CERNER= BIT15     ;CERR BUT NO ERROR SET ERROR
12387
12388 047776 000000 GRP2ER: .WORD 0        ;GROUP 2 ERROR FIELD
12389          000001 ECHERR= BIT0      ;ECC HARD ERROR
12390          000002 DCKERR= BIT1      ;DATA CHECK ERROR
12391          000004 WCKERR= BIT2      ;WRITE CHECK ERROR
12392          000010 DLTERR= BIT3      ;DATA LATE ERROR
12393          000020 OPIERR= BIT4      ;OPERATION INCOMPLETE ERROR
12394          000040 HVRCERR=BIT5     ;HEADER VRC ERROR
12395          000100 BSERR= BIT6       ;BAD SECTOR ERROR
12396
12397 050000 000000 GRP3ER: .WORD 0        ;GROUP 3 ERROR FIELD
12398          000001 NEDERR= BIT0      ;NON-EXISTANT DRIVE ERROR
12399          000002 CTOERR= BIT1     ;CONTROLLER TIME OUT ERROR
12400          000004 UFERR= BIT2     ;UNIT FIELD ERROR
12401          000010 MDSERR= BIT3     ;MULTIPLE DRIVE SELECT ERROR
12402          000020 PGERR= BIT4     ;PROGRAMMING ERROR
12403          000040 NEMERR= BIT5     ;NON-EXISTANT MEMORY ERROR
```

12404		000100		UPERR= BIT6	:UNIBUS PARITY ERROR
12405		000200		ILFERR= BIT7	:ILLEGAL FUNCTION ERROR.
12406					
12407	050002	000000	GRP4ER:	.WORD 0	:GROUP 4 ERROR FIELD
12408		000001		WCERR= BIT0	:WORD COUNT ERROR FLAG
12409		000002		UBAERR= BIT1	:UPPER BA ERROR
12410		000004		BAERR= BIT2	:BUS ADDRESS ERROR FLAG
12411		000010		CYLERR= BIT3	:CYL ADDRESS ERROR FLAG
12412		000020		TRKERR= BIT4	:TRACK ADDRESS ERROR FLAG
12413		000040		SECERR= BIT5	:SECTOR ADDRESS ERROR FLAG
12414					
12415	050004	054006	CRP4MS:	.WORD EM10	
12416	050006	054061		.WORD EM11A	
12417	050010	054033		.WORD EM11	
12418	050012	054135		.WORD EM12	
12419	050014	054170		.WORD EM13	
12420	050016	054220		.WORD EM14	
12421					
12422	050020	054252	GRP3MS:	.WORD EM15	
12423	050022	054275		.WORD EM16	
12424	050024	054320		.WORD EM17	
12425	050026	054341		.WORD EM18	
12426	050030	054366		.WORD EM19	
12427	050032	054410		.WORD EM20	
12428	050034	054434		.WORD EM21	
12429	050036	054460		.WORD EM22	
12430					
12431	050040	054501	GRP2MS:	.WORD EM23	
12432	050042	054512		.WORD EM24	
12433	050044	054525		.WORD EM25	
12434	050046	054541		.WORD EM26	
12435	050050	054553		.WORD EM27	
12436	050052	054600		.WORD EM28	
12437	050054	054613		.WORD EM29	
12438					
12439	050056	054634	GRP1MS:	.WORD EM30	
12440	050060	054707		.WORD EM31	
12441	050062	054727		.WORD EM32	
12442	050064	054765		.WORD EM33	
12443	050066	055033		.WORD EM34	
12444	050070	055050		.WORD EM35	
12445	050072	055071		.WORD EM36	
12446	050074	055100		.WORD EM37	
12447	050076	055123		.WORD EM38	
12448	050100	055143		.WORD EM39	
12449	050102	055165		.WORD EM40	
12450	050104	055217		.WORD EM41	
12451	050106	055240		.WORD EM42	
12452	050110	055263		.WORD EM43	
12453	050112	055325		.WORD EM44	
12454	050114	055342		.WORD EM45	
12455					
12456	050116	000000	GPSUMF:	.WORD 0	:GROUP ERROR SUMMARY FLAGS
12457		000001		GRP1ST= BIT0	:GROUP 1 ERROR SET
12458		000002		GRP2ST= BIT1	:GROUP 2 ERROR SET
12459		000004		GRP3ST= BIT2	:GROUP 3 ERROR SET

12460	000010	GP1NR= BIT3	:GROUP 1 ERROR NOT RECEIVED
12461	000020	GP2NR= BIT4	:GROUP 2 ERROR NOT RECEIVED
12462	000040	GP3NR= BIT5	:GROUP 3 ERROR NOT RECEIVED
12463	040000	DRSTER= BIT14	:ERROR IN GETTING DRIVE STATUS FLAG.
12464	100000	REPNR= BIT15	:REPORTING NOT RECEIVED SWITCH

12465
12466 .SBTTL TABLE OF OPERATION MESSAGE ADDRESS
12467 :* THIS TABLE CONTAINS THE ADDRESS OF ASCIZ FIELDS THAT ARE
12468 :* USED IN REPORTING TO IDENTIFY THE OPERATION BEING PERFORMED.

12470	050120	050160	CMNDLB: .WORD	OPER00	:ADDRESS OF SELECT MESSAGE
12471	050122	050175	.WORD	OPER02	:PACK ACK
12472	050124	050206	.WORD	OPER04	:DRIVE CLEAR
12473	050126	050222	.WORD	OPER06	:UNLOAD
12474	050130	050231	.WORD	OPER10	:START SPINDLE
12475	050132	050247	.WORD	OPER12	:RECALIBRATE
12476	050134	050263	.WORD	OPER14	:OFFSET
12477	050136	050272	.WORD	OPER16	:SEEK
12478	050140	050277	.WORD	OPER20	:READ DATA
12479	050142	050311	.WORD	OPER22	:WRITE DATA
12480	050144	050324	.WORD	OPER24	:READ HEADER
12481	050146	050341	.WORD	OPER26	:WRITE HEADER
12482	050150	050357	.WORD	OPER30	:WRITE CHECK
12483	050152	050373	.WORD	OPER32	:ILLEGAL OPERATION 33
12484	050154	050417	.WORD	OPER34	:35
12485	050156	050443	.WORD	OPER36	:37

12486
12487 .SBTTL OPERATION MESSAGES
12488 050160 051104 053111 020105 OPER00: .ASCIZ /DRIVE SELECT/
12489 050166 042523 042514 052103
12490 050174 000
12491 050175 120 041501 020113 OPER02: .ASCIZ /PACK ACK/
12492 050202 041501 000113
12493 050206 051104 053111 020105 OPER04: .ASCIZ /DRIVE CLEAR/
12494 050214 046103 040505 000122
12495 050222 047125 047514 042101 OPER06: .ASCIZ /UNLOAD/
12496 050230 000
12497 050231 123 040524 052122 OPER10: .ASCIZ /START SPINDLE/
12498 050236 051440 044520 042116
12499 050244 042514 000
12500 050247 122 041505 046101 OPER12: .ASCIZ /RECALIBRATE/
12501 050254 041111 040522 042524
12502 050262 000
12503 050263 117 043106 042523 OPER14: .ASCIZ /OFFSET/
12504 050270 000124
12505 050272 042523 045505 000 OPER16: .ASCIZ /SEEK/
12506 050277 122 040505 020104 OPER20: .ASCIZ /READ DATA/
12507 050304 040504 040524 000
12508 050311 127 044522 042524 OPER22: .ASCIZ /WRITE DATA/
12509 050316 042040 052101 000101
12510 050324 042522 042101 044040 OPER24: .ASCIZ /READ HEADERS/
12511 050332 040505 042504 051522
12512 050340 000
12513 050341 127 044522 042524 OPER26: .ASCIZ /WRITE HEADERS/
12514 050346 044040 040505 042504
12515 050354 051522 000

12516	050357	127	044522	042524	OPER30: .ASCIZ /WRITE CHECK/
12517	050364	041440	042510	045503	
12518	050372	000			
12519	050373	111	046114	043505	OPER32: .ASCIZ /ILLEGAL FUNCTION 33/
12520	050400	046101	043040	047125	
12521	050406	052103	047511	020116	
12522	050414	031463	000		
12523	050417	111	046114	043505	OPER34: .ASCIZ /ILLEGAL FUNCTION 35/
12524	050424	046101	043040	047125	
12525	050432	052103	047511	020116	
12526	050440	032463	000		
12527	050443	111	046114	043505	OPER36: .ASCIZ /ILLEGAL FUNCTION 37/
12528	050450	046101	043040	047125	
12529	050456	052103	047511	020116	
12530	050464	033463	000		
12531	050467	127	044522	042524	OPER37: .ASCIZ /WRITE DATA ABORTED WITH BSE/
12532	050474	042040	052101	020101	
12533	050502	041101	051117	042524	
12534	050510	020104	044527	044124	
12535	050516	041040	042523	000	
12536	050523				OPER40:
12537	050523	127	044522	042524	OPER41: .ASCIZ /WRITE CHECK ABORTED WITH WCE/
12538	050530	041440	042510	045503	
12539	050536	040440	047502	052122	
12540	050544	042105	053440	052111	
12541	050552	020110	041527	000105	
12542	050560	051127	052111	020105	OPER42: .ASCIZ /WRITE DATA ABORTED WITH NON-EXISTANT MEMORY/
12543	050566	040504	040524	040440	
12544	050574	047502	052122	042105	
12545	050602	053440	052111	020110	
12546	050610	047516	026516	054105	
12547	050616	051511	040524	052116	
12548	050624	046440	046505	051117	
12549	050632	000131			
12550	050634	042522	042101	042040	OPER43: .ASCIZ /READ DATA ABORTED WITH NON-EXISTANT MEMORY/
12551	050642	052101	020101	041101	
12552	050650	051117	042524	020104	
12553	050656	044527	044124	047040	
12554	050664	047117	042455	044530	
12555	050672	052123	047101	020124	
12556	050700	042515	047515	054522	
12557	050706	000			
12558	050707	127	044522	042524	OPER44: .ASCIZ /WRITE DATA ABORTED WITH UNIBUS PARITY ERROR/
12559	050714	042040	052101	020101	
12560	050722	041101	051117	042524	
12561	050730	020104	044527	044124	
12562	050736	052440	044516	052502	
12563	050744	020123	040520	044522	
12564	050752	054524	042440	051122	
12565	050760	051117	000		
12566					

```
12567 .SBTTL ASCII MESSAGES
12568
