

RK611
RK06, RK07

RK611/06 PERF EXER
CZR6PD0

AH-9145D-MC
FICHE 1 OF 2

JUL 1982
COPYRIGHT © 76-82
MADE IN USA



The main body of the document is a large grid of approximately 15 columns and 20 rows of data. Each cell in the grid contains a small, dense table or set of data points, likely representing performance metrics or exercise results. The text is very small and difficult to read, but the overall structure is a regular grid of data blocks.

RK611
RK06, RK07 •

RK611/06 PERF EXER
CZR6PDO

AH-9145D-MC
FICHE 2 OF 2

JUL 1982
COPYRIGHT © 76-82
MADE IN USA



The left side of the page contains a grid of approximately 12 columns and 15 rows of small, illegible data tables. Each cell in the grid appears to contain a small table or set of data points, but the text is too small to be read. The tables are arranged in a regular, repeating pattern across the left half of the page.



.REM @

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

IDENTIFICATION

PRODUCT CODE: AC-9144D-MC
PRODUCT NAME: CZR6PDO RK611/06/07 PERF EXER
DATE: JANUARY 1982
MAINTAINER: STORAGE SYSTEM SOFTWARE TEST APPLICATION
AUTHOR: B. T. LEBLANC

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

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

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

COPYRIGHT (C) 1976,1982 BY DIGITAL EQUIPMENT CORPORATION

CZR6PDO RK611/06 PERF EXEC
SYSMAC.SML 22-FEB-82 15:23

MACY11 30(1046) 23-FEB-82 08:29 C 1
PAGE 1

SEQ 0002

1

43
44
45
46
47
48
49
50

REVISION HISTORY

REVISION

CHANGES

DATE

CZR6PDO

IMPLEMENTED XXDP LOAD MEDIA OPTION
FIXED TO WORK UNDER ACT

JAN 82

TABLE OF CONTENTS

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
95
96
97
98
99

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PREREQUISITS
- 3.0 OPERATING PROCEDURES
 - 3.1 LOADING PROCEDURES
 - 3.2 STARTING PROCEDURE
 - 3.3 SYSTEM PARAMETERS
 - 3.4 SWITCHES
 - 3.4.1 SWITCH 15
 - 3.4.2 SWITCH 14
 - 3.4.3 SWITCH 13
 - 3.4.4 SWITCH 12
 - 3.4.5 SWITCH 11
 - 3.4.6 SWITCH 10
 - 3.4.7 SWITCH 9
 - 3.4.8 SWITCH 8
 - 3.4.9 SWITCH 7
 - 3.4.10 SWITCH 6
 - 3.4.11 SWITCH 5
 - 3.4.12 SWITCH 4
 - 3.4.13 SWITCH 3
 - 3.4.14 SWITCH 2
 - 3.4.15 SWITCH 1
 - 3.4.16 SWITCH 0
 - 3.5 RUN TIME
- 4.0 OPERATING PROCEDURE
 - 4.1 OPERATOR COMMANDS
 - 4.2 DRIVE PARAMETER (PER EACH DRIVE)
 - 4.2.1 FORMS FOR PACK AREA EXCLUSIONS
 - 4.3 DIAGNOSTIC DUMP
- 5.0 PROGRAM DESCRIPTION
- 6.0 PRINT OUTS
 - 6.1 PERFORMANCE SUMMARY TYPEOUT
 - 6.2 ERROR REPORTS
- APPENDIX A - DATA WRITTEN ON PACKS

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153

1.0 ABSTRACT

THE RK06-RK07 PERFORMANCE EXERCISER PROGRAM WILL EXERCISE IN A RANDOM OVERLAPPED MANNER 1 TO 8 RK06-RK07 DISK DRIVES ATTACHED TO THE SAME RK06-RK07 UNIBUS CONTROLLER IN A STAND ALONE MODE.

DRIVES UNDER TEST CAN BE ADDED TO OR DROPPED FROM THE TESTING SEQUENCE BY OPERATOR COMMAND.

AT ANY GIVEN POINT IN TIME, THE NEXT DRIVE ON WHICH AN OPERATION IS TO BE INITIATED IS CHOSEN RANDOMLY FROM AMONG THE RK06 DRIVES UNDER TEST NOT HAVING ANY ERROR CONDITIONS PRESENT AND NOT CURRENTLY UNDERGOING ERROR RECOVERY. THE COMMAND IS THEN CHOSEN RANDOMLY FROM THE FOLLOWING SET OF COMMANDS:

READ
WRITE
WRITE FOLLOWED BY WRITE CHECK

THEN THE CYLINDER, TRACK, SECTOR, WORD COUNT, AND DATA ARE DETERMINED RANDOMLY. THE DATA WRITTEN IS RANDOMLY SELECTED FROM A SET OF FIXED DATA PATTERNS. TO OPTIMIZE THE THROUGHPUT ON THE RK06 CONTROLLER EACH DATA TRANSFER COMMAND IS TRANSLATED TO AN EXPLICIT SEEK COMMAND FOLLOWED BY THE DATA TRANSFER COMMAND.

DRIVE ERRORS ARE QUEUED UP AS THEY OCCUR. WHILE ERROR RECOVERY IS BEING PROCESSED ON ONE DRIVE ALL OTHER DRIVES HAVING NO ERRORS WILL BE EXECUTING ORDERS GENERATED RANDOMLY AS DESCRIBED ABOVE. EACH DRIVE ERROR IS PROCESSED ON A FIRST-IN-FIRST-OUT BASIS. THIS MEANS THAT A DRIVE ERROR WILL BE REPORTED AND ERROR RECOVERY COMPLETELY PROCESSED BEFORE THE NEXT DRIVE ERROR IS REPORTED AND ERROR PROCESSING HAS BEGUN. ERROR RECOVERY IS PROCESSED AS DESCRIBED IN THE RK06 DISK DRIVE SPECIFICATION.

REPORTING OF SYSTEM ERRORS SUCH AS CONTROLLER PROBLEMS WILL BE IMMEDIATE UPON OCCURRENCE, NOT DEFERRED, PERHAPS CAUSING PREMATURE PROGRAM TERMINATION.

PERFORMANCE STATISTICS ARE KEPT ON EACH DRIVE. THESE STATISTICS INCLUDE BOTH OPERATION COUNTS AND ERROR SUMMARY INFORMATION. (SEE SECTION 6.1) THESE STATISTICS WILL BE INITIALIZED WHEN TESTING BEGINS ON A DRIVE. AT ANY TIME AFTER TESTING BEGINS ON THE DRIVE(S) UNDER PERFORMANCE EVALUATION, THE OPERATOR CAN DEMAND THESE PERFORMANCE STATISTICS. IF A REAL TIME CLOCK IS AVAILABLE, THE OPERATOR HAS THE OPTION OF HAVING THE PROGRAM GIVE PERIODIC PERFORMANCE SUMMARIES.

ONCE A PARTICULAR DRIVE HAS AN UNRECOVERABLE ERROR (OTHER THAN A DATA TRANSFER AND SEEK INCOMPLETE), THAT DRIVE WILL BE AUTOMATICLY DEASSIGNED.

ONCE A PARTICULAR DRIVE HAS EXCEEDED AN ERROR THRESHOLD (DATA TRANSFER OR SEEK) SPECIFIED FOR A DRIVE, THAT DRIVE WILL BE AUTOMATICLY DEASSIGNED UNLESS SWITCH 4 IS SET.

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

ONCE A PARTICULAR DRIVE HAS EXCEEDED THE MAXIMUM NUMBER OF PERMITTED OPERATIONS FOR THAT DRIVE, THE DRIVE WILL BE AUTOMATICALLY DESELECTED UNLESS SWITCH 5 IS SET.

2.0 REQUIREMENTS

2.1 EQUIPMENT

- PDP-11 WITH AT LEAST 16K OF MEMORY
- CONSOLE TERMINAL
- DECTAPE, PAPER TAPE READER; OR DECDISK
- RK611 CONTROLLER
- 1 TO 8 RK06-RK07 DISK DRIVE WITH FORMATTED PACK (16 BIT FORMAT)
- REAL TIME CLOCK (KW11-P OR KW11-L OPTIONAL) FOR INTERVAL PERFORMANCE REPORTING AND TIME ASSOCIATIONS WITH ERROR REPORTS

2.2 PRELIMINARY PROGRAMS

THE RK06-RK07 SUBSYSTEM IS ASSUMED TO BE BASICALLY OPERATIONAL AND FREE OF HARD FAULTS. THE FOLLOWING RK06 PROGRAMS ARE ASSUMED TO BE RUNNING WITHOUT ERROR BEFORE THIS PROGRAM IS RUN:

- RK611 DISKLESS CONTROLLER DIAGNOSTICS
- RK06-RK07 DRIVE DIAGNOSTICS
- RK611 FUNCTIONAL CONTROLLER DIAGNOSTIC
- RK06-RK07 SUBSYSTEM VERIFICATION
- RK06-RK07 PACK FORMATTER (IF PACKS ARE UNFORMATTED)

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

3.0 OPERATING PROCEDURE

3.1 LOADING PROCEDURE

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

3.2 STARTING PROCEDURES

AFTER THE PROGRAM IS LOADED, THE OPERATOR STARTS THE PROGRAM AT ONE OF THE FOLLOWING LOCATIONS:

<u>LOCATION</u>	<u>FUNCTION</u>
-----------------	-----------------

200	INITIALIZE ALL DRIVE STATISTIC TABLES, THE RANDOM NUMBER SEED, AND USE DEFAULT SYSTEM PARAMETERS. THE PROGRAM WILL THEN DO AUTO-SIZING OF ALL DRIVES ON THE BUS & BEGIN BY FIRST WRITING THOSE DRIVES & THEN BEGIN PERFORMANCE TESTING.
-----	---

204	RESTART PROGRAM. IF DRIVE IS UNDER TEST, CONTINUE TESTING DRIVE AND DO NOT DESTROY STATISTICS. IF PACK IS BEING WRITTEN, START WRITING PACK FROM THE BEGINNING. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.
-----	--

214	START PROGRAM AND ALTER SYSTEM PARMETERS. THE PROGRAM WILL THEN IDENTIFY ITSELF AND ASK FOR THE PARAMETERS AS DESCRIBED IN SECTION 3.3.2. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.
-----	--

240	INITIALIZE ALL DRIVE STATISTIC TABLES, THE RANDOM NUMBER SEED, AND USE DEFAULT SYSTEM PARAMETERS. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.
-----	--

NOTE: IF A POWER FAIL OCCURS, THE PROGRAM WILL AUTOMATICALLY PERFORM THE RESTART FUNCTION.

THE PROGRAM IS NOW IN AN IDLE LOOP WAITING FOR THE INITIATION OF RK06 DRIVE TESTING.

THE OPERATOR THEN STARTS PERFORMANCE EVALUATION OF THE RK06 DRIVE(S) BY TYPING A CONTROL-C <C> FOLLOWED BY A TN OR PN AS DESCRIBED IN SECTION 4.1. IF THE PACK DOES NOT HAVE THE PROPER RANDOMLY CHOSEN

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

PREDETERIMED TEST PATTERNS, THE OPERATOR MUST USE THE WN COMMAND AS DESCRIBED IN SECTION 4.1 BEFORE INITIATING THE PERFORMANCE EVALUATION SEQUENCE ON THAT DRIVE.

3.3 SYSTEM PARAMETERS

3.3.1 DIRECT MEMORY ALTERATION

AS PART OF THE PROGRAM START-UP PROCEDURE, THE FOLLOWING SYSTEM PARAMETERS WILL BE LOADED BY THE OPERATOR OR DEFAULTED BY THE PROGRAM. THE OPERATOR MUST PHYSICALLY ALTER THE APPROPRIATE MEMORY LOCATIONS TO CHANGE THESE VALUES.

PARAMETER	TAG	DEFAULT VALUE
- KW11-P STATUS REGISTER ADDRESS	\$LKCSR	172540
- KW11-P COUNTER BUFFER ADDRESS	\$LKCSB	172542
- KW11-P VECTOR ADDRESS	\$LPVEC	104
- KW11-L STATUS REGISTER ADDRESS	\$LKS	177506
- KW11-L VECTOR ADDRESS	\$LLVEC	100
- SYSTEM POWER (HERTZ)	HZ	60(DECIMAL)

3.3.2 PROGRAM MODIFIED PARAMETERS

THE FOLLOWING PARAMETERS ARE MODIFIED BY A 214 PROGRAM START.

A <CR> WILL DEFAULT CURRENT PARAMETERS. A CONTROL Z <^Z> <CR> WILL DEFAULT ALL THE REST OF THE PARAMETERS. A CONTROL C <^C> <CR> WILL RETURN TO FIRST PARAMETER ENTRY REQUEST FOR THIS DRIVE. A RUBOUT CANCELS THE LAST CHARACTER TYPED. A CONTROL U <^U> CANCELS LINE BEING TYPED. ALL ENTRIES MUST BE TERMINATED WITH CARRIAGE RETURN.