12569 050763 040 000040 SPACE2: .ASCIZ / /
12570 050766 005015 045522 030466 OPR001: .ASCIZ <15><12>/RK611 BUS ADDRESS ( /
12571 050774 020061 052502 020123
12572 051002 042101 051104 051505
12573 051010 020123 020050 000
12574 051015 040 020051 020075 OPR002: .ASCIZ / ) = /
12575 051022 000
12576 051023 122 033113 030461 OPR003: .ASCIZ /RK611 VECTOR ADDRESS ( /
12577 051030 053040 041505 047524
12578 051036 020122 042101 051104
12579 051044 051505 020123 020050
12580 051052 000
12581 051053 122 033113 030461 OPR004: .ASCIZ /RK611 PRIORITY ( /
12582 051060 050040 044522 051117
12583 051066 052111 020131 020050
12584 051074 000
12585 051075 111 051516 043125 OPR005: .ASCIZ /INSUFFICIENT MEMORY. PROGRAM ABORTING./<15><12>
12586 051102 044506 044503 047105
12587 051110 020124 042515 047515
12588 051116 054522 020056 051120
12589 051124 043517 040522 020115
12590 051132 041101 051117 044524
12591 051140 043516 006456 000012
12592 051146
12593 OPR006:
: .ASCII <15><12>/TO BYPASS TESTING DRIVE 0, PLACE IT OFF-LINE/
12594 : .ASCIZ <15><12>/TO TEST DRIVE 0, REPLACE PROGRAM PACK WITH SCRATCH PACK/
12595 051146 OPR007:
12596 051146 005015 044103 047101 .ASCII <CR><LF>/CHANGE XXDP PACK/
12597 051154 042507 054040 042130
12598 051162 020120 040520 045503
12599 051170 005015 046103 040505 .ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
12600 051176 020122 047514 020103
12601 051204 030064 051054 051505
12602 051212 040524 052122 050040
12603 051220 047522 051107 046501
12604 051226 000
12605 : .ASCII <15><12>/DRIVE 0 WILL NOT BE TESTED. TO TEST DRIVE 0./<15><12>
12606 : .ASCIZ /RESTART AT 214 AND MOUNT SCRATCH PACK AS DIRECTED./
12607 051227 015 047012 020117 OPR008: .ASCIZ <15><12>/NO DRIVES AVAILABLE FOR TESTING. PROGRAM ABORTED/
12608 051234 051104 053111 051505
12609 051242 040440 040526 046111
12610 051250 041101 042514 043040
12611 051256 051117 052040 051505
12612 051264 044524 043516 020056
12613 051272 051120 043517 040522
12614 051300 020115 041101 051117
12615 051306 042524 000104
12616 051312 005015 044124 020105 OPR009: .ASCIZ <15><12>/THE FOLLOWING DRIVES WILL BE TESTED/<15><12>
12617 051320 047506 046114 053517
12618 051326 047111 020107 051104
12619 051334 053111 051505 053440
12620 051342 046111 020114 042502
12621 051350 052040 051505 042524
12622 051356 006504 000012
```

12623	051362	005015	047516	050040	OPR010: .ASCII <15><12>/NO PARITY OPTION FOR MEMORY AREA IN USE/<15><12>
12624	051370	051101	052111	020131	
12625	051376	050117	044524	047117	
12626	051404	043040	051117	046440	
12627	051412	046505	051117	020131	
12628	051420	051101	040505	044440	
12629	051426	020116	051525	006505	
12630	051434	012			
12631	051435	055	052440	044516	.ASCIZ /- UNIBUS PARITY ERROR TEST BYPASSED/<15><12>
12632	051442	052502	020123	040520	
12633	051450	044522	054524	042440	
12634	051456	051122	051117	052040	
12635	051464	051505	020124	054502	
12636	051472	040520	051523	042105	
12637	051500	005015	000		
12638	051503	015	046412	046505	OPR011: .ASCII <15><12>/MEMORY SIZE NOT LARGE ENOUGH FOR BUS ADDRESS BITS 16 & 17 TESTS
12639	051510	051117	020131	044523	
12640	051516	042532	047040	052117	
12641	051524	046040	051101	042507	
12642	051532	042440	047516	043525	
12643	051540	020110	047506	020122	
12644	051546	052502	020123	042101	
12645	051554	051104	051505	020123	
12646	051562	044502	051524	030440	
12647	051570	020066	020046	033461	
12648	051576	052040	051505	051524	
12649	051604	005015			
12650	051606	046101	020114	040504	.ASCIZ /ALL DATA XFER TESTS WITH ADDR >/
12651	051614	040524	054040	042506	
12652	051622	020122	042524	052123	
12653	051630	020123	044527	044124	
12654	051636	040440	042104	020122	
12655	051644	000076			
12656	051646	031063	000113		OPR012: .ASCIZ /32K/
12657	051652	032066	000113		OPR013: .ASCIZ /64K/
12658	051656	033071	000113		OPR014: .ASCIZ /96K/
12659	051662	041040	050131	051501	OPR015: .ASCIZ / BYPASSED/<15><12>
12660	051670	042523	006504	000012	
12661	051676	005015	020012	020040	OPR016: .ASCII <15><12><12>/ *** CAUTION ***/<15><12><12>
12662	051704	020040	020040	025052	
12663	051712	020052	040503	052125	
12664	051720	047511	020116	025052	
12665	051726	006452	005012		
12666	051732	044124	051511	050040	.ASCII /THIS PROGRAM SHOULD BE HALTED BY TYPING ^C./<15><12>
12667	051740	047522	051107	046501	
12668	051746	051440	047510	046125	
12669	051754	020104	042502	044040	
12670	051762	046101	042524	020104	
12671	051770	054502	052040	050131	
12672	051776	047111	020107	041536	
12673	052004	006456	012		
12674	052007	111	020106	040510	.ASCII /IF HALTED USING THE HALT KEY, THE STATE OF THE DRIVE/<15><12>
12675	052014	052114	042105	052440	
12676	052022	044523	043516	052040	
12677	052030	042510	044040	046101	
12678	052036	020124	042513	026131	

12679	052044	052040	042510	051440
12680	052052	040524	042524	047440
12681	052060	020106	044124	020105
12682	052066	051104	053111	006505
12683	052074	012		
12684	052075	117	020122	040503
12685	052102	052122	044522	043504
12686	052110	020105	040503	047116
12687	052116	052117	041040	020105
12688	052124	051120	042105	041511
12689	052132	042524	027104	005015
12690	052140	012		
12691	052141	101	046114	042040
12692	052146	044522	042526	020123
12693	052154	047524	041040	020105
12694	052162	042524	052123	042105
12695	052170	046440	051525	020124
12696	052176	042502	047440	026516
12697	052204	044514	042516	006454
12698	052212	012		
12699	052213	122	040505	054504
12700	052220	020054	047101	020104
12701	052226	051127	052111	020105
12702	052234	047514	045503	051040
12703	052242	051505	052105	006456
12704	052250	012		
12705	052251	101	054516	042040
12706	052256	044522	042526	047040
12707	052264	052117	052040	020117
12708	052272	042502	052040	051505
12709	052300	042524	020104	052515
12710	052306	052123	041040	020105
12711	052314	043117	026506	044514
12712	052322	042516	006456	005012
12713	052330	047516	042524	020072
12714	052336	047062	020104	047101
12715	052344	020104	052523	051502
12716	052352	050505	042525	052116
12717	052360	050040	051501	020123
12718	052366	052522	020116	044524
12719	052374	042515	044440	006523
12720	052402	012		
12721	052403	040	020040	020040
12722	052410	040440	050120	047522
12723	052416	020130	020062	044515
12724	052424	020116	030063	051440
12725	052432	041505	043040	051117
12726	052440	042440	041501	020110
12727	052446	051104	053111	027105
12728	052454	005015	000	
12729	052457	015	043012	051111
12730	052464	052123	031040	033065
12731	052472	051440	041505	047524
12732	052500	051522	047040	052117
12733	052506	041040	042523	042440
12734	052514	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>

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

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

12735	052522	042522	027105	
12736	052526	040515	044530	052515 .ASCIZ /MAXIMUM DATA TRANSFER TEST BYPASSED/<15><12>
12737	052534	020115	040504	040524
12738	052542	052040	040522	051516
12739	052550	042506	020122	042524
12740	052556	052123	041040	050131
12741	052564	051501	042523	006504
12742	052572	000012		
12743	052574	020040	020040	006440 OPR018: .ASCIZ / /<15><12>/ONLY 1 DRIVE. OVERLAPPED OPERATIONS BYPASSED/<15><12>
12744	052602	047412	046116	020131
12745	052610	020061	051104	053111
12746	052616	027105	047440	042526
12747	052624	046122	050101	042520
12748	052632	020104	050117	051105
12749	052640	052101	047511	051516
12750	052646	041040	050131	051501
12751	052654	042523	006504	000012
12752	052662	005015	041523	050117 OPR019: .ASCII <15><12>@SCOPE: CH1 (TRIG), E53-8; CH2, E49-2 (AC COUPLE, .2V/CM)@
12753	052670	035105	041440	030510
12754	052676	024040	051124	043511
12755	052704	026051	042440	031465
12756	052712	034055	020073	044103
12757	052720	026062	042440	034464
12758	052726	031055	024040	041501
12759	052734	041440	052517	046120
12760	052742	026105	027040	053062
12761	052750	041457	024515	
12762	052754	005015	042101	052512 .ASCIZ <15><12>/ADJUST R72 FOR CONSTANT LEVEL ON CH2/<15><12>
12763	052762	052123	051040	031067
12764	052770	043040	051117	041440
12765	052776	047117	052123	047101
12766	053004	020124	042514	042526
12767	053012	020114	047117	041440
12768	053020	031110	005015	000
12769	053025	015	050012	047522 OPR020: .ASCIZ <15><12>/PROGRAM HALT PENDING - CARTRIDGE FORMAT BEING CORRECTED/<15><12>
12770	053032	051107	046501	044040
12771	053040	046101	020124	042520
12772	053046	042116	047111	020107
12773	053054	020055	040503	052122
12774	053062	044522	043504	020105
12775	053070	047506	046522	052101
12776	053076	041040	044505	043516
12777	053104	041440	051117	042522
12778	053112	052103	042105	005015
12779	053120	000		
12780	053121	015	025012	025052 OPR021: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
12781	053126	025052	020040	051120
12782	053134	043517	040522	020115
12783	053142	040510	052114	042105
12784	053150	020040	025052	025052
12785	053156	006452	000012	
12786	053162	040503	052122	044522 OPR022: .ASCIZ /CARTRIDGE FORMAT CORRECTION FAILED/<15><12>
12787	053170	043504	020105	047506
12788	053176	046522	052101	041440
12789	053204	051117	042522	052103
12790	053212	047511	020116	040506

CZ
CZ
AB
AB
AC
AC
AC
AC
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AD
AE
AF
AM
AM
AM
AM
AM
AM
AM
AM
AM
AM
AP
AP
AP
AP
AS
AT
AU
AV

12791 053220 046111 042105 005015
12792 053226 000
12793 053227 015 050012 047522
12794 053234 051107 046501 040440
12795 053242 047502 052122 047111
12796 053250 020107 042502 040503
12797 053256 051525 020105 051105
12798 053264 047522 020122 044124
12799 053272 042522 044123 046117
12800 053300 020104 054105 042503
12801 053306 042105 042105 005015
12802 053314 000

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

12803
12804
12805 053315 106 052101 046101
12806 053322 047055 047117 042440
12807 053330 044530 052123 047101
12808 053336 020124 042515 047515
12809 053344 054522 040440 020124
12810 053352 045522 030466 020061
12811 053360 040502 042523 040440
12812 053366 042104 042522 051523

.SBTTL ERROR MESSAGES

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

12813 053374 000
12814 053375 106 052101 046101
12815 053402 053455 044522 042524
12816 053410 051040 040505 054504
12817 053416 040440 042116 044440
12818 053424 020105 044504 020104
12819 053432 047516 020124 040503
12820 053440 051525 020105 047111
12821 053446 042524 051122 050125

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

12822 053454 000124
12823 053456 040506 040524 026514
12824 053464 040520 044522 054524
12825 053472 042440 051122 051117
12826 053500 052040 040522 027120
12827 053506 050040 020103 052101
12828 053514 042440 051122 051117

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

12829 053522 036440 000040
12830 053526 054105 042520 052103
12831 053534 042105 044440 052116
12832 053542 051105 052522 052120
12833 053550 042040 042111 047040
12834 053556 052117 047440 041503
12835 053564 051125 047440 020122
12836 053572 040527 020123 040514
12837 053600 042524 020056 047503
12838 053605 046515 047101 020104

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

12839 053614 040527 035123 000
12840 053621 123 041125 054523
12841 053626 052123 046505 041440
12842 053634 042514 051101 042040
12843 053642 042111 047040 052117
12844 053650 051040 051505 052105
12845 053656 042440 051122 051117
12846 053664 000

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

12847	053665	123	041125	054523	EM6:	.ASCIZ /SUBSYSTEM CLEAR DID NOT RESET DEVICE INTERRUPT/
12848	053672	052123	046505	041440		
12849	053700	042514	051101	042040		
12850	053706	042111	047040	052117		
12851	053714	051040	051505	052105		
12852	053722	042040	053105	041511		
12853	053730	020105	047111	042524		
12854	053736	051122	050125	000124		
12855	053744	052523	051502	051531	EM7:	.ASCIZ /SUBSYSTEM CLEAR DID NOT SET READY/
12856	053752	042524	020115	046103		
12857	053760	040505	020122	044504		
12858	053766	020104	047516	020124		
12859	053774	042523	021124	042522		
12860	054002	042101	000131			
12861	054006	047527	042122	041440	EM10:	.ASCIZ /WORD COUNT INCORRECT/
12862	054014	052517	052116	044440		
12863	054022	041516	051117	042522		
12864	054030	052103	000			
12865	054033	102	051525	040440	EM11:	.ASCIZ /BUS ADDRESS INCORRECT/
12866	054040	042104	042522	051523		
12867	054046	044440	041516	051117		
12868	054054	042522	052103	000		
12869	054061	125	050120	051105	EM11A:	.ASCIZ /UPPER BUS ADDRESS BITS INCORRECT (BA16, 17)/
12870	054066	041040	051525	040440		
12871	054074	042104	042522	051523		
12872	054102	041040	052111	020123		
12873	054110	047111	047503	051122		
12874	054116	041505	020124	041050		
12875	054124	030501	026066	030440		
12876	054132	024467	000			
12877	054135	103	046131	047111	EM12:	.ASCIZ /CYLINDER ADDRESS INCORRECT/
12878	054142	042504	020122	042101		
12879	054150	051104	051505	020123		
12880	054156	047111	047503	051122		
12881	054164	041505	000124			
12882	054170	051124	041501	020113	EM13:	.ASCIZ /TRACK ADDRESS INCORRECT/
12883	054176	042101	051104	051505		
12884	054204	020123	047111	047503		
12885	054212	051122	041505	000124		
12886	054220	042523	052103	051117	EM14:	.ASCIZ /SECTOR ADDRESS INCORRECT./
12887	054226	040440	042104	042522		
12888	054234	051523	044440	041516		
12889	054242	051117	042522	052103		
12890	054250	000056				
12891	054252	047516	026516	054105	EM15:	.ASCIZ /NCN-EXISTANT DR'VE/
12892	054260	051511	040524	052116		
12893	054266	042040	044522	042526		
12894	054274	000				
12895	054275	103	047117	051124	EM16:	.ASCIZ /CONTROLLER TIMEOUT/
12896	054302	046117	042514	020122		
12897	054310	044524	042515	052517		
12898	054316	000124				
12899	054320	047125	052111	043040	EM17:	.ASCIZ /UNIT FIELD ERROR/
12900	054326	042511	042114	042440		
12901	054334	051122	051117	000		
12902	054341	115	046125	050111	EM18:	.ASCIZ /MULIPLE DRIVE SELECT/

12903	054346	042514	042040	044522		
12904	054354	042526	051440	046105		
12905	054362	041505	000124			
12906	054366	051120	043517	040522	EM19:	.ASCIZ /PROGRAMMING ERROR/
12907	054374	046515	047111	020107		
12908	054402	051105	047522	000122		
12909	054410	047516	026516	054105	EM20:	.ASCIZ /NON-EXISTANT MEMORY/
12910	054416	051511	040524	052116		
12911	054424	046440	046505	051117		
12912	054432	000131				
12913	054434	047125	041111	051525	EM21:	.ASCIZ /UNIBUS PARITY ERROR/
12914	054442	050040	051101	052111		
12915	054450	020131	051105	047522		
12916	054456	000122				
12917	054460	046111	042514	040507	EM22:	.ASCIZ /ILLEGAL FUNCTION/
12918	054466	020114	052506	041516		
12919	054474	044524	047117	000		
12920	054501	105	041503	044040	EM23:	.ASCIZ /ECC HARD/
12921	054506	051101	000104			
12922	054512	040504	040524	041440	EM24:	.ASCIZ /DATA CHECK/
12923	054520	042510	045503	000		
12924	054525	127	044522	042524	EM25:	.ASCIZ /WRITE CHECK/
12925	054532	041440	042510	045503		
12926	054540	000				
12927	054541	104	052101	020101	EM26:	.ASCIZ /DATA LATE/
12928	054546	040514	042524	000		
12929	054553	117	042520	040522	EM27:	.ASCIZ /OPERATION INCOMPLETE/
12930	054560	044524	047117	044440		
12931	054566	041516	046517	046120		