299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340

PARAMETER

DEFAULT VALUE

- CPU IDENTIFIER (OCTAL NUMBER 0-377 FOR DUAL ACCESS PORT DISCRIMINATION) 0
- OVERLAY OF LOADER BY PROGRAM? (ALLOWS GREATER EXPANSION OF MEMORY FOR BUFFER SPACE IF LOADER IS OVERLAYED) NO
- MAXIMUM DATA BUFFER (SEE NOTE) UP TO 5888 (23 SECTORS) SEE NOTE
- NUMBER OF DATA WORDS COMPARED PER READ (0 - MAXIMUM BUFFER LENGTH) EACH READ COMMAND ISSUED BY THE PROGRAM HAS AN IMPLICIT DATA COMPARISON ASSOCIATED WITH IT VERIFYING THAT THE DATA READ IS ONE OF LEGAL RANDOM PATTERNS. 3 WORDS
- INTERVAL BETWEEN AUTOMATIC PERFORMANCE TYPE-OUTS (EVERY 1-255 MINUTES OR NO AUTOMATIC PERFORMANCE TYPE-OUTS) NO AUTOMATIC PERFORMANCE TYPE-OUTS
- RK611/RK06-RK07 UNIBUS ADDRESS 177440
- RK611/RK06-RK07 VECTOR ADDRESS 210
- RK611/RK06-RK07 PRIORITY 5

NOTE

THE PROGRAM WILL SELECT A MAXIMUM BUFFER SIZE WHICH IS DETERMINED BY THE MEMORY AVAILABLE. THE MAXIMUM BUFFER SIZE ASSIGNED BY THE PROGRAM WILL BE 5888 WORDS. THE DEFAULT VALUE OF THIS PARAMETER WEIGHTS THE OPERATIONS TOWARDS MAXIMIZING DATA TRANSFERS PER UNIT OF TIME. BY CHANGING THIS PARAMETER TO 1 SECTOR (256 WORDS), THE OPERATIONS WILL BE WEIGHTED TOWARDS MAXIMIZING SEKS PER UNIT OF TIME.

341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393

3.4 SWITCHES

3.4.1 SWITCH 15

SW<15> = 1 HALT ON ERROR

IF THIS SWITCH IS SET, THE PROGRAM WILL NOT ISSUE ANOTHER COMMAND TO THE RK06 SUBSYSTEM WHEN AN ERROR OCCURS AND THE PROGRAM WILL HALT IN THE SYSMAC ERROR HANDLER. IF THE OPERATOR DESIRES TO CONTINUE FROM WHERE THE PROGRAM LEFT OFF, THE OPERATOR SIMPLY PRESSES THE CONTINUE SWITCH ON THE CONSOLE PANEL AND THE PROGRAM CONTINUES WITH THE NEXT RANDOMLY GENERATED COMMAND.

3.4.2 SWITCH 14

SW<14> = 0 NORMAL OPERATION

THE OPERATION THE PROGRAM IN REGARD TO DRIVE SELECTION AND COMMAND GENERATION IS NORMAL. NORMAL OPERATION IS TO RANDOMLY SELECT A DRIVE, STORE THE PREVIOUS OPERATION EXECUTED ON THAT DRIVE, AND GENERATE A NEW RANDOM OPERATION TO BE EXECUTED ON THAT DRIVE. THE GENERATED OPERATION INCLUDES RANDOMLY GENERATED VALUES FOR THE FOLLOWING PARAMETERS:

- DRIVE COMMAND
- CYLINDER ADDRESS
- TRACK ADDRESS
- SECTOR ADDRESS
- WORD COUNT
- DATA PATTERN

THESE VALUES ARE STORED AS THE PREVIOUS OPERATION WHEN A NEW OPERATION IS GENERATED. IT IS IMPORTANT TO NOT THAT AT ANY POINT IN TIME THERE IS BOTH A CURRENT OPERATION AND A STORED PREVIOUS OPERATION ASSOCIATED WITH EACH DRIVE UNDER TEST. THE CURRENT OPERATIGN MAY HAVE ALREADY BEEN EXECUTED AND A NEW COMMAND HAS NOT YET BEEN GENERATED OR THE COMMAND IS WAITING TO BE EXECUTED.

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

SW<14> = 1

LOOP ON CURRENT OPERATIONS (TEST)

DO NOT GENERATE NEW RANDOM OPERATION BUT CONTINUE TO EXECUTE CURRENT OPERATIONS.

WHEN SWITCH 14 IS SET THE GENERATION OF A NEW OPERATION IS INHIBITED AND THE PROGRAM OPERATES AS FOLLOWS:

- 1.) A DRIVE IS RANDOMLY SELECTED.
- 2.) A SEEK IS EXECUTED ON THE DRIVE TO THE CYLINDER ADDRESS STORED IN THAT DRIVE'S PREVIOUS OPERATION.
- 3.) THE CURRENT OPERATION FOR THAT DRIVE IS EXECUTED.
- 4.) REPEAT THE ENTIRE PROCESS.

USEAGE

THIS SWITCH CAN BE USED IN CONJUNCTION WITH SWITCH 15 TO ENABLE THE REPETITION OF A RANDOMLY GENERATED OPERATION. THE OPERATOR WOULD START WITH SWITCH 15 SET AND SWITCH 14 RESET. WHEN THE PROGRAM HAS INDICATED THAT AN ERROR HAS OCCURRED BY HALTING. THE CURRENT OPERATION FOR THE DRIVE IN ERROR WILL NOT HAVE BEEN MODIFIED WHEN THE PROGRAM HALTS. THE OPERATOR CAN THEN CAUSE ALL DRIVES TO LOOP ON THEIR INDIVIDUAL CURRENT OPERATIONS BY SETTING SWITCH 14, RESETTING SWITCH 15, AND HITTING CONTINUE.

TO HELP ISOLATE PROBLEMS, THE OPERATOR CAN DROP THE RK06 DRIVE(S) FROM THE TEST SEQUENCE AT ANY TIME BY USING THE DN COMMAND AS DESCRIBED IN SECTION 4.1.

3.4.3 SWITCH 13

SW<13> = 1

INHIBIT ERROR TYPE OUT

3.4.4 SWITCH 12

NO EFFECT.

446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489

3.4.5 SWITCH 11

NO EFFECT.

3.4.6 SWITCH 10

SW<10> = 1

RING TELETYPE BELL IF ERROR

3.4.7 SWITCH 9

NO EFFECT.

3.4.8 SWITCH 8

NO EFFECT.

3.4.9 SWITCH 7

NO EFFECT.

3.4.10 SWITCH 6

SW<6> = 1

INHIBIT AUTOMATIC DESELECT FOR CLEARABLE UNSAFES,
SPEED LOSS, AND AC LOW.

3.4.11 SWITCH 5

SW<5> = 1

INHIBIT AUTOMATIC DRIVE DEASSIGNMENT IF OPERATION
COUNT THRESHOLD IS EXCEEDED.

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

3.4.12 SWITCH 4

SW<4> = 1 INHIBIT AUTOMATIC DRIVE DEASSIGNMENT IF ERROR
THRESHOLD EXCEEDED.

3.4.13 SWITCH 3

SW<3> = 1 DISPLAY ENTIRE SECTOR READ BEFORE STARTING RETRY
SEQUENCE

ALL HARDWARE DETECTED DATA ERRORS WILL BE RETRIED
ON A SECTOR BASIS. IF THIS SWITCH IS SET, THE
DATA SECTOR IN ERROR WILL BE DISPLAYED ENTIRELY
BEFORE THE RETRY SEQUENCE BEGINS ON THE SECTOR IN
ERROR.

3.4.14 SWITCH 2

NO EFFECT.

3.4.15 SWITCH 1

SW<1> = 1 INHIBIT SOFTWARE DATA COMPARISONS

THE IMPLICIT DATA COMPARISONS ASSOCIATED WITH EACH
READ WILL NOT BE PERFORMED WHEN THIS SWITCH IS
SET. THE ONLY DATA CHECKING PERFORMED WILL BE
HARDWARE DATA CHECKING.

3.4.16 SWITCH 0

NO EFFECT.

3.5 RUN TIME

THIS PROGRAM IS DESIGNED TO RUN FOR AN EXTENDED PERIOD OF TIME.
(ESTIMATED RUN TIME WITH PRESENT DEFAULT PARAMETERS IS APPROXIMATELY
250 HOURS PER DRIVE.) MEANINGFUL INFORMATION CAN STILL BE DERIVED BY
RUNNING THIS PROGRAM FOR AN HOUR OR LESS SINCE STATISTICS ARE TAKEN AS
THE PROGRAM IS RUNNING.

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

4.0 OPERATING PROCEDURE

4.1 OPERATOR COMMANDS

AFTER THE PROGRAM IS LOADED AND HAS PRINTED OUT 'READY TO BEGIN PERFORMANCE TESTING', THE OPERATOR CAN ONLY TYPE A CONTROL-G <G> OR A CONTROL-C <C>.

A CONTROL-G <G> IS USED FOR SOFTWARE SWITCH REGISTER MODIFICATION.

UPON RECEIVING A CONTROL-C <C> THE PROGRAM WILL RESPOND WITH 'PLEASE TYPE COMMAND'. THE FOLLOWING TWO CHARACTER COMMANDS FOLLOWED BY CARRIAGE RETURN WILL BE THE ONLY COMMANDS ACCEPTED: TN, PN, DN, SN, WN.

THE SECOND CHARACTER N=0,1,....,7 OR A DESIGNATES WHICH DRIVE IS REFERENCED BY THE COMMAND. IF N=A, ALL ADDRESSABLE DRIVES ARE REFERENCED. THE DESCRIPTION OF EACH OPERATOR COMMAND FOLLOWS:

- TN INITIATE TESTING OF DRIVE N AND USE PREVIOUS PARAMETERS. (DEFAULT PARMETERS ARE USED IF PN COMMAND HAS NOT PREVIOUSLY BEEN ISSUED TO THE DRIVE(S) AT THE BEGINNING OF THEIR TEST SEQUENCE.) IF N=A, INITIATE TESTING ON ALL DRIVES.

- PN THIS COMMAND HAS TWO USES:

1.) CHANGE CURRENT DRIVE PARAMETERS AND INITIATE TESTING ON DRIVE N.

2.) CHANGE CURRENT DRIVE PARAMETERS AND CONTINUE TESTING ON DRIVE N.

IN BOTH CASES THE PROGRAM WILL THEN ASK FOR PARMETERS IN THE ORDER AS DESCRIBED IN SECTION 5.5. A <CR> INDICATES DEFAULT THIS PARAMETER AND A CONTROL-Z <Z> INDICATES THE THE REST OF THE PARAMETERS FOR THIS DRIVE ARE TO BE DEFAULTED.

IF N=A, PERFORM PN COMMAND ON ALL DRIVES.

- DN DROP DRIVE FROM TESTING SEQUENCE AND PRINT PERFORMANCE SUMMARY. IF N=A, DROP ALL CURRENTLY BEING TESTED DRIVES FROM TESTING SEQUENCE

594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646

- SN DEMAND PERFORMANCE SUMMARY ON DRIVE N. IF N=A DEMAND PERFORMANCE SUMMARY ON ALL CURRENTLY BEING TESTED DRIVES.

- WN WRITE AND VERIFY THE RK06 DISK PACK ON DRIVE N WITH A RANDOM SET OF LEGAL PATTERNS. THIS IS USED PRIOR TO THE START OF THE TEST SEQUENCE TO GUARENTEE THAT THE PROPER PREDETERMINED TEST PATTERNS HAVE BEEN WRITTEN ON THE PACK. (SEE APPENDIX A FOR DETAILS) AFTER THE PACK HAS BEEN COMPLETELY WRITTEN TESTING WILL BEGIN USING THE PREVIOUS PARAMETERS AS DESCRIBED IN THE TN COMMAND.

WRITING OF THE PACKS WILL TAKE 4 TO 5 MINUTES PER PACK WITH MAXIMUM BUFFER SIZE. THE WRITING OF THE PACK MAY TAKE LONGER IF THE MAXIMUM BUFFER SIZE SPECIFIED IS SMALL. RANDOM EXERCISE CAN BE DONE ON SOME DRIVES WHILE EXERCISE ON OTHER DRIVES ARE IN PROGRESS.

NOTE: IF AN ERROR OCCURS DURING THE DRIVE ASSIGNMENT SEQUENCE OR WHILE WRITING THE RANDOM SET OF LEGAL PATTERNS, THE ERROR WILL BE REPORTED AND TESTING WILL NOT START.