12932	054574	052105	000105			
12933	054600	042510	042101	051105	EM28:	.ASCIZ /HEADER VRC/
12934	054606	053040	041522	000		
12935	054613	102	042101	051440	EM29:	.ASCIZ /BAD SECTOR ERROR/
12936	054620	041505	047524	020122		
12937	054626	051105	047522	000122		
12938	054634	047503	052116	047522	EM30:	.ASCIZ /CONTROLLER DETECTED DRIVE BUS PARITY ERROR/
12939	054642	046114	051105	042040		
12940	054650	052105	041505	042524		
12941	054656	020104	051104	053111		
12942	054664	020105	052502	020123		
12943	054672	040520	044522	054524		
12944	054700	042440	051122	051117		
12945	054706	000				
12946	054707	123	042505	020113	EM31:	.ASCIZ /SEEK INCOMPLETE/
12947	054714	047111	047503	050115		
12948	054722	042514	042524	000		
12949	054727	116	047117	042455	EM32:	.ASCIZ /NON-EXECUTABLE DRIVE FUNCTION/
12950	054734	042530	052503	040524		
12951	054742	046102	020105	051104		
12952	054750	053111	020105	052506		
12953	054756	041516	044524	047117		
12954	054764	000				
12955	054765	104	044522	042526	EM33:	.ASCIZ /DRIVE DETECTED DRIVE BUS PARITY ERROR/
12956	054772	042040	052105	041505		
12957	055000	042524	020104	051104		
12958	055006	053111	020105	052502		

12959	055014	020123	040520	044522		
12960	055022	054524	042440	051122		
12961	055030	051117	000			
12962	055033	106	051117	040515	EM34:	.ASCIZ /FORMAT ERROR/
12963	055040	020124	051105	047522		
12964	055046	000122				
12965	055050	051104	053111	020105	EM35:	.ASCIZ /DRIVE TYPE ERROR/
12966	055056	054524	042520	042440		
12967	055064	051122	051117	000		
12968	055071	101	020103	047514	EM36:	.ASCIZ /AC LOW/
12969	055076	000127				
12970	055100	050123	047111	046104	EM37:	.ASCIZ /SPINDLE SPEED LOSS/
12971	055106	020105	050123	042505		
12972	055114	020104	047514	051523		
12973	055122	000				
12974	055123	104	044522	042526	EM38:	.ASCIZ /DRIVE OFF TRACK/
12975	055130	047440	043106	052040		
12976	055136	040522	045503	000		
12977	055143	103	046131	047111	EM39:	.ASCIZ /CYLINDER OVERFLOW/
12978	055150	042504	020122	053117		
12979	055156	051105	046106	053517		
12980	055164	000				
12981	055165	111	046114	043505	EM40:	.ASCIZ /ILLEGAL DISK PACK ADDRESS/
12982	055172	046101	042040	051511		
12983	055200	020113	040520	045503		
12984	055206	040440	042104	042522		
12985	055214	051523	000			
12986	055217	127	044522	042524	EM41:	.ASCIZ /WRITE LOCK ERROR/
12987	055224	046040	041517	020113		
12988	055232	051105	047522	000122		
12989	055240	051104	053111	020105	EM42:	.ASCIZ /DRIVE TIMING ERROR/
12990	055246	044524	044515	043516		
12991	055254	042440	051122	051117		
12992	055262	000				
12993	055263	116	020117	042503	EM43:	.ASCIZ /NO CERR WITH SOME OTHER ERROR SET/
12994	055270	051122	053440	052111		
12995	055276	020110	047523	042515		
12996	055304	047440	044124	051105		
12997	055312	042440	051122	051117		
12998	055320	051440	052105	000		
12999	055325	104	044522	042526	EM44:	.ASCIZ /DRIVE UNSAFE/
13000	055332	052440	051516	043101		
13001	055340	000105				
13002	055342	042503	051122	051440	EM45:	.ASCIZ /CERR SET BUT NO OTHER ERROR SET/
13003	055350	052105	041040	052125		
13004	055356	047040	020117	052117		
13005	055364	042510	020122	051105		
13006	055372	047522	020122	042523		
13007	055400	000124				
13008						
13009	055402	053126	042040	042111	EM46:	.ASCIZ /VV DID NOT SET WITH PACK ACK/
13010	055410	047040	052117	051440		
13011	055416	052105	053440	052111		
13012	055424	020110	040520	045503		
13013	055432	040440	045503	000		
13014	055437	123	040524	052524	EM47:	.ASCIZ /STATUS VALID SET ON SELECT TO NON-EXISTANT DRIVE/

13015	055444	020123	040526	044514	
13016	055452	020104	042523	020124	
13017	055460	047117	051440	046105	
13018	055466	041505	020124	047524	
13019	055474	047040	047117	042455	
13020	055502	044530	052123	047101	
13021	055510	020124	051104	053111	
13022	055516	000105			
13023	055520	052123	052101	051525	EM48: .ASCIZ /STATUS VALID RESET ON SELECT TO EXISTANT DRIVE/
13024	055526	053040	046101	042111	
13025	055534	051040	051505	052105	
13026	055542	047440	020116	042523	
13027	055550	042514	052103	052040	
13028	055556	020117	054105	051511	
13029	055564	040524	052116	042040	
13030	055572	044522	042526	000	
13031	055577	123	040524	052524	EM49: .ASCIZ /STATUS VALID NOT RESET ON DRIVE RELEASE/
13032	055604	020123	040526	044514	
13033	055612	020104	047516	020124	
13034	055620	042522	042523	020124	
13035	055626	047117	042040	044522	
13036	055634	042526	051040	046105	
13037	055642	040505	042523	000	
13038	055647	105	050130	041505	EM50: .ASCIZ /EXPECTED 2ND INTERRUPT DID NOT OCCUR OR WAS LATE. COMMAND WAS:/
13039	055654	042524	020104	047062	
13040	055662	020104	047111	042524	
13041	055670	051122	050125	020124	
13042	055676	044504	020104	047516	
13043	055704	020124	041517	052503	
13044	055712	020122	051117	053440	
13045	055720	051501	046040	052101	
13046	055726	027105	041440	046517	
13047	055734	040515	042116	053440	
13048	055742	051501	000072		
13049	055746	040503	047116	052117	EM51: .ASCII /CANNOT READ BAD SECTOR FILE/<15><12>
13050	055754	051040	040505	020104	
13051	055762	040502	020104	042523	
13052	055770	052103	051117	043040	
13053	055776	046111	006505	012	
13054	056003	124	051505	044524	.ASCIZ /TESTING ABORTED ON THIS DRIVE/
13055	056010	043516	040440	047502	
13056	056016	052122	042105	047440	
13057	056024	020116	044124	051511	
13058	056032	042040	044522	042526	
13059	056040	000			
13060	056041	101	044514	047107	EM52: .ASCII /ALIGNMENT PACK ON THIS DRIVE./<15><12>
13061	056046	042515	052116	050040	
13062	056054	041501	020113	047117	
13063	056062	052040	044510	020123	
13064	056070	051104	053111	027105	
13065	056076	005015			
13066	056100	042524	052123	047111	.ASCIZ /TESTING ABORTED ON THIS DRIVE./
13067	056106	020107	041101	051117	
13068	056114	042524	020104	047117	
13069	056122	052040	044510	020123	
13070	056130	051104	053111	027105	

13071	056136	000			
13072	056137	124	020117	040515	EM53: .ASCIZ /TO MANY BAD SECTORS. TESTING ABORTED ON THIS DRIVE/
13073	056144	054516	041040	042101	
13074	056152	051440	041505	047524	
13075	056160	051522	020056	042524	
13076	056166	052123	047111	020107	
13077	056174	041101	051117	042524	
13078	056202	020104	047117	052040	
13079	056210	044510	020123	051104	
13080	056216	053111	000105		
13081	056222	042510	042101	051105	EM54: .ASCIZ /HEADER WORD MISCOMPARE/
13082	056230	053440	051117	020104	
13083	056236	044515	041523	046517	
13084	056244	040520	042522	000	
13085	056251	104	052101	020101	EM55: .ASCIZ /DATA WORD MISCOMPARE/
13086	056256	047527	042122	046440	
13087	056264	051511	047503	050115	
13088	056272	051101	000105		
13089					
13090	056276	040503	047116	052117	EM56: .ASCIZ /CANNOT FIND HEADER 0 IN READ AND SORT HEADERS/
13091	056304	043040	047111	020104	
13092	056312	042510	042101	051105	
13093	056320	030040	044440	020116	
13094	056326	042522	042101	040440	
13095	056334	042116	051440	051117	
13096	056342	020124	042510	042101	
13097	056350	051105	000123		
13098	056354	052123	052101	051525	EMSVL: .ASCIZ /STATUS VALID/
13099	056362	053040	046101	042111	
13100	056370	000			
13101	056371	123	046105	041505	EMNZPR: .ASCIZ /SELECT WITH NON-ZERO STATUS PAIR/
13102	056376	020124	044527	044124	
13103	056404	047040	047117	055055	
13104	056412	051105	020117	052123	
13105	056420	052101	051525	050040	
13106	056426	044501	000122		
13107	056432	051127	052111	047111	EMWCS2: .ASCIZ /WRITING COMMAND STATUS REGISTER 2/
13108	056440	020107	047503	046515	
13109	056446	047101	020104	052123	
13110	056454	052101	051525	051040	
13111	056462	043505	051511	042524	
13112	056470	020122	000062		