EXAMPLES:

COMMAND SEQUENCE	ACTION TAKEN
TA<CR>	INITIATE TESTING ON ALL DRIVES ON SYSTEM
T1<CR>	INITIATE TESTING ON DRIVE 1
T3<CR>	INITIATE TESTING ON DRIVE 3 (TESTING STILL CONTINUES ON DRIVE 1)
SA<CR>	GATHER STATISTICS ON ALL DRIVES UNDER TEST.
DA<CR>	DROP ALL DRIVES UNDER TEST.

4.2 DRIVE PARAMETERS (PER EACH DRIVE)

A SET OF THESE PARAMETERS EXISTS FOR EACH DRIVE. A TN COMMAND WILL UTILIZE THE PREVIOUSLY ESTABLISHED PARAMETERS FOR THE DRIVE(S). A PN COMMAND WILL ALLOW THE OPERATOR TO ALTER ALL THE PARAMETERS FOR THE DRIVE(S).

A <CR> WILL DEFAULT CURRENT PARAMETERS. A CONTROL Z <^Z> <CR> WILL DEFAULT ALL THE REST OF THE PARAMETERS. A CONTROL C <^C> <CR> WILL RETURN TO FIRST PARAMETER ENTRY REQUEST. FOR THIS DRIVE. A RUBOUT CANCELS THE LAST CHARACTER TYPED. A CONTROL U <^U> CANCELS LINE BEING TYPED. ALL ENTRIES MUST BE TERMINATED WITH CARRIAGE RETURN.

647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691

PARAMETER

DEFAULT VALUE (RK06/RK07)

- | PARAMETER | DEFAULT VALUE (RK06/RK07) |
|--|---------------------------|
| - MAXIMUM CYLINDER FOR START OF DATA TRANSFER
0 - 632/1456 | 632/1456 |
| - MINIMUM CYLINDER FOR START OF DATA TRANSFER
0 - 632/1456 | 0 |
| - MAXIMUM TRACK FOR START OF DATA TRANSFER
0 - 2 | 2 |
| - MINIMUM TRACK FOR START OF DATA TRANSFER
0 - 2 | 0 |
| - MAXIMUM SECTOR FOR START OF DATA TRANSFER
0 - 25 | 25 |
| - MINIMUM SECTOR FOR START OF DATA TRANSFER
0 - 25 | 0 |
| - RATIO OF READ/WRITE
EVERY COMMAND RANDOMLY GENERATED BY THE
PROGRAM WILL BE EITHER A READ OR A WRITE.
FOR EACH DRIVE ONE CAN WEIGHT THE COMMANDS
TOWARDS READING OR WRITING. THE POSSIBLE
WEIGHTING RATIO'S ARE:
READ ONLY =0
7/1 =1
3/1 =2
5/3 =3
1/1 =4
3/5 =5
1/3 =6
1/7 =7 | 5/3 |
| - DO WRITE CHECK AFTER EVERY WRITE COMMAND
OTHERWISE, WRITE CHECK COMMANDS WILL BE
ISSUED RANDOMLY AFTER WRITE COMMANDS. | NO |
| - CORRECTABLE READ ERROR THRESHOLD | 10 |
| - UNCORRECTABLE READ ERROR THRESHOLD | 5 |
| - SEEK ERROR THRESHOLD | 5 |
| - INHIBIT ERROR CORRECTION | NO |
| - OPERATION COUNT THRESHOLD*65K | 4.290*10**9 |
| - DATA TRANSFER THRESHOLD*520K WORDS | 291*10**12 |
| - SAMPLED COMPARES | YES |
| - UP TO 5 EXCLUDED PACK AREAS
(NO DATA TRANSFER WILL BE DONE TO
THESE AREAS.) | NO EXCLUDED
PACK AREAS |

692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741

4.2.1 FORMATS FOR PACK AREA EXCLUSION

<u>FORMAT</u>	<u>ACTION</u>
CYL<CR>	ALL TRACK ON CYLINDER SPECIFIED WILL BE EXCLUDED.
CYL,TRK<CR>	THE ENTIRE TRACK WILL BE EXCLUDED.
CYL1,TRK1,CYL2,TRK2<CR>	ALL TRACKS FROM CYLINDER 1 TRACK 1 TO AND INCLUDING CYLINDER 2 TRACK 2 WILL BE EXCLUDED.

4.3 DIAGNOSTIC DUMP

DUE TO THE COMPLEXITY OF THIS PROGRAM A DUMP OF THE PROGRAM TABLES MUST ACCOMPANY EACH PROBLEM REPORT. THIS DUMP IS PROVIDED IN THE PROGRAM AND IS ACTIVATED BY STARTING THE PROGRAM AT LOCATION 220. THIS DUMP WILL PRINT THE PROCESSOR REGISTERS WHEN THE DUMP IS CALLED. IT WILL THEN PRINT ABOUT 12-15 TELETYPE PAGES OF SOFTWARE TABES NECESSARY TO DETERMINE WHAT THE PROGRAM WAS DOING AT A GIVEN TIME. APPROXIMATE TIME OF DUMP IS 10-12 MINUTES.

5.0 PROGRAM DESCRIPTION

THIS PROGRAM ASSUMES THAT ONLY THE PATTERNS WRITTEN BY THIS PROGRAM WILL BE ON THE PACK. TO PRECONDITION THE PACK A WN COMMAND MUST BE DONE. PACK SERIAL, DRIVE SERIAL NUMBER, THE TIME, AND DRIVE N UNDER TEST WILL BE PRINTED AT THE END OF PACK PRECONDITIONING. THE OPERATION OF THE PROGRAM IS DEPENDENT ON THE SWITCH REGISTER, SYSTEM PARAMETERS, AND DRIVE PARAMETERS. DRIVES MAY BE DROPPED FROM TEST BY USING THE DN COMMAND. IF A DRIVE IS DROPPED FOR ANY REASON, PERFORMANCE STATISTICS WILL AUTOMATICALLY BE PRINTED.

5.1 PROGRAM RESTRICTIONS

THIS PROGRAM WILL NOT RUN IN CHAIN MODE UNDER XXDP. NO END-OF-PASS INDICATOR IS PROVIDED FOR ACT. THE APT PROTOCOL HAS NOT BEEN DETERMINED YET.

742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791

6.0 PRINT OUTS

6.1 PERFORMANCE SUMMARY TYPEOUT

THE PERFORMANCE SUMMARY WILL CONTAIN THE FOLLOWING INFORMATION:

DRIVE NUMBER
DRIVE SERIAL NUMBER
TIME OF REPORT (IF REAL TIME CLOCK IS AVAILIABLE)
NUMBER OF ORDERS PERFORMED BY DRIVE
NUMBER OF SEEK OPERATIONS PERFORMED
TOTAL NUMBER OF WORDS WRITTEN BY DRIVE*65K
TOTAL NUMBER OF WORDS READ BY DRIVE*65K
NUMBER OF SOFT DATA ERRORS
(ECC CORRECTABLE)
(REREAD CORRECTABLE)
(OFFSET CORRECTABLE)
NUMBER OF HARD DATA ERRORS
NUMBER OF SEEK INCOMPLETES
NUMBER OF MISPOSITIONING ERRORS
TOTAL NUMBER OF ALL OTHER ERRORS.

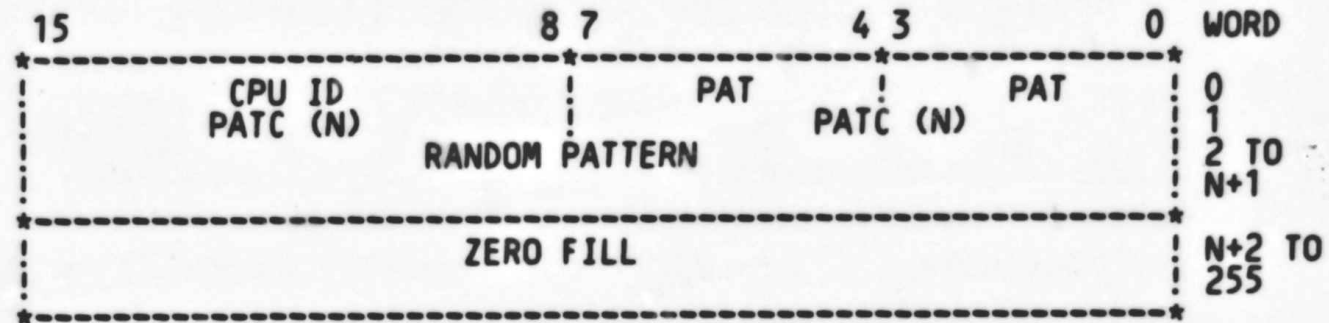
6.2 ERROR REPORTS

THE ERROR REPORTS WILL CONTAIN THE FOLLOWING INFORMATION:

DRIVE NUMBER
DRIVE SERIAL NUMBER
TIME ERROR OCCURRED (IF REAL TIME CLOCK IS AVAILIABLE)
DESCRIPTION OF ERROR
PRESENT COMMAND
PREVIOUS COMMAND
PRESENT POSITION
PREVIOUS POSITION
ALL DRIVE STATUS REGISTERS
ALL CONTROLLER REGISTERS
MEMORY ADDRESS, GOOD DATA AND BAD DATA (IF DATA ERROR)
OFFSET VALUE (IF APPLICABLE)
HEADER READ (IF MISPOSITIONING ERROR)
ECC PATTERN AND POSITION REGISTER IF DATA CHECKED

APPENDIX A

FIGURE A-1 SHOWS THE DATA EXPECTED BY THIS PROGRAM FOR EACH SECTOR OF THE RK06 PACK USED BY THIS PROGRAM.



CPU ID	CPU IDENTIFIER SUPPLIED BY THE OPERATOR
PAT	PATTERN USED
PATC	PATTERN COUNT (NUMBER OF WORDS OF PATTERN WRITTEN)

FIGURE A-1
 EXPECTED DATA FORMAT

792
 793
 794
 795
 796
 797
 798
 799
 800
 801
 802
 803
 804
 805
 806
 807
 808
 809
 810
 811
 812
 813
 814
 815
 816
 817
 818
 819
 820
 821
 822
 823
 824
 825
 826

827
828
829
830
831
832
833
834
835
836
837
838
839
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

DATA PATTERNS USED

PATTERN 0	PATTERN 1	PATTERN 2	PATTERN 3
165555	026455	007417	052525
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	026455	007417	052525
133333	151322	170360	125252
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	026455	007417	052525
133333	151322	170360	125252
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	151322	170360	125252

	PATTERN 4	PATTERN 5	PATTERN 6	PATTERN 7
867				
868				
869				
870				
871				
872	000000	000000	000001	000001
873	010421	177777	000003	000002
874	021042	000000	000007	000004
875	031463	000000	000017	000010
876	042104	177777	000037	000020
877	052525	177777	000077	000040
878	063146	000000	000177	000100
879	073567	000000	000377	000200
880	104210	000000	000777	000400
881	114631	177777	001777	001000
882	125252	177777	003777	002000
883	135673	177777	007777	004000
884	146314	000000	017777	010000
885	156735	000000	037777	020000
886	167356	000000	077777	040000
887	177777	000000	177777	100000
888	177777	177777	177776	100000
889	167356	177777	177774	040000
890	156735	177777	177770	020000
891	146314	177777	177760	010000
892	135673	000000	177740	004000
893	125252	000000	177700	002000
894	114631	000000	177600	001000
895	104210	177777	177400	000400
896	073567	177777	177000	000200
897	063146	177777	176000	001000
898	052525	000000	174000	000040
899	042104	000000	170000	000020
900	031463	177777	160000	000010
901	021042	177777	140000	000004
902	010421	000000	100000	000002
903	000000	177777	000000	000001

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
935
936
937
938
939
940

PATTERN 10	PATTERN 11	PATTERN 12	PATTERN 13
177776	172666	153333	000000
177775	155555	066667	177777
177773	172666	153333	177777
177767	155555	066667	177777
177757	172666	153333	177777
177737	155555	066667	177777
177677	172666	153333	177777
177577	155555	066667	177777
177377	172666	153333	177777
176777	155555	066667	177777
175777	172666	153333	177777
173777	155555	066667	177777
167777	172666	153333	177777
157777	155555	066667	177777
137777	172666	153333	177777
077777	155555	066667	177777
077777	172666	153333	177777
137777	155555	066667	177777
157777	172666	153333	177777
167777	155555	066667	177777
173777	172666	153333	177777
175777	155555	066667	177777
176777	172666	153333	177777
177377	155555	066667	177777
177577	172666	153333	177777
177677	155555	066667	177777
177737	172666	153333	177777
177757	155555	066667	177777
177767	172666	153333	177777
177773	155555	066667	177777
177775	172666	153333	177777
177776	155555	066667	177777

	PATTERN 14	PATTERN 15	PATTERN 16	PATTERN 17
941	177777	172304	070627	133467
942	000000	172304	113431	133467
943	000000	172304	014561	133467
944	000000	172304	070627	133467
945	000000	172304	113431	133467
946	000000	172304	014561	133467
947	000000	172304	070627	133467
948	000000	172304	113431	133467
949	000000	172304	014561	133467
950	000000	172304	070627	133467
951	000000	172304	113431	133467
952	000000	172304	014561	133467
953	000000	172304	070627	133467
954	000000	172304	113431	133467
955	000000	172304	014561	133467
956	000000	172304	070627	133467
957	000000	172304	113431	133467
958	000000	172304	014561	133467
959	000000	172304	070627	133467
960	000000	172304	113431	133467
961	000000	172304	014561	133467
962	000000	172304	070627	133467
963	000000	172304	113431	133467
964	000000	172304	014561	133467
965	000000	172304	070627	133467
966	000000	172304	113431	133467
967	000000	172304	014561	133467
968	000000	172304	070627	133467
969	000000	172304	113431	133467
970	000000	172304	014561	133467
971	000000	172304	070627	133467
972	000000	172304	113431	133467
973	000000	172304	014561	133467
974	000000	172304	070627	133467
975	000000	172304	113431	133467
976	000000	172304	014561	133467
977	000000	172304	070627	133467
978	a	172304	113431	133467

979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022

: *** REV 005 ***

.TITLE CZR6P00 RK611/06 PERF EXEC
:*COPYRIGHT (C) 1976,1982
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY ROY SPITZER
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C) FEB 1982.
:*

:*****

:* THE RK06-RK07 PERFORMANCE EXERCISER WILL EXERCISE 1 TO 8
:* RK06-RK07 DISK DRIVES ATTACHED TO THE SAME RK06
:* UNIBUS CONTROLLER IN A DEDICATED STAND ALONE MODE.