13113		055050			EMDTPE= EM35 ;DRIVE TYPE ERROR
13114	056474	051104	053111	020105	EMDI: .ASCIZ /DRIVE INTERRUPT/
13115	056502	047111	042524	051122	
13116	056510	050125	000124		
13117		054765			EMDPAR= EM33 ;DRIVE DETECTED DRIVE BUS PARITY ERROR
13118	056514	051104	053111	020105	EMDSC: .ASCIZ /DRIVE STATUS CHANGE/
13119	056522	052123	052101	051525	
13120	056530	041440	040510	043516	
13121	056536	000105			
13122	056540	051104	053111	020105	EMDA: .ASCIZ /DRIVE ATTENTION/
13123	056546	052101	042524	052116	
13124	056554	047511	000116		
13125	056560	047503	052116	047522	EMCCLR: .ASCIZ /CONTROLLER CLEAR/
13126	056566	046114	051105	041440	

13127	056574	042514	051101	000	
13128		050160			EMSELD= OPER00 ;DRIVE SELECT
13129		050206			EMDCLR= OPER04 ;DRIVE CLEAR
13130		050247			EMRCAL= OPER12 ;RECALIBRATE
13131	056601	123	041505	047117	EM2INT: .ASCIZ /SECOND INTERRUPT/
13132	056606	020104	047111	042524	
13133	056614	051122	050125	000124	
13134	056622	042523	045505	052040	EMSKSF: .ASCIZ /SEEK TO SELF/
13135	056630	020117	042523	043114	
13136	056636	000			
13137		050272			EMSK= OPER16 ;SEEK
13138	056637	125	042516	050130	EMUXIT: .ASCIZ /UNEXPECTED INTERRUPT/
13139	056644	041505	042524	020104	
13140	056652	047111	042524	051122	
13141	056660	050125	000124		
13142	056664	047125	041111	051525	EMUR: .ASCIZ /UNIBUS RESET/
13143	056672	051040	051505	052105	
13144	056700	000			
13145	056701	122	051505	052105	EMRSET: .ASCIZ /RESET/
13146	056706	000			
13147		054541			EMDLT= EM26 ;DATA LATE
13148	056707	122	040505	044504	EMRDB: .ASCIZ /READING DATA BUFFER/
13149	056714	043516	042040	052101	
13150	056722	020101	052502	043106	
13151	056730	051105	000		
13152	056733	103	047117	051124	EMCERR: .ASCIZ /CONTROLLER ERROR/
13153	056740	046117	042514	020122	
13154	056746	051105	047522	000122	
13155	056754	052523	051502	051531	EMSCLR: .ASCIZ /SUBSYSTEM CLEAR/
13156	056762	042524	020115	046103	
13157	056770	040505	000122		
13158	056774	052515	052114	050111	EMMI: .ASCIZ /MULTIPLE INTERRUPTS/
13159	057002	042514	044440	052116	
13160	057010	051105	052522	052120	
13161	057016	000123			
13162	057020	040502	020104	042523	DRVABT: .ASCII /BAD SECTORS ON PACK IN AREAS USED BY TEST (CYL 312(8))/<15><12>
13163	057026	052103	051117	020123	
13164	057034	047117	050040	041501	
13165	057042	020113	047111	040440	
13166	057050	042522	051501	052440	
13167	057056	042523	020104	054502	
13168	057064	052040	051505	020124	
13169	057072	041450	046131	031440	
13170	057100	031061	034050	024451	
13171	057106	005015			
13172	057110	042524	052123	047111	.ASCIZ /TESTING ABORTED ON THIS DRIVE/
13173	057116	020107	041101	051117	
13174	057124	042524	020104	047117	
13175	057132	052040	044510	020123	
13176	057140	051104	053111	000105	
13177					
13178					.SBTTL DATA HEADERS FOR ERROR REPORTS
13179	057146	051524	020124	052516	DH001: .ASCIZ /TST NUM ERR PC DRIVE/
13180	057154	020115	051105	020122	
13181	057162	041520	020040	051104	
13182	057170	053111	000105		

13183	057174	051524	020124	052516	DH002A: .ASCIZ /TST NUM ERR PC DRIVE/
13184	057202	020115	051105	020122	
13185	057210	041520	020040	051104	
13186	057216	053111	000105		
13187	057222	045522	051503	020061	DH002B: .ASCIZ /RKCS1 RKCS2 RKDS RKER RKASOF RKDCYL RKDA/
13188	057230	020040	045522	051503	
13189	057236	020062	020040	045522	
13190	057244	051504	020040	020040	
13191	057252	045522	051105	020040	
13192	057260	020040	045522	051501	
13193	057266	043117	020040	045522	
13194	057274	041504	046131	020040	
13195	057302	045522	040504	000	
13196	057307	122	041113	020101	DH002C: .ASCIZ /RKBA RKWC/
13197	057314	020040	051040	053513	
13198	057322	000103			
13199	057324	030101	020060	020040	DH002D: .ASCIZ /A00 B00 A01 B01 A02 B02 A03 B03/
13200	057332	020040	030102	020060	
13201	057340	020040	020040	030101	
13202	057346	020061	020040	020040	
13203	057354	030102	020061	020040	
13204	057362	020040	030101	020062	
13205	057370	020040	020040	030102	
13206	057376	020062	020040	020040	
13207	057404	030101	020063	020040	
13208	057412	020040	030102	000063	
13209	057420	045522	051503	020061	DH003B: .ASCIZ /RKCS1 RKCS2 RKDS RKER RKASOF RKMR1/
13210	057426	020040	045522	051503	
13211	057434	020062	020040	045522	
13212	057442	051504	020040	020040	
13213	057450	045522	051105	020040	
13214	057456	020040	045522	051501	
13215	057464	043117	020040	045522	
13216	057472	051115	000061		
13217	057476	044124	020105	041101	DH005: .ASCIZ /THE ABOVE ARE EXPECTED ERRORS THAT DID NOT SET IN OPERATION:/
13218	057504	053117	020105	051101	
13219	057512	020105	054105	042520	
13220	057520	052103	042105	042440	
13221	057526	051122	051117	020123	
13222	057534	044124	052101	042040	
13223	057542	042111	047040	052117	
13224	057550	051440	052105	044440	
13225	057556	020116	050117	051105	
13226	057564	052101	047511	035116	
13227	057572	000			
13228	057573	124	042510	040440	DH006: .ASCIZ /THE ABOVE ARE UNEXPECTED ERRORS SET IN OPERATION:/
13229	057600	047502	042526	040440	
13230	057606	042522	052440	042516	
13231	057614	050130	041505	042524	
13232	057622	020104	051105	047522	
13233	057630	051522	051440	052105	
13234	057636	044440	020116	050117	
13235	057644	051105	052101	047511	
13236	057652	035116	000		
13237	057655	124	042510	040440	DH007: .ASCIZ /THE ABOVE ARE ERRORS SET IN OPERATION:/
13238	057662	047502	042526	040440	

13239	057670	042522	042440	051122	
13240	057676	051117	020123	042523	
13241	057704	020124	047111	047440	
13242	057712	042520	040522	044524	
13243	057720	047117	000072		
13244	057724	047101	020131	044506	DH005A: .ASCIZ /ANY FIELD WITH ALL ONES MUST BE CONSIDERED INVALID/
13245	057732	046105	020104	044527	
13246	057740	044124	040440	046114	
13247	057746	047440	042516	020123	
13248	057754	052515	052123	041040	
13249	057762	020105	047503	051516	
13250	057770	042111	051105	042105	
13251	057776	044440	053116	046101	
13252	060004	042111	000		
13253	060007	105	051122	051117	DH010: .ASCIZ /ERROR AT COMPLETION OF OPERATION/
13254	060014	040440	020124	047503	
13255	060022	050115	042514	044524	
13256	060030	047117	047440	020106	
13257	060036	050117	051105	052101	
13258	060044	047511	000116		
13259	060050	054105	052120	020104	DH010A: .ASCIZ /EXPTD IS/
13260	060056	020040	051511	000	
13261	060063	122	040505	020104	DH010B: .ASCIZ /READ DATA WITH FORCED CYLINDER OVERFLOW/
13262	060070	040504	040524	053440	
13263	060076	052111	020110	047506	
13264	060104	041522	042105	041440	
13265	060112	046131	047111	042504	
13266	060120	020122	053117	051105	
13267	060126	046106	053517	000	
13268	060133	116	052117	051440	DH011: .ASCIZ /NOT SET AS A RESULT OF/
13269	060140	052105	040440	020123	
13270	060146	020101	042522	052523	
13271	060154	052114	047440	000106	
13272	060162	047516	020124	042522	DH012: .ASCIZ /NOT RESET AS A RESULT OF/
13273	060170	042523	020124	051501	
13274	060176	040440	051040	051505	
13275	060204	046125	020124	043117	
13276	060212	000			
13277	060213	123	052105	040440	DH013: .ASCIZ /SET AS A RESULT OF/
13278	060220	020123	020101	042522	
13279	060226	052523	052114	047440	
13280	060234	000106			
13281	060236	042522	042523	020124	DH014: .ASCIZ /RESET AS A RESULT OF/
13282	060244	051501	040440	051040	
13283	060252	051505	046125	020124	
13284	060260	043117	000		
13285	060263	107	047517	020104	DH015: .ASCIZ /GOOD BAD WORD NUM/
13286	060270	020040	041040	042101	
13287	060276	020040	020040	053440	
13288	060304	051117	020104	052516	
13289	060312	000115			
13290	060314	052523	051502	051531	DH016: .ASCIZ /SUBSYSTEM CLEAR TO RESET SEEK LIMIT ALLOWING HEADS TO RELOAD/