:* UNDER NORMAL OPERATION SITUATIONS, THE DRIVE IS
:* CHOSEN RANDOMLY FROM ANY ONE OF THE RK06 DRIVES UNDER TEST
:* WHICH IS NOT CURRENTLY UNDERGOING ERROR RECOVERY.
:* TO OPTIMIZE THE THROUGHOUT ON THE RK06 CONTROLLER EACH
:* DATA TRANSFER COMMAND IS TRANSLATED INTO A SEEK FOLLOWED
:* BY THE DATA TRANSFER COMMAND.

:*****

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
10	BELL ON ERROR
6	INHIBIT AUTOMATIC DRIVE DEASSIGNMENT IF UNSAFE, AC LOW, OR SPEED LOSS OCCURS
5	INHIBIT AUTOMATIC DRIVE DEASSIGNMENT WHEN MAXIMUM COMMAND COUNT FOR DRIVE IS EXCEEDED
4	INHIBIT AUTOMATIC DRIVE DEASSIGNMENT WHEN ERROR THRESHOLD EXCEEDED
3	DUMP ENTIRE SECTOR BEFORE BEGINNING RETRY SEQUENCE
1	INHIBIT SOFTWARE DATA COMPARISONS

```
1023      .SBTTL BASIC DEFINITIONS
1024
1025      :*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1026      001100      STACK= 1100
1027      .EQUIV EMT,ERROR      ::BASIC DEFINITION OF ERROR CALL
1028      .EQUIV IOT,SCOPE      ::BASIC DEFINITION OF SCOPE CALL
1029
1030      :*MISCELLANEOUS DEFINITIONS
1031      000011      HT= 11      ::CODE FOR HORIZONTAL TAB
1032      000012      LF= 12      ::CODE FOR LINE FEED
1033      000015      CR= 15      ::CODE FOR CARRIAGE RETURN
1034      000200      CRLF= 200      ::CODE FOR CARRIAGE RETURN-LINE FEED
1035      177776      PS= 177776      ::PROCESSOR STATUS WORD
1036      .EQUIV PS,PSW
1037      177774      STKLMT= 177774      ::STACK LIMIT REGISTER
1038      177772      PIRQ= 177772      ::PROGRAM INTERRUPT REQUEST REGISTER
1039      177570      DSWR= 177570      ::HARDWARE SWITCH REGISTER
1040      177570      DDISP= 177570      ::HARDWARE DISPLAY REGISTER
1041
1042      :*GENERAL PURPOSE REGISTER DEFINITIONS
1043      000000      R0= %0      ::GENERAL REGISTER
1044      000001      R1= %1      ::GENERAL REGISTER
1045      000002      R2= %2      ::GENERAL REGISTER
1046      000003      R3= %3      ::GENERAL REGISTER
1047      000004      R4= %4      ::GENERAL REGISTER
1048      000005      R5= %5      ::GENERAL REGISTER
1049      000006      R6= %6      ::GENERAL REGISTER
1050      000007      R7= %7      ::GENERAL REGISTER
1051      000006      SP= %6      ::STACK POINTER
1052      000007      PC= %7      ::PROGRAM COUNTER
1053
1054      :*PRIORITY LEVEL DEFINITIONS
1055      000000      PR0= 0      ::PRIORITY LEVEL 0
1056      000040      PR1= 40      ::PRIORITY LEVEL 1
1057      000100      PR2= 100      ::PRIORITY LEVEL 2
1058      000140      PR3= 140      ::PRIORITY LEVEL 3
1059      000200      PR4= 200      ::PRIORITY LEVEL 4
1060      000240      PR5= 240      ::PRIORITY LEVEL 5
1061      000300      PR6= 300      ::PRIORITY LEVEL 6
1062      000340      PR7= 340      ::PRIORITY LEVEL 7
1063
1064      :*"SWITCH REGISTER" SWITCH DEFINITIONS
1065      100000      SW15= 100000
1066      040000      SW14= 40000
1067      020000      SW13= 20000
1068      010000      SW12= 10000
1069      004000      SW11= 4000
1070      002000      SW10= 2000
1071      001000      SW09= 1000
1072      000400      SW08= 400
1073      000200      SW07= 200
1074      000100      SW06= 100
1075      000040      SW05= 40
1076      000020      SW04= 20
1077      000010      SW03= 10
1078      000004      SW02= 4
```

1079 000002
1080 000001
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090

SW01= 2
SW00= 1
.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

1092
1093 100000
1094 040000
1095 020000
1096 010000
1097 004000
1098 002000
1099 001000
1100 000400
1101 000200
1102 000100
1103 000040
1104 000020
1105 000010
1106 000004
1107 000002
1108 000001
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

1120
1121 000004
1122 000010
1123 000014
1124 000014
1125 000014
1126 000020
1127 000024
1128 000030
1129 000034
1130 000060
1131 000064
1132 000240
1133 000240
1134 120210

;*BASIC "CPU" TRAP VECTOR ADDRESSES

ERRVEC= 4
RESVEC= 10
TBITVEC= 14
TRTVEC= 14
BPTVEC= 14
IOTVEC= 20
PWRVEC= 24
EMTVEC= 30
TRAPVEC= 34
TKVEC= 60
TPVEC= 64
PIRQVEC= 240
APRIOR= PR5
AVECT1= 120210
::TIME OUT AND OTHER ERRORS
::RESERVED AND ILLEGAL INSTRUCTIONS
::"T" BIT
::TRACE TRAP
::BREAKPOINT TRAP (BPT)
::INPUT/OUTPUT TRAP (IOT) **SCOPE**
::POWER FAIL
::EMULATOR TRAP (EMT) **ERROR**
::"TRAP" TRAP
::TTY KEYBOARD VECTOR
::TTY PRINTER VECTOR
::PROGRAM INTERRUPT REQUEST VECTOR
:RK611 PRIORITY
:RK611 VECTOR

```
1135      177440      ABASE= 177440      :RK611 BASE
1136      000114      MEMVEC= 114        :VECTOR FOR MEMORY CHECK ENABLE
1137      172100      MEMBAS= 172100     :BUS ADDRESS FOR MEMORY CHECK ENABLE
1138      000001      PAR.EN= 1         :MEMORY ENABLE MEMORY CHECKING
1139
1140      .SBTTL  RK06-RK07 CONTROLLER REGISTER DEFINITION
1141
1142      000000      RKCS1= 0         :CONTROL AND STATUS REGISTER 1
1143      000002      RKWC= 2         :WORD COUNT REGISTER
1144      000004      RKBA= 4         :BUS ADDRESS REGISTER
1145      000006      RKDA= 6         :DESIRED TRACK SECTOR REGISTER
1146      000010      RKCS2= 10      :CONTROL AND STATUS REGISTER 2
1147      000012      RKDS= 12      :DRIVE STATUS REGISTER
1148      000014      RKER= 14      :ERROR REGISTER
1149      000016      RKASOF= 16     :ATTENTION SUMMARY AND OFFSET REGISTER
1150      000020      RKDC= 20      :DESIRED CYLINDER REGISTER
1151      000020      RKDCYL= 20     :DESIRED CYLINDER REGISTER
1152      000024      RKDB= 24      :DATA BUFFER
1153      000026      RKMR1= 26     :MAINTENANCE REGISTER 1
1154      000034      RKMR2= 34     :MAINTENANCE REGISTER 2
1155      000036      RKMR3= 36     :MAINTENANCE REGISTER 3
1156      000030      RKPOS= 30     :ECC POSITION INFORMATION
1157      000030      RKECPS= 30    :ECC POSITION INFORMATION
1158      000032      RKPAT= 32     :ECC PATTERN INFORMATION
1159      000032      RKECPT= 32    :ECC PATTERN INFORMATION
1160
1161      .SBTTL  DRIVE COMMANDS
1162
1163      000101      SELDRV= 101     :SELECT DRIVE
1164      000103      PACK= 103      :PACK ACKNOWLEDGE
1165      000105      CLEAR= 105     :DRIVE CLEAR
1166      000107      UNLOAD= 107    :UNLOAD
1167      000111      SRTSPL= 111    :START SPINDLE
1168      000113      RECAL= 113     :RECALIBRATE
1169      000115      OFFSET= 115    :OFFSET
1170      000117      SEEK= 117     :SEEK
1171      000121      RDDATA= 121    :READ DATA
1172      000123      WRDATA= 123    :WRITE DATA
1173      000125      RDHEAD= 125    :READ HEADER
1174      000127      WRHEAD= 127    :WRITE HEADER AND DATA
1175      000131      WRTCHK= 131    :WRITE CHECK
1176
1177      :          THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
1178      :          TO SIMULATE A SPECIFIC DESIRED OPERATION
1179
1180      000140      RELEAS= 140     :RELEASE DRIVE
1181      000141      RDSTAT= 141    :GET ALL STATUS FROM DRIVE
1182      000164      RDALHD= 164    :READ ALL HEADERS
1183      000176      CONCLR= 176    :CONTROLLER CLEAR (BIT 15 OF CS1)
1184      000177      SUBCLR= 177    :SUBSYSTEM CLEAR (BIT 5 OF CS2)
1185      000300      INTR= 300      :GENERATE INTERRUPT TO CPU
1186
1187      :          DRIVER ISSUED SERVICE COMMANDS
1188
1189      000001      DR.SEL= 001     :DRIVE SELECT
1190      000005      DR.CLR= 005     :DRIVE CLEAR
```

```

1191
1192
1193
1194      000001
1195      000100
1196      000200
1197      000400
1198      001000
1199      002000
1200      004000
1201
1202      010000
1203      020000
1204      040000
1205      100000
1206      100000
1207
1208
1209
1210
1211      000001
1212      000002
1213      000004
1214      000020
1215
1216
1217
1218      000007
1219      000010
1220      000010
1221      000020
1222      000040
1223      000040
1224      000100
1225      000200
1226      000400
1227      001000
1228      002000
1229      004000
1230      010000
1231      020000
1232      040000
1233      100000
1234
1235
1236
1237      000001
1238
1239      000002
1240      000004
1241      000004
1242      000010
1243      000020
1244      000040
1245      000100
1246      000200

```

.SBTTL CONTROL AND ATUS REGISTER 1 BITS

```

GO=      BIT0      :GO BIT
IE=      BIT6      :INTERRUPT ENABLE
RDY=     BIT7      :CONTROLLER READY
BA16=    BIT8      :BUS ADDRESS BIT 16
BA17=    BIT9      :BUS ADDRESS BIT 17
CDT=     BIT10     :CONTROLLER DRIVE TYPE (0=RK06, 1=RK07)
CTO=     BIT11     :CONTROLLER TIMED OUT WAITING FOR
           :DRIVE RESPONSE
CFMT=    BIT12     :CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
SPAR=    BIT13     :DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
DI=      BIT14     :DRIVE INTERRUPT
CERR=    BIT15     :CONTROLLER ERROR
CCLR=    BIT15     :CONTROLLER CLEAR

:      THESE BIT DEFINITIONS ARE USED FOR ADDRESS
:      THE HIGH BYTE OF RKCS1

B.BA16=  BIT0      :BUS ADDRESS BIT 16
B.BA17=  BIT1      :BUS ADDRESS BIT 17
B.CDT=   BIT2      :CONTROLLER DRIVE TYPE (0=RK06, 1=RK07)
B.CFMT=  BIT4      :CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

DRVMSK=  7         :MASK FOR DRIVE SELECTION CODE
DESL=    BIT3      :DESELECT OR RELEASE DRIVE IN BITS 0-2
RLS=     BIT3      :DESELECT OR RELEASE DRIVE IN BITS 0-2
BAI=     BIT4      :BUS ADDRESS INCREMENT INHIBIT
CLR=     BIT5      :CLEAR CONTROLLER AND ALL DRIVES
SCLR=    BIT5      :CLEAR CONTROLLER AND ALL DRIVES
IR=      BIT6      :INPUT READY
OR=      BIT7      :OUTPUT READY
UFE=     BIT8      :UNIT FIELD ERROR
MDS=     BIT9      :MULTIPLE DRIVE SELECT
PGE=     BIT10     :PROGRAMMING ERROR
NEM=     BIT11     :NON-EXISTENT MEMORY
NED=     BIT12     :NON-EXISTENT DRIVE
UPE=     BIT13     :UNIBUS PARITY ERROR
WCE=     BIT14     :WRITE CHECK ERROR
DLT=     BIT15     :DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

ILC=     BIT0      :ILLEGAL FUNCTION CODE
:*ILF=   BIT0      :ILLEGAL FUNCTION CODE
SKI=     BIT1      :SEEK INCOMPLETE
ILF=     BIT2      :ILLEGAL DRIVE FUNCTION
NXF=     BIT2      :ILLEGAL DRIVE FUNCTION
DRPAR=   BIT3      :DRIVE DETECTED DRIVE BUS PARITY ERROR
FMTE=    BIT4      :FORMAT ERROR
DTYE=    BIT5      :DRIVE TYPE ERROR
ECH=     BIT6      :ECC HARD
BSE=     BIT7      :BAD SECTOR ERROR

```

1247	000400	HCRC= BIT8	:HEADER CRC ERROR
1248	000400	HVRC= BIT8	:HEADER VRC ERROR
1249	001000	COE= BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1250	002000	IDAE= BIT10	:INVALID DISK ADDRESS ERROR
1251	004000	WLE= BIT11	:WRITE LOCK ERROR
1252	010000	DTF= BIT12	:DRIVE TIMING ERROR
1253	020000	OP1= BIT13	:OPERATION (SEARCH) INCOMPLETE
1254	040000	UNS= BIT14	:DRIVE UNSAFE
1255	100000	DCK= BIT15	:DATA CHECK
1256			
1257		.SBTTL STATUS REGISTER BIT DEFINITION	
1258			
1259	000001	DRA= BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF : THIS BIT IS RESET)
1260			
1261	000004	OFST= BIT2	:DRIVE OFFSET
1262	000010	ACLO= BIT3	:AC LOW
1263	000020	SPDLSS= BIT4	:SPEED LOSS
1264	000020	DCLO= BIT4	:DC LOW
1265	000040	DROT= BIT5	:DRIVE OFF TRACK
1266	000100	VV= BIT6	:VOLUME VALID
1267	000200	DRY= BIT7	:DRIVE READY
1268	000200	DRDY= BIT7	:DRIVE READY
1269	000400	DDT= BIT8	:DRIVE TYPE (0=RK06, 1=RK07)
1270	004000	WRL= BIT11	:WRITE LOCK
1271	020000	PIP= BIT13	:POSITIONING IN PROGRESS
1272	040000	DSC= BIT14	:DRIVE STATUS CHANGE
1273	100000	SVAL= BIT15	:STATUS VALID
1274			
1275		.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION	
1276			
1277	000017	MESMSK= 17	:MESSAGE MASK
1278			
1279	000020	PAT= BIT4	:FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
1280	000040	DMD= BIT5	:DIAGNOSTIC MODE
1281	000100	MSP= BIT6	:MAINTENANCE SECTOR PULSE
1282	000200	MIND= BIT7	:MAINTENANCE INDEX
1283	000400	MCLK= BIT8	:MAINTENANCE CLOCK
1284	001000	MERD= BIT9	:MAINTENANCE ENCODED READ DATA
1285	002000	MEWD= BIT10	:MAINTENANCE ENCODED WRITE DATA
1286	004000	PCA= BIT11	:PRECOMPENSATION ADVANCE
1287	010000	PCD= BIT12	:PRECOMPENSATION DELAY
1288	020000	ECCW= BIT13	:ECC WORD IS BEING READ OR WRITTEN
1289	040000	WRTGAT= BIT14	:WRITE GATE
1290	100000	RDGATE= BIT15	:READ GATE
1291			
1292		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A	
1293			
1294	000040	S.DRA= BIT5	:DRIVE AVAILIABLE
1295	000100	S.VV= BIT6	:VOLUME VALID
1296	000200	S.DRY= BIT7	:DRIVE READY
1297	000400	S.TYPE= BIT8	:DRIVE TYPE
1298	001000	S.FORM= BIT9	:DRIVE FORMAT
1299	002000	S.OFF= BIT10	:OFFSET
1300	004000	S.WRL= BIT11	:WRITE LOCK
1301	010000	S.SPIN= BIT12	:SPINDLE ON
1302	020000	S.PIP= BIT13	:POSITIONING IN PROGRESS

```
1303      040000      S.DSC= BIT14          :DRIVE STATUS CHANGE
1304
1305      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
1306
1307      000040      S.ICYL= BIT5          :ILLEGAL CYLINDER ADDRESS
1308      000100      S.ACLN= BIT6          :AC LOW
1309      000200      S.FLT= BIT7          :DRIVE FAULT
1310      000400      S.ILF= BIT8          :ILLEGAL FUNCTION
1311      001000      S.PAR= BIT9          :DRIVE DETECTED DRIVE BUS PARITY ERROR
1312      002000      S.SKI= BIT10         :SEEK INCOMPLETE
1313      004000      S.WLE= BIT11         :WRITE LOCK ERROR
1314      010000      S.SPLS= BIT12        :SPEED LOSS
1315      010000      S.DCLO= BIT12        :DC LOW
1316      020000      S.DROT= BIT13        :DRIVE OFF TRACK
1317      040000      S.UNS= BIT14         :DRIVE UNSAFE
1318
1319      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
1320
1321      000020      S.XDOK= BIT4          :TRANSDUCER OK
1322      000040      S.HDHM= BIT5          :HEADS HOME
1323      000100      S.BRHM= BIT6          :BRUSHES HOME
1324      000200      S.DOOR= BIT7          :DOOR INTERLOCKED
1325      000400      S.CART= BIT8          :CARTRAGE INTERLOCK
1326      001000      S.SPOK= BIT9          :SPEED OK
1327      002000      S.FWD= BIT10         :FORWARD
1328      004000      S.REV= BIT11         :REVERSE
1329      010000      S.LOAD= BIT12        :HEADS LOADING
1330      020000      S.RTZ= BIT13        :RETURN TO ZERO
1331      040000      S.UNLD= BIT14        :HEADS UNLOADING
1332
1333      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
1334
1335      000020      S.SECT= BIT4          :SECTOR ERROR
1336      000040      S.WCLK= BIT5          :WRITE CLOCK AND NO WRITE GATE
1337      000100      S.WGAT= BIT6          :WRITE GATE AND NO TRANSISTIONS
1338      000200      S.HDFL= BIT7          :HEAD FAULT
1339      000400      S.MHD= BIT8          :MULTIPLE HEAD SELECT
1340      001000      S.XERR= BIT9          :INDEX ERROR
1341      002000      S.DIB= BIT10         :DIBIT ERROR
1342      004000      S.PLO= BIT11         :PLO ERROR
1343      010000      S.NMOV= BIT12        :SEEK AND NO MOTION
1344      020000      S.LIMD= BIT13        :LIMIT DETECT ON SEEK
1345      040000      S.BRKE= BIT14        :SERVO-BRAKE
1346
1347      .SBTTL  COMMON MASKS
1348
1349      000007      M.DRV= 7          :DRIVE CODE
1350      100000      M.PAR= BIT15         :PARITY
1351      000003      M.ID= 3          :BYTE ID
1352      017760      M.CDIF= 17760       :CYLINDER DIFFERENCE/OFFSET
1353      017760      M.CADD= 17760       :CYLINDER ADDRESS
1354      077770      M.SER= 77770        :DRIVE SERIAL NUMBER
1355      000760      M.SECT= 760         :SECTOR COUNT
1356      007000      M.HEAD= 7000        :HEAD DECODE
1357
1358      .SBTTL  DEFAULT DRIVE VALUES
```


1359			
1360	000000	D.MNCL= 0	:MINIMUM CYLINDER
1361	000000	D.MNTR= 0	:MINIMUM TRACK
1362	000002	D.MXTR= 2	:MAXIMUM TRACK
1363	000000	D.MNSC= 0	:MINIMUM SECTOR
1364	000025	D.MXSC= 21.	:MAXIMUM SECTOR
1365	000003	D.RATE= 3	:READ/WRITE RATIO (5/3)
1366	000000	D.AWCK= 0	:NO AUTOMATIC WRITE-CHECK AFTER EVERY WRITE
1367	000012	D.CERT= 10.	:CORRECTABLE READ ERROR THRESHOLD
1368	000005	D.UERT= 5	:UNCORRECTABLE READ ERROR THRESHOLD
1369	000005	D.SKET= 5	:SEEK ERROR THRESHOLD
1370	077777	D.CTRH= 77777	:COMMAND COUNT THRESHOLD
1371	177770	D.CTRL= 177770	: (MAXIMUM OF COMMANDS ISSUED TO DRIVE)
1372	077777	D.WTHI= 77777	:WORDS TRANSFERRED THRESHOLD
1373	177770	D.WTLO= 177770	: (MAXIMUM NUMBER OF WORDS READ OR WRITTEN
1374			: ON PACK)
1375	000020	T.NER= 20	:THRESHOLD OF OTHER DRIVE ERRORS
1376			
1377		.SBTTL OFFSET VALUES	
1378			
1379	000060	OFFP12= 060	:OFFSET +1200 MICRO-INCHES
1380	000040	OFFP8= 040	:OFFSET +800 MICRO-INCHES
1381	000020	OFFP4= 020	:OFFSET +400 MICRO-INCHES
1382	000220	OFFM4= 220	:OFFSET -400 MICRO-INCHES
1383	000240	OFFM8= 240	:OFFSET -800 MICRO-INCHES
1384	000260	OFFM12= 260	:OFFSET -1200 MICRO-INCHES
1385			
1386		.SBTTL DEFAULT CONTROLLER THRESHOLD	
1387			
1388	001000	T.DLT= 1000	:THRESHOLD FOR DATA LATE
1389	000012	T.CONT= 10.	:THRESHOLD FOR OTHER CONTROLLER ERRORS

1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445

.SBTTL PARAMETER BLOCK ALLOCATION

```

*****
: * 1  : COMMAND          ! DRIVE NO.          : 0
: * 3  : CYLINDER ADDRESS      : 2
: * 5  : TRACK              ! SECTOR          : 4
: * 7  : BA16-17,FORMAT,DRV TYPE! OFFSET          : 6
: * 11 : BUS ADDRESS (LOW 16 BITS) : 10
: * 13 : WORD COUNT (2'S COMPLEMENT) : 12
: * 15 : PROGRAM DRIVE STATUS INFORMATION : 14
: * 17 : COMMAND AND STATUS REGISTER 1 : 16
: * 21 : COMMAND AND STATUS REGISTER 2 : 20
: * 23 : WORD COUNT REGISTER          : 22
: * 25 : BUS ADDRESS REGISTER         : 24
: * 27 : DESIRED TRACK AND SECTOR     : 26
: * 31 : DESIRED CYLINDER             : 30
: * 33 : ATTENTION SUMMARY AND DRIVE OFFSET : 32
: * 35 : ERROR REGISTER              : 34
: * 37 : STATUS REGISTER             : 36
: * 41 : MESSAGE LINE A STATUS BYTE 00 : 40
: * 43 : MESSAGE LINE B STATUS BYTE 00 : 42
: * 45 : MESSAGE LINE A STATUS BYTE 01 : 44
: * 47 : MESSAGE LINE B STATUS BYTE 01 : 46
: * 51 : MESSAGE LINE A STATUS BYTE 10 : 50
: * 53 : MESSAGE LINE B STATUS BYTE 10 : 52
: * 55 : MESSAGE LINE A STATUS BYTE 11 : 54
: * 57 : MESSAGE LINE B STATUS BYTE 11 : 56
: * 61 : ECC POSITION INFORMATION      : 60
: * 63 : ECC PATTERN INFORMATION     : 62
*****

```

.SBTTL PARAMETERS PASSED TO THE DRIVER

: THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
: TO THE RK06/07 DRIVER