13291	060322	042524	020115	046103	
13292	060330	040505	020122	047524	
13293	060336	051040	051505	052105	
13294	060344	051440	042505	020113	

13295	060352	044514	044515	020124		
13296	060360	046101	047514	044527		
13297	060366	043516	044040	040505		
13298	060374	051504	052040	020117		
13299	060402	042522	047514	042101		
13300	060410	000				
13301	060411	123	046105	041505	DH017:	.ASCIZ /SELECT AFTER RECAL/
13302	060416	020124	043101	042524		
13303	060424	020122	042522	040503		
13304	060432	000114				
13305	060434	042522	040503	020114	DH018:	.ASCIZ /RECAL COMPLETE ATTN AFTER SELECT/
13306	060442	047503	050115	042514		
13307	060450	042524	040440	052124		
13308	060456	020116	043101	042524		
13309	060464	020122	042523	042514		
13310	060472	052103	000			
13311						
13312					.SBTTL	DATA TABLES FOR ERROR REPORTS
13313		060476			.EVEN	
13314	060476	001302	001116	001626	DT003:	.WORD \$TESTN,\$ERRPC,DRVNUM,T.CS1,T.CS2,T.DS,T.ER,T.ASOF,T.MR1
13315	060504	001540	001550	001552		
13316	060512	001554	001556	001566		
13317	060520	001302	001116	001626	DT001:	.WORD \$TESTN,\$ERRPC,DRVNUM
13318	060526				DT002:	
13319	060526				DT004:	
13320	060526				DT005:	
13321	060526				DT006:	
13322	060526				DT007:	
13323	060526	001302	001116	001626	DT010:	.WORD \$TESTN,\$ERRPC,DRVNUM
13324	060534	001540	001550	001552		.WORD T.CS1,T.CS2,T.DS,T.ER,T.ASOF
13325	060542	001554	001556			
13326	060546	001560	001546		.WORD	T.DCYL,T.DA
13327	060552	001544	001542		.WORD	T.BA,T.WC
13328	060556	001202	001204	001206	.WORD	\$REG10,\$REG11,\$REG12,\$REG13,\$REG14,\$REG15,\$REG16,\$REG17
13329	060564	001210	001212	001214		
13330	060572	001216	001220			
13331	060576	001302	001116	001626	DT015:	.WORD \$TESTN,\$ERRPC,DRVNUM
13332	060604	001202	001204	001206	DT015A:	.WORD \$REG10,\$REG11,\$REG12
13333					.SBTTL	DATA FORMATS FOR ERROR REPORTS
13334	060612	000001			DF001:	.WORD 1
13335	060614	003	000		.BYTE	3,0
13336						
13337	060616	000004			DF002:	.WORD 4
13338	060620	000	000		.BYTE	0,0
13339	060622	057174			.WORD	DH002A
13340	060624	003	000		.BYTE	3,0
13341	060626	057222			.WORD	DH002B
13342	060630	007	000		.BYTE	7,0
13343	060632	057307			.WORD	DH002C
13344	060634	002	000		.BYTE	2,0
13345						
13346						
13347	060636	000002			DF003:	.WORD 2
13348	060640	003	000		.BYTE	3,0
13349	060642	057420			.WORD	DH003B
13350	060644	006	000		.BYTE	6,0

13351					
13352	060646	000006		DF004:	.WORD 6
13353	060650	000	000		.BYTE 0,0
13354	060652	000000		DF004A:	.WORD 0
13355	060654	000	000		.BYTE 0,0
13356	060656	057174			.WORD DH002A
13357	060660	003	000		.BYTE 3,0
13358	060662	057222			.WORD DH002B
13359	060664	007	000		.BYTE 7,0
13360	060666	057307			.WORD DH002C
13361	060670	002	000		.BYTE 2,0
13362	060672	057324			.WORD DH002D
13363	060674	010	000		.BYTE 10,0
13364					
13365	060676	000007		DF005:	.WORD 7
13366	060700	000	000		.BYTE 0,0
13367	060702	000000		DF005A:	.WORD 0
13368	060704	000	000		.BYTE 0,0
13369	060706	057174			.WORD DH002A
13370	060710	003	000		.BYTE 3,0
13371	060712	057222			.WORD DH002B
13372	060714	007	000		.BYTE 7,0
13373	060716	057307			.WORD DH002C
13374	060720	002	000		.BYTE 2,0
13375	060722	057324			.WORD DH002D
13376	060724	010	000		.BYTE 10,0
13377	060726	057724			.WORD DH005A
13378	060730	000	000		.BYTE 0,0
13379					
13380	060732	000005		DF006:	.WORD 5
13381	060734	000	000		.BYTE 0,0
13382	060736	000000		DF006A:	.WORD 0
13383	060740	000	000		.BYTE 0,0
13384	060742	057174			.WORD DH002A
13385	060744	003	000		.BYTE 3,0
13386	060746	057222			.WORD DH002B
13387	060750	007	000		.BYTE 7,0
13388	060752	057307			.WORD DH002C
13389	060754	002	000		.BYTE 2,0
13390					
13391	060756	000004		DF007:	.WORD 4
13392	060760	000	000		.BYTE 0,0
13393	060762	000000		DF007A:	.WORD 0
13394	060764	000	000		.BYTE 0,0
13395	060766	057174			.WORD DH002A
13396	060770	003	000		.BYTE 3,0
13397	060772	057420			.WORD DH003B
13398	060774	006	000		.BYTE 6,0
13399					
13400	060776	000004		DF010:	.WORD 4
13401	061000	000	000		.BYTE 0,0
13402	061002	000000		DF010A:	.WORD 0
13403	061004	000	000		.BYTE 0,0
13404	061006	057174			.WORD DH002A
13405	061010	003	000		.BYTE 3,0
13406	061012	060050			.WORD DH010A

13407	061014	002	000		.BYTE	2,0		
13408								
13409	061016	000004			DF011:	.WORD	4	
13410	061020	000	000			.BYTE	0,0	
13411	061022	000000			DF011A:	.WORD	0	
13412	061024	000	000			.BYTE	0,0	
13413	061026	057174				.WORD	DH002A	
13414	061030	003	000			.BYTE	3,0	
13415	061032	057420				.WORD	DH003B	
13416	061034	000006	000000			.WORD	6,0	
13417								
13418	061040	000002			DF015:	.WORD	2	
13419	061042	003	000			.BYTE	3,0	
13420	061044	060263				.WORD	DH015	
13421	061046	003	000			.BYTE	3,0	
13422								
13423	061050	000001			DF016:	.WORD	1	
13424	061052	003	000			.BYTE	3,0	
13425								
13426	061054	000052			BS26:	.BLKW	52	
13427	061200	000052			BS24:	.BLKW	52	
13428		070376				.=70376		
13429	070376	000240				NOP		
13430	070400	005014			SPCHLR:	CLR	(R4)	:CLEAR MEM MAINT REG
13431	070402	005237	001662			INC	INTSET	:COUNT THE INTERRUPT
13432	070406	000240				NOP		
13433	070410	000002			SPCPAR:	RTI		:RETURN
13434	070412	000240				NOP		
13435	070414	001000			!BUFF:	.BLKW	1000	
13436	072414	001000			!BUFF:	.BLKW	1000	
13437		000001			.END			

ABASE = 177440	1967#	2289	2330		
ABORT 053227	9590	12793#			
ABTFAI 044170	9385	11412#			
ACDW1 = 000000	2289	2332			
ACDW2 = 000000	2289				
ACLO = 000010	2062#	10506			
ACLOER= 000100	10508	12377#			
ACPUOP= 000000	2289	2304			
ADDW0 = 000000	2289				
ADDW1 = 000000	2289				
ADDW10= 000000	2289				
ADDW11= 000000	2289				
ADDW12= 000000	2289				
ADDW13= 000000	2289				
ADDW14= 000000	2289				
ADDW15= 000000	2289				
ADDW2 = 000000	2289				
ADDW3 = 000000	2289				
ADDW4 = 000000	2289				
ADDW5 = 000000	2289				
ADDW6 = 000000	2289				
ADDW7 = 000000	2289				
ADDW8 = 000000	2289				
ADDW9 = 000000	2289				
ADEVCT= 000000	2289	2295			
ADEVM = 000000	2289	2331			
ADJCLK 043614	3268	11305#			
AENV = 000000	2289	2300			
AENVM = 000000	2289	2301			
AFATAL= 000000	2289	2292			
AMADR1= 000000	2289	2317			
AMADR2= 000000	2289	2321			
AMADR3= 000000	2289	2324			
AMADR4= 000000	2289	2327			
AMAMS1= 000000	2289	2311			
AMAMS2= 000000	2289	2319			
AMAMS3= 000000	2289	2322			
AMAMS4= 000000	2289	2325			
AMSGAD= 000000	2289	2297			
AMSGLG= 000000	2289	2298			
AMSGTY= 000000	2289	2291			
AMTYP1= 000000	2289	2312			
AMTYP2= 000000	2289	2320			
AMTYP3= 000000	2289	2323			
AMTYP4= 000000	2289	2326			
APASS = 000000	2289	2294			
APRIOR= 000240	1966#	2289			
APTCSU= 000040	9443#	11445			
APTENV= 000001	9142	9399	9441#	9498	11438
APTSIZ= 000200	2803	9440#			
APTSP0= 000100	9401	9442#	11440		
ASWREG= 000000	2289	2302			
ATESTN= 000000	2289	2293			
AUNIT = 000000	2289	2296			
AUSWR = 000000	2289	2303			
AVECT1= 000210	1965#	2289	2328		

IDAE = 002000	2050#													
IDAERR= 002000	3942	3950	4011	12381#										
IE = 000100	2008#	2974	3872	7821	7924	8122	8494							
ILF = 000001	2040#	10500	10581											
ILFERR= 000200	10583	12405#												
INTHLR 034442	2921	3045	5735	9613#										
INTR = 000300	2003#													
INTSET 001662	2697#	2948*	2953	2973*	2978	3745*	3827	3863*	3918*	3981*	5656	7432*	7469*	
	7627*	7688*	7768*	7820*	7857*	7926*	8021	8080*	8123*	8227*	8263*	8360*	8487*	
	8636*	8705*	8793*	8813*	8946*	8968*	9604*	9614*	9720*	9728	9733*	9766	9789	
	9857	9912*	10072*	10077	11357*	13431*								
	1923#	2772*	2773*											
IOTVEC= 000020	2027#													
IR = 000100	9813#	12099												
IWAT1M 035370	9819#	12096												
IWAT1S 035410	9831#	12092												
IWAT11 035450	9828#	12093												
IWAT12 035440	9825#	12094												
IWAT14 035430	9822#	12095												
IWAT15 035420	9849#	12086												
IWAT16 035530	9816#	12097												
IWAT2S 035400	9846#	12087												
IWAT32 035520	9843#	12088												
IWAT48 035510	9840#	12089												
IWAT64 035500	9810#	12098												
IWAT8S 035360	9837#	12090												
IWAT80 035470	9834#	12091												
IWAT96 035460	1956#	9155	9187											
KIPAR0= 172340	1957#													
KIPAR1= 172342	1958#													
KIPAR2= 172344	1959#													
KIPAR3= 172346	1960#													
KIPAR4= 172350	1961#													
KIPAR5= 172352	1962#	11145*												
KIPAR6= 172354	1963#	9203												
KIPAR7= 172356	1945#													
KIPDR0= 172300	1946#													
KIPDR1= 172302	1947#													
KIPDR2= 172304	1948#													
KIPDR3= 172306	1949#													
KIPDR4= 172310	1950#													
KIPDR5= 172312	1951#													
KIPDR6= 172314	1952#													
KIPDR7= 172316	2725#	5664*	5680*	9634*	9724*	10057*	10104*	11314*						
KWLADD 001710	2726#	9721												
KWLVEC 001712	9633#	9722												
LCKHLR 034506	2710#	9731	9762	9859	10055	10102	11312							
LCLKPR= 100000	2716#	3867*	3869	9633*	9761*	9768	9770*	9795*	9862	10073*	10080			
LCLKTK 001674	3308	9938#												
LDCON 036034	1829#	11488	11494	12596	12599									
LF = 000012	9912#	12082												
LOADRK 035734	4071	4229	4257	4354	4382	4430	4463	4509	4529	4561	4582	4615	4643	
LRLoad 035674	4676	4702	4737	4766	4785	4831	4851	4918	4942	4993	5010	5072	5090	
	5141	5158	5218	5237	5292	5309	5354	5373	5429	5447	5498	5551	5623	
	5757	5780	5795	5841	5864	5883	5927	5955	5969	6012	6036	6051	6097	
	6121	6136	6195	6252	6310	6366	6422	6468	6496	6620	6635	6693	6708	

S.HDFL= 000200	2135#												
S.HDHM= 000040	2119#	7918	8481										
S.ICYL= 000040	2105#												
S.ILF = 000400	2108#												
S.LIMD= 020000	2141#												
S.LOAD= 010000	2126#												
S.MHD = 000400	2136#												
S.NMOV= 010000	2140#												
S.OFF = 002000	2097#												
S.PAR = 001000	2109#												
S.PIP = 020000	2100#												
S.PLO = 004000	2139#												
S.REV = 004000	2125#												
S.RTZ = 020000	2127#												
S.SECT= 000020	2132#												
S.SKI = 002000	2110#												
S.SPIN= 010000	2099#												
S.SPLS= 010000	2112#												
S.SPOK= 001000	2123#												
S.TYPE= 000400	2095#												
S.UNLD= 040000	2128#												
S.UNS = 040000	2114#												
S.VV = 000100	2093#												
S.WCLK= 000040	2133#												
S.WGAT= 000100	2134#												
S.WLE = 004000	2111#												
S.WRL = 004000	2098#												
S.XDOK= 000020	2118#												
S.XERR= 001000	2137#												
TBITVE= 000014	1920#												
TCHKOP= 104421	3088	3154	3224	3238	3291	3300	3330	3346	3381	3513	3576	3603	3646
	3694	3782	3833	4082	4248	4273	4299	4377	4410	4456	4490	4526	4579
	4640	4699	4713	4752	4782	4797	4848	4871	4934	4952	5008	5021	5084
	5101	5156	5170	5234	5249	5307	5321	5370	5390	5444	5461	5773	5792
	5807	5857	5880	5895	5943	5967	6028	6048	6063	6114	6133	6148	6207
	6264	6322	6378	6434	6480	6549	6569	6632	6647	6705	6720	6781	6845
	6935	6965	7008	7023	7065	7080	7201	7216	7330	7347	7394	7425	7451
	7470	7496	7521	7611	7629	7635	7650	7675	7690	7695	7710	7733	7765
	7771	7786	7861	7915	8004	8048	8139	8172	8198	8244	8260	8268	8283
	8357	8364	8379	8394	8437	8478	8513	8531	8582	8653	8722	8790	8804
	8956	8976	9004	11354	11361	11389	12084#						
TCHKWE= 104422	3207	3425	3469	3534	3663	3713	3941	3949	4010	4397	4478	4541	4594
	4655	5509	5562	5698	6508	6804	6880	7100	7238	7813	7891	7965	8040
	8113	8320	8454	12085#									
TGETRK= 104420	3074	3124	3393	3610	3746	3761	3813	3854	3882	4277	8665	8732	8822
	9864	10084	10130	10230	10429	12083#							
TIMCNT 001654	2694#	9764*	9793*	9796*	9855*	10071*							
TKVEC = 000060	1927#	11660*	11661*										
TLOADR= 104417	3069	3119	3201	3233	3287	3296	3327	3342	3377	3421	3465	3510	3530
	3565	3572	3599	3642	3652	3690	3700	3742	3778	3808	3850	3913	3932
	3976	4006	4079	4244	4269	4373	4390	4453	4471	4522	4537	4575	4590
	4636	4651	4695	4709	4748	4778	4793	4844	4867	4930	4949	5004	5018
	5080	5097	5152	5166	5230	5245	5303	5317	5366	5386	5440	5458	5506
	5559	5660	5769	5788	5803	5853	5876	5891	5939	5963	5977	6024	6044
	6059	6109	6129	6144	6203	6260	6318	6374	6430	6476	6504	6545	6628
	6643	6701	6716	6777	6800	6841	6876	6931	6961	7004	7019	7061	7076

TST4	004616	3279#	9315											
TST40	014470	5349#	9343											
TST41	014662	5424#	9344											
TST42	015056	5493#	9345											
TST43	015262	5546#	9346											
TST44	015466	5609#	9347											
TST45	016326	5752#	9348											
TST46	016532	5836#	9349											
TST47	016736	5922#	9350											
TST5	004724	3319#	9316											
TST50	017154	6007#	9351											
TST51	017360	6092#	9352											
TST52	017564	6179#	9353											
TST53	017710	6238#	9354											
TST54	020034	6295#	9355											
TST55	020160	6352#	9356											
TST56	020304	6408#	9357											
TST57	020430	6462#	9358											
TST6	005102	3369#	9317											
TST60	020502	6491#	9359											
TST61	020562	6526#	9360											
TST62	021046	6606#	9361											
TST63	021230	6678#	9362											
TST64	021416	6757#	9363											
TST65	022122	6911#	9364											
TST66	022340	6987#	9365											
TST67	022444	7043#	9366											
TST7	005230	3412#	9318											
TST70	023066	7179#	9367											
TST71	023510	7309#	9368											
TST72	023620	7371#	9369											
TST73	024466	7576#	9370											
TST74	025160	7747#	9371											
TST75	025402	7839#	9372											
TST76	025660	7945#	9373											
TST77	025740	7983#	9374											
TWAT1M=	104440	12099#												
TWAT1S=	104435	12096#												
TWAT11=	104431	4080	4245	4374	4454	4523	4576	4637	4696	4749	4779	6778	6932	7391
		12092#												
TWAT12=	104432	12093#												
TWAT14=	104433	12094#												
TWAT15=	104434	6477	6505	6546	7422	7592	7666	7724	8488	11386	12095#			
TWAT16=	104423	3071	3121	3202	3235	3288	3297	3328	3343	3378	3422	3466	3531	3573
		3600	3643	3660	3691	3708	3743	3779	3809	3851	3875	3915	3933	3978
		4007	4270	7467	7625	7647	7686	7707	7762	7783	7798	7854	7873	7888
		7912	7962	8078	8093	8137	8169	8225	8241	8257	8280	8303	8317	8354
		8376	8406	8451	8475	8510	8633	8645	8702	8714	8788	8301	8944	8953
		8966	8973	8988	11351	12086#								
TWAT2S=	104436	7810	8814	12097#										
TWAT32=	104424	4391	4472	4538	4710	4794	4868	4950	5019	5098	5167	5246	5318	5387
		5459	5789	5804	5877	5892	5964	5978	6644	6801	6842	6877	6962	7020
		7077	7097	7213	7235	7344	8111	8195	8265	8391	12087#			
TWAT48=	104425	4591	4652	6045	6060	6130	6145	7438	7518	8037	12088#			
TWAT64=	104426	6717	8301	8434	8528	8579	12089#							
TWAT8S=	104437	3754	3920	3983	7492	7822	7858	7927	8125	8229	8361	8496	8948	8969

\$SIZE	032234	2830	9165#															
\$SIZEX	032522	9207	9217#															
\$STUP =	177777	2744#																
\$SVLAD	033050	9262	9301#															
\$SWR =	167400	1797#	1808	1812	1813	1814	1815	1816	1817	1818	1819	2279	2280	2281				
		2781	2782	2784	2785	2939	2970	3035	3280	3320	3370	3413	3457	3635				
		3683	3735	3801	3843	3904	3969	4043	4222	4349	4425	4504	4556	4610				
		4670	4729	4826	4901	4982	5050	5131	5201	5281	5350	5425	5494	5547				
		5610	5753	5837	5923	6008	6093	6180	6239	6296	6353	6409	6463	6492				
		6527	6607	6679	6758	6912	6988	7044	7180	7310	7372	7577	7748	7840				