000000	P.DRVN= 0	:DRIVE NUMBER
000001	P.CMND= 1	:COMMAND
000002	P.CYLN= 2	:CYLINDER ADDRESS
000004	P.SECT= 4	:SECTOR
000005	P.TRCK= 5	:TRACK
000006	P.OFST= 6	:OFFSET
000007	P.CS1H= 7	:RKCS1 BITS 8-15
000007	P.BAHI= 7	:BUS ADDRESS (BITS 16 AND 17)
000010	P.BALO= 10	:BUS ADDRESS (BITS 0-15)
000012	P.WC= 12	:WORD COUNT (2'S COMPLEMENT)
000014	P.PRST= 14	:PROGRAM DRIVE STATUS INFORMATION

.SBTTL PROGRAM DEVICE STATUS REGISTER DEFINITION

000001	DRVUSE= BIT0	:DRIVE IN USE
000002	DRVPOS= BIT1	:DRIVE POSITIONING
000004	DRVPDT= BIT2	:DRIVE POSITIONED FOR DATA TRANSFER
000010	UEXATT= BIT3	:UNEXPECTED ATTENTION
000020	DRVHRD= BIT4	:DRIVE HAS HARD ERROR
000040	DRVDSC= BIT5	:DRIVE STATUS CHANGE DID NOT CLEAR

```
1446      000100      CMDTO= BIT6      ;NO TERMINATION TO COMMAND FOR AT
1447      ;          ;      LEAST 1 SECOND
1448      000200      W.WCK= BIT7      ;WRITE FOR WRITE WRITE CHECK
1449      000400      NOCHK= BIT8      ;NO CHECK, DO NOT SET INTERRUPT ENABLE
1450      001000      PBSVAL= BIT9      ;PARAMETER STATUS WORDS VALID
1451      ;          ;      (SET WHEN ERROR TERMINATION OR
1452      ;          ;      READ STATUS COMMAND)
1453      002000      DRPDRV= BIT10      ;DROP DRIVE FROM TEST SEQUENCE
1454      004000      NODSC= BIT11      ;ATTENTION SET BUT DCS AND FAULT RESET
1455      010000      DRVSZD= BIT12      ;DRIVE SEIZED BY OTHER PORT
1456      020000      E.UNLD= BIT13      ;DRIVE UNLOADED DUE TO ERROR
1457      040000      Q.INIT= BIT14      ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
1458      100000      DTBAII= BIT15      ;INHIBIT BUS ADDRESS INCREMENT
1459
1460      .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
1461
1462      ;          ;      THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
1463      ;          ;      FROM THE DRIVER TO THE CALLING PROGRAM
1464
1465      000016      P.CS1= 16      ;COMMAND AND STATUS REGISTER 1
1466      000020      P.CS2= 20      ;COMMAND AND STATUS REGISTER 2
1467      000022      P.WCR= 22      ;WORD COUNT REGISTER
1468      000024      P.BAR= 24      ;BUS ADDRESS REGISTER
1469      000026      P.DTS= 26      ;DESIRED TRACK SECTOR REGISTER
1470      000030      P.DCYL= 30      ;DESIRED CYLINDER REGISTER
1471      000032      P.ASOF= 32      ;ATTENTION SUMMARY/OFFSET REGISTER
1472      000034      P.ER= 34      ;ERROR REGISTER
1473      000036      P.DS= 36      ;STATUS REGISTER
1474      000040      P.A00= 40      ;MESSAGE A STATUS BYTE 00
1475      000042      P.B00= 42      ;MESSAGE B STATUS BYTE 00
1476      000044      P.A01= 44      ;MESSAGE A STATUS BYTE 01
1477      000046      P.B01= 46      ;MESSAGE B STATUS BYTE 01
1478      000050      P.A10= 50      ;MESSGGE A STATUS BYTE 10
1479      000052      P.B10= 52      ;MESSAGE B STATUS BYTE 10
1480      000054      P.A11= 54      ;MESSAGE A STATUS BYTE 11
1481      000056      P.B11= 56      ;MESSAGE B STATUS BYTE 11
1482      000060      P.EPOS= 60      ;ECC POSITION INFORMATION
1483      000062      P.EPAT= 62      ;ECC PATTERN INFORMATION
1484
1485      .SBTTL  DRIVE SPECIFIC PARAMETERS
1486
1487      000064      P.QLNK= 64      ;QUEUE LINK
```

1488 ; THE FOLLOWING DEFINITIONS ARE USED FOR DRIVE PARAMETER ACCESS

1489			
1490	000066	P.MNCL= 66	:MINIMUM CYLINDER
1491	000070	P.MXCL= 70	:MAXIMUM CYLINDER
1492	000072	P.MNTR= 72	:MINIMUM TRACK
1493	000073	P.MXTR= 73	:MAXIMUM TRACK
1494	000074	P.MNSC= 74	:MINIMUM SECTOR
1495	000075	P.MXSC= 75	:MAXIMUM SECTOR
1496	000076	P.RATE= 76	:READ/WRITE RATIO
1497	000077	P.AWCK= 77	:AUTOMATIC WRITE CHECK
1498	000100	P.CERT= 100	:CORRECTABLE READ ERROR THRESHOLD
1499	000102	P.UERT= 102	:UNCORRECTABLE READ ERROR THRESHOLD
1500	000104	P.SKET= 104	:SEEK ERROR THRESHOLD
1501	000106	P.MXCD= 106	:MAXIMUM COMMANDS ISSUED TO DRIVE
1502	000112	P.MXWT= 112	:MAXIMUM NUMBER WORDS TRANSFERRED
1503	000116	P.SMPL= 116	:SAMPLE COMPARE FLAG
1504	000117	P.ECMP= 117	:ECC COMPARE FLAG
1505	000120	P.IERC= 120	:INHIBIT ERROR CORRECTION
1506	000121	P.DPAT= 121	:DATA PATTERN USED

1507 ;
1508 ; THE FOLLOWING DEFINITIONS ARE USED IN STORING
1509 ; CURRENT RANDOMLY GENERATED COMMAND

1510			
1511	000122	P.RDPT= 122	:CURRENT RANDOMLY GENERATED DATA PATTERN
1512	000123	P.RCMD= 123	:CURRENT RANDOMLY GENERATED COMMAND
1513	000124	P.RCYL= 124	:CURRENT RANDOMLY GENERATED CYLINDER
1514	000126	P.RSEC= 126	:CURRENT RANDOMLY GENERATED SECTOR
1515	000127	P.RTRK= 127	:CURRENT RANDOMLY GENERATED TRACK
1516	000130	P.RWC= 130	:CURRENT RANDOMLY GENERATED WORD COUNT
1517	000132	P.RBAL= 132	:CURRENT RANDOMLY GENERATER BUS ADDRESS
1518	000134	P.RBAH= 134	

1519 ;
1520 ; THE FOLLOWING DEFINITIONS ARE USED IN STORING
1521 ; LAST ISSUED

1522			
1523	000135	P.LCMD= 135	:LAST COMMAND
1524	000136	P.LCYL= 136	:LAST CYLINDER
1525	000140	P.LSEC= 140	:LAST SECTOR
1526	000141	P.LTRK= 141	:LAST TRACK
1527	000142	P.LWC= 142	:LAST WORD COUNT
1528	000144	P.LDPT= 144	:LAST DATA PATTERN
1529	000145	P.BUFF= 145	:BUFFER ALLOCATED

: THE FOLLOWING IS USED FOR THE RETRY SEQUENCE

1530			
1531			
1532	000146	P.RECT= 146	:RETRY COUNT
1533	000147	P.RERD= 147	:RETRY INDICATION FOR DATA ERRORS
1534			:0 NOT IN RETRY SEQUENCE
1535			:1-20 REREAD AT CENTER LINE
1536			:21 OFFSET +400 MICRO-INCHES
1537			:22-23 REREAD OFFSET +400 MICRO-INCHES
1538			:24 OFFSET -400 MICRO-INCHES
1539			:25-26 REREAD OFFSET -400 MICRO-INCHES
1540			:27 OFFSET +800 MICRO-INCHES
1541			:30-31 REREAD OFFSET +800 MICRO-INCHES
1542			:32 OFFSET -800 MICRO-INCHES
1543			:33-34 REREAD OFFSET -800 MICRO-INCHES
1544			:35 OFFSET +1200 MICRO-INCHES
1545			:36-37 REREAD OFFSET +1200 MICRO-INCHES
1546			:40 OFFSET -1200 MICRO-INCHES
1547			:41-42 REREAD OFFSET -1200 MICRO-INCHES
1548			
1549	000150	P.DSTT= 150	:DRIVE STATUS INFORMATION FOR ERROR PROCESSING
1550			:BIT 0 INITIAL DATA ERROR
1551			:BIT 1 DATA ERROR BEING PROCESSED
1552			:BIT 2 NON-DATA ERROR BEING PROCESSED
1553			:BIT 3 SECTOR IN ERROR PRINTED
1554			:BIT 4 READ HEADER ISSUED
1555			:BIT 6 WAITING FOR READ STATUS FOR
1556			: POSITIONING TIME OUT
1557			:BIT 7 FIRST ERROR
1558			:BIT 11 RECALIBRATE HAS BEEN ISSUED
1559			:BIT 12 SEEK HAS BEEN ISSUED
1560			:BIT 13 OFFSET HAS BEEN ISSUED
1561			:BIT 14 BAD SECTOR PROCESSING FOR WRITE CHECK
1562			:BIT 15 ERROR ENQUEUED
1563			
1564			
1565			
1566	000152	P.NODR= 152	:NUMBER OF ORDERS (2 WORDS)
1567	000156	P.NWRT= 156	:NUMBER OR WORDS WRITTEN (3 WORDS)
1568	000164	P.NRD= 164	:NUMBER OF WORDS READ (3 WORDS)
1569	000172	P.SRRD= 172	:NUMBER SOFT ERRORS (REREAD CORRECTABLE)
1570	000174	P.SOFF= 174	:NUMBER SOFT ERRORS (OFFSET CORRECTABLE)
1571	000176	P.SECC= 176	:NUMBER SOFT ERRORS (ECC CORRECTABLE)
1572	000200	P.HARD= 200	:NUMBER OF HARD DATA ERRORS
1573	000202	P.NSKI= 202	:NUMBER OF SEEK INCOMPLETE
1574	000204	P.NOPI= 204	:NUMBER OF OPERATION INCOMPLETES
1575	000206	P.NER= 206	:NUMBER OF ALL OTHER ERRORS