		7946	7984	8064	8156	8212	8340	8547	8614	8684	8758	8924	9099	9105				
		9132	9138	9140	9238	9239	9240	9241	9242	9248	9265	9267	9268	9281				
		9282	9283	9290	9291	9292	9304	9307	9310	9450	9451	9452	9453	9454				
		9463	9470	9505	9509	9521	12028											
\$SWREG	001320	2302#	2805															
\$SWRMK=	000000	1819	1820	9242	9243	9271												
\$SWOBT	033122	9277	9311#	9871	10133	10233	10860											
\$TERM =	000102	12100#																
\$TESTN	001302	2293#	9302*	9532*	9533*	9869	10131	10231	10858	11333*	13314	13317	13323	13331				
\$TIMES	001262	2279#	2781*	2939*	2970*	3035*	3280*	3320*	3370*	3413*	3457*	3635*	3683*	3735*				
		3801*	3843*	3904*	3969*	4043*	4222*	4349*	4425*	4504*	4556*	4610*	4670*	4729*				
		4826*	4901*	4982*	5050*	5131*	5201*	5281*	5350*	5425*	5494*	5547*	5610*	5753*				
		5837*	5923*	6008*	6093*	6180*	6239*	6296*	6353*	6409*	6463*	6492*	6527*	6607*				
		6679*	6758*	6912*	6988*	7044*	7180*	7310*	7372*	7577*	7748*	7840*	7946*	7984*				
		8064*	8156*	8212*	8340*	8547*	8587*	8614*	8684*	8758*	8924*	9105*	9290*	9297				
		9300*	9310															
\$TKB	001146	2238#	11641	11662	11673	11698	11726	11753										
\$TKCNT	045136	11642#	11657*	11687	11704*	11818	11820*											
\$TKINT	045146	2762	3865	8658	11657#	11678	11739											
\$TKQEN=	045145	11646#	11712	11823														
\$TKQIN	045140	11643#	11658*	11659	11710*	11711*	11712	11714*										
\$TKQOU	045142	11644#	11659*	11821	11822*	11823	11825*											
\$TKQSR	045144	11645#	11658	11714	11825													
\$TKS	001144	2237#	5665*	5681*	5737*	11641	11663*	11694*	11696	11702*	11724	11740*	11750	11762*				
		11782*																
\$TKSRV	045216	11660	11673#															
\$TMP0	001222	2263#	2873*	2875	2877	2884*	2886	2889	2901*	2903	2905	2907*	2908*	2909*				
		2910*	2911*	2913	10219*	10223*												
\$TMP1	001224	2264#	8769*	8847*	8872	8883	8905*	8926*	9031*	9056	9067	9089*	10645	11026*				
		11028*	11030	11044*	11060	11061	11065	11067*										
\$TMP10	001242	2271#	10419*	10599	10701	10831												
\$TMP11	001244	2272#	10420*	10680	10825													
\$TMP12	001246	2273#	10421*	10660	10820													
\$TMP13	001250	2274#	10423*	10427*	10457	10596	10614	10658	10678	10699	10750	10805						
\$TMP14	001252	2275#	10773*	10792	10796*	10798	10802*	10823	10827*	10829	10833*							
\$TMP15	001254	2276#	10774*	10779*														
\$TMP16	001256	2277#	10002*	10007*														
\$TMP17	001260	2278#	10004*	10012	10014*													
\$TMP2	001226	2265#	10646															
\$TMP3	001230	2266#	10158*	10172*	10187													
\$TMP4	001232	2267#	3066*	3095	3099*													
\$TMP5	001234	2268#	10883*	10893*	10900													
\$TMP6	001236	2269#	10880*	10901														
\$TMP7	001240	2270#	3187*	3257*	3262	10881*	10902											
\$TN =	000111	1798#	1808	2931	2939#	2962	2970#	2992	3035#	3272	3280#	3309	3320#	3361				
		3370#	3405	3413#	3438	3457#	3624	3635#	3671	3683#	3721	3735#	3792	3801#				

MSG	2931#	2933	2962#	2964	2991#	2994	3272#	3274	3309#	3311	3361#	3363	3405#	3407	3438#
	3440	3624#	3626	3671#	3673	3721#	3723	3792#	3794	3835#	3837	3893#	3895	3959#	3961
	4018#	4020	4212#	4214	4332#	4334	4414#	4416	4494#	4496	4547#	4549	4599#	4601	4660#
	4662	4719#	4721	4816#	4818	4890#	4892	4971#	4973	5039#	5041	5120#	5122	5190#	5192
	5270#	5272	5341#	5343	5414#	5416	5486#	5488	5539#	5541	5593#	5595	5744#	5746	5827#
	5829	5914#	5916	5998#	6000	6083#	6085	6168#	6170	6227#	6229	6284#	6286	6341#	6343
	6397#	6399	6453#	6455	6482#	6484	6517#	6519	6598#	6600	6669#	6671	6743#	6745	6903#
	6905	6979#	6981	7025#	7027	7161#	7163	7299#	7301	7352#	7354	7559#	7561	7737#	7739
	7829#	7832	7937#	7939	7971#	7974	8052#	8055	8144#	8146	8200#	8202	8325#	8327	8534#
	8536	8606#	8608	8675#	8677	8742#	8744	8908#	8910						
MULT NEWTST	1#	1930#													
	1#	1930#	2931	2962	2992	3272	3309	3361	3405	3438	3624	3671	3721	3792	3835
	3893	3959	4018	4212	4332	4414	4494	4547	4599	4660	4719	4816	4890	4971	5039
	5120	5190	5270	5341	5414	5486	5539	5593	5744	5827	5914	5998	6083	6168	6227
	6284	6341	6397	6453	6482	6517	6598	6669	6743	6903	6979	7025	7161	7299	7352
	7559	7737	7830	7937	7972	8053	8144	8200	8325	8534	8606	8675	8742	8908	
OPCHK	2744#	4848	4871	4934	4952	5008	5021	5084	5101	5156	5170	5234	5249	5307	5321
	5444	5461	5773	5792	5807	5857	5880	5895	5943	5967	6028	6048	6063	6114	6133
	6148	6207	6264	6322	6378	6434	6480	6549	6632	6647	6705	6720	6781	6845	6935
	6965	7008	7023	7065	7080	7201	7216	7330	7347	7394	7425	7450	7470	7495	7521
	7610	7629	7635	7650	7674	7690	7695	7710	7732	7765	7771	7786	7861	7915	8004
	8047	8139	8172	8198	8244	8260	8268	8283	8357	8364	8379	8394	8437	8478	8513
	8531	8582	8652	8721	8790	8804	8955	8975	9004	11354	11361	11389			
POP	1#	1930#	9433	9434	9903	10037	10235	11103	11122	11126	11290	11547	11936	11985	12017
	12018														
PUSH REPORT RESDC	1#	1930#	9394	9396	9417	9892	10024	10209	10992	11003	11506	11910	11965	11998	12004
	1#	1930#													
RKLOAD	2741#	4879	4960	5029	5110	5179	5259	5330	5475	5816	5904	5987	6072	6157	6216
	6273	6331	6387	6443	6581	6656	6729	6854	6892	7532					
	2743#	4844	4867	4930	4949	5004	5018	5080	5097	5152	5166	5230	5245	5303	5317
	5440	5458	5506	5559	5769	5788	5803	5853	5876	5891	5939	5963	5977	6024	6044
	6059	6109	6129	6144	6203	6260	6318	6374	6430	6476	6504	6628	6643	6701	6716
	6777	6800	6841	6876	6931	6961	7004	7019	7061	7076	7096	7197	7212	7234	7326
	7343	7390	7421	7466	7517	7624	7646	7685	7706	7761	7782	7797	7809	7853	7872
	7887	7911	8000	8036	8077	8092	8136	8168	8224	8240	8256	8279	8316	8353	8375
	8390	8405	8450	8473	8509	8527	8578	8632	8701	8787	8800	8943	8952	8965	8972
	8987	11350	11385												
SCOPE	1825#	2938	2969	3034	3279	3319	3369	3412	3456	3634	3682	3734	3800	3842	3903
	3968	4042	4221	4348	4424	4503	4555	4609	4669	4728	4825	4900	4981	5049	5130
	5200	5280	5349	5424	5493	5546	5609	5752	5836	5922	6007	6092	6179	6238	6295
	6352	6408	6462	6491	6526	6606	6678	6757	6911	6987	7043	7179	7309	7371	7576
	7747	7839	7945	7983	8063	8155	8211	8339	8546	8586	8613	8683	8757	8923	9103
SETPRI SETTRA	1#	1930#	11814												
	12058#	12067	12068	12069	12070	12072	12074	12075	12076	12077	12078	12079	12080	12081	12082
	12083	12084	12085	12086	12087	12088	12089	12090	12091	12092	12093	12094	12095	12096	12097
	12098	12099													
SETUP SKIP SLASH SPACE STARS	1#	1930#	2764												
	1#	1930#													
	1#	1930#													
	1930#														
	1#	1930#	2188	2190	2197	2210	2285	2288	2931	2937	2962	2968	2992	3033	3272
	3278	3309	3318	3361	3368	3405	3411	3438	3455	3624	3633	3671	3681	3721	3733
	3792	3799	3835	3841	3893	3902	3959	3967	4018	4041	4212	4220	4332	4347	4414
	4423	4494	4502	4547	4554	4599	4608	4660	4668	4719	4727	4816	4824	4890	4899
	4971	4980	5039	5048	5120	5129	5190	5199	5270	5279	5341	5348	5414	5423	5486
	5492	5539	5545	5593	5608	5744	5751	5827	5835	5914	5921	5998	6006	6083	6091

CZR6KFO RK6 FCTNL CTRL DIAG
CZR6KF.P11 19-JUN-80 14:04

MACY11 30A(1052) 19-JUN-80 14:14 M 5 PAGE 273
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0270

.\$SAVE	1#	1797#	11947
.\$SB2D	1#		
.\$SB2O	1#		
.\$SCOP	1#	1797#	9232
.\$SIZE	1#	1797#	9149
.\$SUPR	1#		
.\$STRAP	1#	1797#	12035
.\$TYPB	1#		
.\$TYPD	1#	1797#	11494
.\$TYPE	1#	1797#	11415
.\$TYPO	1#	1797#	11561
.\$4OCA	1#	1797#	2154
.\$1170	1#		

. ABS. 0744'4 000

ERRORS DETECTED: 0

DSKZ:CZR6KF.BIN,DSKZ:CZR6KF.LST/CRF/SOL/NL:TOC=DSKM:CZR6KF.SML,CZR6KF.P11
RUN-TIME: 88 133 11 SECONDS
RUN-TIME RATIO: 454/233=1.9
CORE USED: 35K (69 PAGES)