: THE FOLLOWING DEFINITIONS ARE USED IN STATISTICAL GATHERING

```
1576 ; THE FOLLOWING IS USED FOR THE DRIVE ASSIGNMENT SEQUENCE
1577
1578 000210 P.ASSN= 210 ;ASSIGNMENT COMMAND SEQUENCE
1579 ;0 INITIAL STATE
1580 ;BIT 0 DRIVE SERIAL NUMBER
1581 ;BIT 1 START SPINDLE
1582 ;BIT 2 PACK ACKNOWLEDGE
1583 ;BIT 3 READ PACK SERIAL NUMBER
1584 ;BIT 4 WRITE PACK COMMAND ISSUED
1585 ;BIT 5 END OF PACK WRITE
1586 ;BIT 6 RECAL
1587 ;BIT 7 RECAL AND RETRY READ PACK
1588 ; SERIAL NUMBER
1589
1590 000211 P.SEEK= 211 ;SEEK TO PREVIOUS CYLINDER FLAG
1591
1592 ; THE FOLLOWING LOCATION IS USED TO INDICATE ERRORS
1593
1594 000212 P.ERR= 212 ;BIT 2 DATA COMPARISON ERROR
1595 ;BIT 3 DRIVE ERROR NOT REPORTED
1596 ;BIT 4 DRIVE SEIZED TIME OUT
1597 ;BIT 5 DRIVE POSITIONING TIME OUT
1598 ;BIT 6 ERROR WHILE ENQUEUED
1599 ;BIT 7 ATTENTION WHEN DRIVE NOT IN USE
1600 ;BIT 8 TIME OUT WHILE WAITING FOR HEADS
1601 ; TO LOAD AFTER ERROR
1602 ;BIT 10 DRIVE NOT READY AFTER START SPINDLE
1603 ;BIT 11 PACK ACKNOWLEDGE DID NOT SET VOLUME
1604 ; VALID
1605
1606 000214 P.ERHD= 214 ;HEADER ADDRESS OF FIRST DATA MISCOMPARE (3 BYTES)
1607 000217 P.ERAD= 217 ;OFFSET COUNT FROM HEADER ADDRESS
1608 000220 P.CMLG= 220 ;COMPARISON LENGTH
1609 000222 P.BFCP= 222 ;BUFFER COMPARE LENGTH
1610
1611 ; THE FOLLOWING LOCATIONS ARE USED FOR DRIVE SERIAL NUMBER,
1612 ; PACK SERIAL NUMBER, AND TRACK EXCLUSIONS
1613
1614 000224 P.SERL= 224 ;DRIVE SERIAL NUMBER
1615 000226 P.PKSR= 226 ;CARTRIDGE SERIAL NUMBER
1616 000232 P.EXAR= 232 ;EXCLUDED AREA INFORMATION
```

```
1617      .SBTTL TRAP CATCHER
1618
1619      000000      .=0
1620      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1621      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1622      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1623      000174      .=174
1624 000174 000000  DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1625 000176 000000  SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
1626      .SBTTL STARTING ADDRESS(ES)
1627 000200 000137 006670  JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1628 000204 000137 006652  JMP RESTR   ;;BRANCH TO RESTART
1629      000214      .=214
1630 000214 000137 006642  JMP PARSRT  ;;GET SYSTEM PARAMETERS AND START TESTING
1631 000220 000137 065460  JMP ..DUMP  ;; *** MEMORY DUMP (DEBUG ONLY)
1632 000224 000700      ..LOW: 700      ;;START OF DUMP
1633 000226 004600      ..HIGH: PATTAB-2 ;;LAST ADDRESS OF DUMP
1634      000240      .=240
1635 000240 000137 006662  JMP START1
1636      001000      .=1000
1637      .SBTTL APT PARAMETER BLOCK
1638
1639      ;*****
1640      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1641      ;*****
1642      001000      .SX=.      ;;SAVE CURRENT LOCATION
1643      000024      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1644 000024 000200  200      ;;FOR APT START UP
1645      000044      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
1646 000044 001000  $APTHDR ;;POINT TO APT HEADER BLOCK
1647      001000      .=.SX      ;;RESET LOCATION COUNTER
1648      ;*****
1649      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1650      ;INTERFACE SPEC.
1651
1652 001000  $APTHD:
1653 001000 000000  $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1654 001002 001170  $MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1655 001004 003000  $TSTM:  .WORD 3000  ;;RUN TIME OF LONGEST TEST
1656 001006 003000  $PASTM: .WORD 3000  ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1657 001010 003000  $UNITM: .WORD 3000  ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1658 001012 000052  .WORD SETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

1659
1660
1661
1662
1663
1664
1665 001100
1666 001100 001100
1667 001100 000000
1668 001102 000
1669 001103 000
1670 001104 000000
1671 001106 000000
1672 001110 000000
1673 001112 000000
1674 001114 000
1675 001115 001
1676 001116 000000
1677 001120 000000
1678 001122 000000
1679 001124 000000
1680 001126 000000
1681 001130 000000
1682 001132 000000
1683 001134 000
1684 001135 000
1685 001136 000000
1686 001140 177570
1687 001142 177570
1688 001144 177560
1689 001146 177562
1690 001150 177564
1691 001152 177566
1692 001154 000
1693 001155 002
1694 001156 012
1695 001157 000
1696 001160 177607 000377
1697 001164 077
1698 001165 015
1699 001166 000012
1700
1701
1702
1703
1704
1705 001170
1706 001170 000000
1707 001172 000000
1708 001174 000000
1709 001176 000000
1710 001200 000000
1711 001202 000000
1712 001204 000000
1713 001206 000000
1714 001210

.SBTTL COMMON TAGS

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

SCMTAG: .=1100

: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)
:CODE FOR BELL
:QUESTION MARK
:CARRIAGE RETURN
:LINE FEED

.SBTTL APT MAILBOX-ETABLE

:EVEN
:APT MAILBOX
:MESSAGE TYPE CODE
:FATAL ERROR NUMBER
:TEST NUMBER
:PASS COUNT
:DEVICE COUNT
:I/O UNIT NUMBER
:MESSAGE ADDRESS
:MESSAGE LENGTH
:APT ENVIRONMENT TABLE

1715	001210	000	\$ENV:	.BYTE	AENV	::ENVIRONMENT BYTE
1716	001211	000	\$ENVM:	.BYTE	AENVM	::ENVIRONMENT MODE BITS
1717	001212	000000	\$SWREG:	.WORD	ASWREG	::APT SWITCH REGISTER
1718	001214	000000	\$USWR:	.WORD	AUSWR	::USER SWITCHES
1719	001216	000000	\$CPUOP:	.WORD	ACPUOP	::CPU TYPE,OPTIONS
1720			::*			BITS 15-11=CPU TYPE
1721			::*			11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1722			::*			11/70=06,P00=07,Q=10
1723			::*			BIT 10=REAL TIME CLOCK
1724			::*			BIT 9=FLOATING POINT PROCESSOR
1725			::*			BIT 8=MEMORY MANAGEMENT
1726	001220	000	\$MAMS1:	.BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
1727	001221	000	\$MTYP1:	.BYTE	AMTYP1	::MEM. TYPE,BLK#1
1728			::*			MEM.TYPE BYTE -- (HIGH BYTE)
1729			::*			900 NSEC CORE=001
1730			::*			300 NSEC BIPOLAR=002
1731			::*			500 NSEC MOS=003
1732	001222	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS,BLK#1
1733			::*			MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1734	001224	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
1735	001225	000	\$MTYP2:	.BYTE	AMTYP2	::MEM. TYPE,BLK#2
1736	001226	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
1737	001230	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
1738	001231	000	\$MTYP3:	.BYTE	AMTYP3	::MEM. TYPE,BLK#3
1739	001232	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
1740	001234	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
1741	001235	000	\$MTYP4:	.BYTE	AMTYP4	::MEM. TYPE,BLK#4
1742	001236	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
1743	001240	120210	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
1744	001242	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
1745	001244	177440	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
1746	001246	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
1747	001250	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
1748	001252	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
1749	001254	000000	\$DDW0:	.WORD	ADDW0	::DEVICE DESCRIPTOR WORD#0
1750	001256	000000	\$DDW1:	.WORD	ADDW1	::DEVICE DESCRIPTOR WORD#1
1751	001260	000000	\$DDW2:	.WORD	ADDW2	::DEVICE DESCRIPTOR WORD#2
1752	001262	000000	\$DDW3:	.WORD	ADDW3	::DEVICE DESCRIPTOR WORD#3
1753	001264	000000	\$DDW4:	.WORD	ADDW4	::DEVICE DESCRIPTOR WORD#4
1754	001266	000000	\$DDW5:	.WORD	ADDW5	::DEVICE DESCRIPTOR WORD#5
1755	001270	000000	\$DDW6:	.WORD	ADDW6	::DEVICE DESCRIPTOR WORD#6
1756	001272	000000	\$DDW7:	.WORD	ADDW7	::DEVICE DESCRIPTOR WORD#7
1757	001274	000000	\$DDW8:	.WORD	ADDW8	::DEVICE DESCRIPTOR WORD#8
1758	001276	000000	\$DDW9:	.WORD	ADDW9	::DEVICE DESCRIPTOR WORD#9
1759	001300	000000	\$DDW10:	.WORD	ADDW10	::DEVICE DESCRIPTOR WORD#10
1760	001302	000000	\$DDW11:	.WORD	ADDW11	::DEVICE DESCRIPTOR WORD#11
1761	001304	000000	\$DDW12:	.WORD	ADDW12	::DEVICE DESCRIPTOR WORD#12
1762	001306	000000	\$DDW13:	.WORD	ADDW13	::DEVICE DESCRIPTOR WORD#13
1763	001310	000000	\$DDW14:	.WORD	ADDW14	::DEVICE DESCRIPTOR WORD#14
1764	001312	000000	\$DDW15:	.WORD	ADDW15	::DEVICE DESCRIPTOR WORD#15
1765						
1766						
1767	001314		SETEND:			
1768						

```
1769 .SBTTL QUEUE HEAD AND TAIL ALLOCATIONS
1770
1771 001314 000000 000000 AVAILQ: .WORD 0,0 :HEAD AND TAIL OF AVAILABLE QUEUE
1772 001320 000000 000000 BWAITQ: .WORD 0,0 :HEAD AND TAIL OF BUFFER WAIT QUEUE
1773 001324 000000 000000 CINITQ: .WORD 0,0 :HEAD AND TAIL OF COMMAND INITIATION QUEUE
1774 001330 000000 000000 ERRPRQ: .WORD 0,0 :HEAD AND TAIL OF ERROR PROCESSING QUEUE
1775
1776 .SBTTL SYSTEM PARAMETERS
1777
1778 001334 172540 SLKCSR: .WORD 172540 :ADDRESS OF KW11-P STATUS REGISTER
1779 001336 172542 SLKCSB: .WORD 172542 :ADDRESS OF KW11-P COUNTER BUFFER
1780 001340 000104 SLPVEC: .WORD 104 :KW11-P VECTOR ADDRESS
1781 001342 177546 SLKS: .WORD 177546 :ADDRESS OF KW11-L STATUS REGISTER
1782 001344 000100 SLLVEC: .WORD 100 :KW11-L VECTOR ADDRESS
1783 001346 177514 $LSCS: .WORD 177514 :ADDRESS OF PRINTER STATUS WORD
1784 001350 177516 $LSDB: .WORD 177516 :ADDRESS OF PRINTER DATA BUFFER
1785 001352 074 HZ: .BYTE 60. :74 (60 DECIMAL) IF SYSTEM IS 60 HZ.
:62 (50 DECIMAL) IF SYSTEM IS 50 HZ.
1786
1787 001353 000 CLKFRQ: .BYTE 0 :COUNT FOR SECOND
1788 001354 013400 MAXBUF: .WORD 256.*23. :MAXIMUM BUFFER FOR READ OR WRITE
1789 001356 177777 PASCNT: .WORD 177777 :PASS COUNT
1790 001360 000 PERINV: .BYTE 0 :INTERVAL BETWEEN PERFORMANCE SUMMARIES
1791 001361 000 FLAG: .BYTE 0 :RESTART AND PARAMETER FLAG
1792 001362 000003 SOFCMP: .WORD 3 :NUMBER OF SOFTWARE COMPARES
1793 001364 000 CPUID: .BYTE 0 :CPU DESIGNATION FOR PACK WRITING WITH DUAL ACCESS
1794 001365 000 OVLYLD: .BYTE 0 :OVERLAY LOADER
1795 001366 000000 HOUR: .WORD 0 :TIMER HOUR
1796 001370 000000 MINUTE: .WORD 0 :TIMER MINUTE
1797 001372 000000 SECOND: .WORD 0 :TIMER SECOND
1798 001374 000 CLKFLG: .BYTE 0 :KW11-P OR KW11-L PRESENT
1799 001375 000 PERIOD: .BYTE 0 :PERIOD SINCE LAST STATISTIC PRINT OUT
1800 001376 000000 TIMHR: .WORD 0 :HOUR COUNT FOR PRINT OUT
1801 001400 000000 TIMMIN: .WORD 0 :MINUTE COUNT FOR PRINT OUT
1802 001402 000000 TIMSEC: .WORD 0 :SECOND COUNT FOR PRINT OUT
1803 001404 000000 SAVSWR: .WORD 0 :SAVED SWITCH REGISTER FROM POWER FAIL
1804 001406 000000 RELFLG: .WORD 0 :FLAG FOR RELEASE DURING WRITE PACK
1805
1806 .SBTTL TEMPORARY CONTROLLER REGISTER STORAGE
1807
1808 001410 000000 T.CS1: .WORD 0 :TEMPORARY STORAGE FOR COMMAND AND STATUS
1809 : REGISTER 1
1810 001412 000000 T.CS2: .WORD 0 :TEMPORARY STORAGE FOR COMMAND AND STATUS
1811 : REGISTER 2
1812 001414 000000 T.WCR: .WORD 0 :TEMPORARY STORAGE FOR WORD COUNT REGISTER
1813 001416 000000 T.BA: .WORD 0 :TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
1814 001420 000000 T.DA: .WORD 0 :TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
1815 001422 000000 T.DC: .WORD 0 :TEMPORARY STORAGE FOR DRIVE CYLINDER
1816 001424 000000 T.ASOF: .WORD 0 :TEMPORARY STORAGE FOR ATTENTION SUMMARY
1817 : AND OFFSET
1818 001426 000000 T.ER: .WORD 0 :TEMPORARY STORAGE FOR ERROR REGISTER
1819 001430 000000 T.DS: .WORD 0 :TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
1820 001432 000000 T.MR1: .WORD 0 :TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
1821 001434 000000 T.MR2: .WORD 0 :TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
1822 001436 000000 T.MR3: .WORD 0 :TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
1823 001440 000000 T.POS: .WORD 0 :TEMPORARY STORAGE FOR ECC POSITION
1824 001442 000000 T.PAT: .WORD 0 :TEMPORARY STORAGE FOR ECC PATTERN
```

1825	001444	000000	T.DB:	.WORD	0	:TEMPORARY STORAGE FOR DATA BUFFER REGISTER
1826						
1827			.SBTTL	DRIVER	PARAMERTERS	
1828						
1829	001446	177440	RKBAS:	.WORD	177440	:ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
1830	001450	000210	RKVEC:	.WORD	210	:ADDRESS OF R611 VECTOR
1831	001452	000240	RKPRI:	.WORD	PR5	:RK611 INTERRUPT PRIORITY
1832	001454	024400	A.NORM:	NORMAL		:ADDRESS OF NORMAL RETURN FROM DRIVER
1833	001456	025470	A.ABNL:	ABNORM		:ADDRESS OF ABNORMAL RETURN FROM DRIVER
1834	001460	026524	A.CONT:	CONTRL		:ADDRESS OF CONTROLLER ERROR RETURN
1835	001462	000000	E.CONT:	.WORD	0	:CONTROLLER ERROR STATUS
1836						: THIS LOCATION IS CLEARED WHEN EVERY COMMAND
1837						: IS INITIATED. IF A CONTROLLER ERROR
1838						: OCCURS THE FOLLOWING BIT ASSIGNMENT IS
1839						: USED:
1840						
1841		000001	E.CCLR=	BIT0		:CLEAR CONTROLLER DID NOT CLEAR ERROR
1842		000002	E.NOAT=	BIT1		:NO ATTENTION IN ATTENTION SUMMARY REG
1843		000004	E.UATT=	BIT2		:UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
1844		000010	E.UDAT=	BIT3		:UNEXPECTED DATA TYPE ERROR
1845		000020	E.CLAT=	BIT4		:ATTENTION DID NOT RESET WITH CLEAR
1846		000040	E.SCLR=	BIT5		:SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
1847						: ATTENTION
1848		000100	E.ILLD=	BIT6		:ILLEGAL DRIVER COMMAND
1849		000400	E.DLT=	BIT8		:DATA LATE WHEN UNLOADING HEADER
1850		001000	E.CERR=	BIT9		:CONTROLLER ERROR DURING DRIVER SERVICING
1851		002000	E.DPAR=	BIT10		:DRIVE DETECTED PARITY ERROR
1852		040000	E.CMTO=	BIT14		:CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
1853		100000	E.MDS=	BIT15		:MULTIPLE DRIVE SELECT
1854						
1855	001464	000000	O.WAIT:	.WORD	0	:PARAMETER BLOCK OF THE DRIVE
1856						:WAITING FOR COMMAND COMPLETION
1857	001466	000400	W.MTIM:	.WORD	400	:LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
1858	001470	000400	W.MILI:	.WORD	400	:16 MILLISECOND TIME FOR PROGRAM
1859						
1860						
1861						
1862						
1863						
1864						
1865						
1866						
1867						
1868						
1869						
1870						
1871	001472	000300	W.SEC:	.WORD	300	:SECOND COUNT COUNT FOR ALL COMMANDS
1872						: EXCEPT START SPINDLE
1873	001474	003000	W.8SEC:	.WORD	3000	:8 SECOND FOR DRIVE CYCLE DOWN
1874	001476	030000	W.MIN:	.WORD	30000	:MINUTE TIME FOR START SPINDLE
1875	001500	000000	OPTFLG:	.WORD	0	:FLAG = 1 WHILE IN CMND OPTIMIZER
1876	001502	000000	HDR.AD:	.WORD	0	:ADDRESS USED FOR READ ALL HEADERS
1877	001504	000000	HDR.CT:	.WORD	0	:NUMBER OF HEADERS LEFT TO READ FOR READ
1878						: ALL HEADERS
1879	001506	000	I.ISRL:	.BYTE	0	:INTERRUPT OR RELEASED COMMAND ISSUED
1880	001507	002	H.HEAD:	.BYTE	2,4,10	:HEAD DECODES

1881 001512 000
1882 001513 377

W.TIME: .BYTE 0 :DRIVES BEING WATCH-DOG TIMED
O.OVER: .BYTE -1 :OVERLAPPED OPERATIONS

1883
1884
1885

.SBTTL TABLE OF INTERRUPT MASKS

1886 001514 001
1887 001515 002
1888 001516 004
1889 001517 010
1890 001520 020
1891 001521 040
1892 001522 100
1893 001523 200

INTMSK: .BYTE 1 :INTERRUPT FOR DRIVE 0
.BYTE 2 :INTERRUPT FOR DRIVE 1
.BYTE 4 :INTERRUPT FOR DRIVE 2
.BYTE 10 :INTERRUPT FOR DRIVE 3
.BYTE 20 :INTERRUPT FOR DRIVE 4
.BYTE 40 :INTERRUPT FOR DRIVE 5
.BYTE 100 :INTERRUPT FOR DRIVE 6
.BYTE 200 :INTERRUPT FOR DRIVE 7

1894
1895
1896

.SBTTL PARAMETER BLOCK TABLE

1897 001524 001736
1898 001526 002214
1899 001530 002472
1900 001532 002750
1901 001534 003226
1902 001536 003504
1903 001540 003762
1904 001542 004240

PBLKT: PARM0 :ADDRESS OF DRIVE 0'S PARAMETER BLOCK
PARM1 :ADDRESS OF DRIVE 1'S PARAMETER BLOCK
PARM2 :ADDRESS OF DRIVE 2'S PARAMETER BLOCK
PARM3 :ADDRESS OF DRIVE 3'S PARAMETER BLOCK
PARM4 :ADDRESS OF DRIVE 4'S PARAMETER BLOCK
PARM5 :ADDRESS OF DRIVE 5'S PARAMETER BLOCK
PARM6 :ADDRESS OF DRIVE 6'S PARAMETER BLOCK
PARM7 :ADDRESS OF DRIVE 7'S PARAMETER BLOCK

1905
1906
1907

.SBTTL WATCH-DOG TIMER COUNTS

1908 001544 000000
1909 001546 000000
1910 001550 000000
1911 001552 000000
1912 001554 000000
1913 001556 000000
1914 001560 000000
1915 001562 000000

W.DRV: .WORD 0 :WATCH-DOG FOR DRIVE 0
.WORD 0 :WATCH-DOG FOR DRIVE 1
.WORD 0 :WATCH-DOG FOR DRIVE 2
.WORD 0 :WATCH-DOG FOR DRIVE 3
.WORD 0 :WATCH-DOG FOR DRIVE 4
.WORD 0 :WATCH-DOG FOR DRIVE 5
.WORD 0 :WATCH-DOG FOR DRIVE 6
.WORD 0 :WATCH-DOG FOR DRIVE 7

.SBTTL PERFORMANCE EXERCISER STATUS INFORMATION

1916							
1917							
1918	001564	000000		ERCONT: .WORD	0	:CONTROLLER ERROR (RECOVERABLE)	
1919						:BIT 0 CONTROLLER ERROR NOT FLAGGED	
1920						:BIT 1 WORD COUNT NOT EQUAL 0	
1921						:BIT 2 BUS ADDRESS INCORRECT	
1922						:BIT 3 CYLINDER, TRACK, SECTOR INCORRECT	
1923						:BIT 15 CONTROLLER ERROR DETECTED	
1924							
1925	001566	000000		ERRPRO: .WORD	0	:ADDRESS OF PARAMETER BLOCK PROCESSING ERROR	
1926	001570	000000		TMPCNT: .WORD	0	:TRANSFER COUNT CALCULATIONS	
1927	001572	000000	000000	TMPTRK: .WORD	0,0	:RANDOM TRACK CALCULATIONS	
1928	001576	000000	000000	TMPSEC: .WORD	0,0	:RANDOM SECTOR CALCULATIONS	
1929	001602	000000	000000	PARCYL: .WORD	0,0	:PARAMETER CYLINDER MODIFICATION	
1930	001606	000000	000000	PARTRK: .WORD	0,0	:PARAMETER TRACK MODIFICATION	
1931	001612	000000	000000	PARSEC: .WORD	0,0	:PARAMETER SECTOR MODIFICATION	
1932	001616	000000	000000	EXAREA: .WORD	0,0	:EXCLUDED AREA CALCULATIONS	
1933	001622	000000	000000	ECCPAT: .WORD	0,0	:ECC PATTERN FOR ECC CORRECTION	
1934	001626	000000		DLTCNT: .WORD	0	:DATA LATE COUNT	
1935	001630	000000		CNTCNT: .WORD	0	:OTHER CONTROLLER ERRORS COUNT	
1936	001632	000		DRVCNT: .BYTE	0	:DRIVE COUNT FOR OPERATOR COMMAND	
1937	001633	000		STATIS: .BYTE	0	:REPORT INTERVAL STATISTICS	
1938	001634	000000		RTYBA: .WORD	0	:RETRY BUS ADDRESS	
1939	001636	000000		RTYWC: .WORD	0	:RETRY WORD COUNT	
1940	001640	000		RTYSEC: .BYTE	0	:SECTOR ADDRESS RETRY	
1941	001641	000		RTYTRK: .BYTE	0	:TRACK ADDRESS RETRY	
1942	001642	000000		RTYCYL: .WORD	0	:CYLINDER ADDRESS RETRY	
1943	001644	000		RTYSCN: .BYTE	0	:NUMBER OF SECTORS TO BE TRANSFERRED IN RETRY	
1944	001645	000		RTYCMD: .BYTE	0	:COMMAND BEING RETRIED	
1945	001646	000000		HEAD1: .WORD	0	:TEMPORARY STORAGE FOR FIRST WORD OF HEADER	
1946	001650	000000		HEAD2: .WORD	0	:TEMPORARY STORAGE FOR SECOND WORD OF HEADER	
1947	001652	000000		HEAD3: .WORD	0	:TEMPORARY STORAGE FOR THIRD WORD OF HEADER	
1948	001654	000		BSSECT: .BYTE	0	:SECTOR IN BAD SECTOR SERVICING	
1949	001655	000		BSTRCK: .BYTE	0	:TRACK IN BAD SECTOR SERVICING	
1950	001656	000000		BSCYLN: .WORD	0	:CYLINDER IN BAD SECTOR SERVICING	
1951	001660	000000		BSWC: .WORD	0	:WORD COUNT IN BAD SECTOR SERVICING	
1952	001662	000000		BSBA: .WORD	0	:BUS ADDRESS IN BAD SECTOR SERVICING	
1953	001664	000000		BSWORD: .WORD	0	:WORDS TRANSFERRED IN BAD SECTOR SERVICING	
1954	001666	000000		POSCMD: .WORD	0	:POSITIONING TIME OUT COMMAND	

.SBTTL FREE SPACE BLOCK

1955							
1956							
1957							
1958	001670	000001		FBLKC: .WORD	1	:FREE BLOCK COUNT	
1959							
1960	001672	000000	000000	FBLKT: .WORD	0,0	:FREE BLOCK 0	
1961	001676	000000	000000		0,0	:FREE BLOCK 1	
1962	001702	000000	000000		0,0	:FREE BLOCK 2	
1963	001706	000000	000000		0,0	:FREE BLOCK 3	
1964	001712	000000	000000		0,0	:FREE BLOCK 4	
1965	001716	000000	000000		0,0	:FREE BLOCK 5	
1966	001722	000000	000000		0,0	:FREE BLOCK 6	
1967	001726	000000	000000		0,0	:FREE BLOCK 7	
1968	001732	000000	000000		0,0	:FREE BLOCK 8	

1969					
1970					
1971					
1972	001736	000			
1973	001737	000			
1974	001740	000000			
1975	001742	000			
1976	001743	000			
1977	001744	000			
1978	001745	000			
1979	001746	000000			
1980	001750	000000			
1981	001752	000000			
1982	001754	000000			
1983	001756	000000			
1984	001760	000000			
1985	001762	000000			
1986	001764	000000			
1987	001766	000000			
1988	001770	000000			
1989	001772	000000			
1990	001774	000000			
1991	001776	000000			
1992	002000	000000			
1993	002002	000000			
1994	002004	000000			
1995	002006	000000			
1996	002010	000000			
1997	002012	000000			
1998	002014	000000			
1999	002016	000000			
2000	002020	000000			
2001	002022	000000			
2002	002024	000000			
2003	002026	000632			
2004	002030	000			
2005	002031	002			
2006	002032	000			
2007	002033	025			
2008	002034	003			
2009	002035	000			
2010	002036	000012			
2011	002040	000005			
2012	002042	000005			
2013	002044	177770	177777		
2014	002050	177770	177777		
2015	002054	000			
2016	002055	000			
2017	002056	000			
2018	002057	000			
2019	002060	000			
2020	002061	000			
2021	002062	000000			
2022	002064	000			
2023	002065	000			
2024	002066	000000			

.SBTTL PARAMETER BLOCK FOR DRIVE 0		
PARMO:	.BYTE 0	:DRIVE 0
	.BYTE 0	:COMMAND
	.WORD 0	:CYLINDER ADDRESS
	.BYTE 0	:SECTOR ADDRESS
	.BYTE 0	:TRACK ADDRESS
	.BYTE 0	:OFFSET VALUE
	.BYTE 0	:BUS ADDRESS (BITS 16 AND 17)
	.WORD 0	:BUS ADDRESS (BITS 0 - 15)
	.WORD 0	:WORD COUNT (2'S COMPLEMENT)
	.WORD 0	:PROGRAM DRIVE STATUS INFORMATION
	.WORD 0	:COMMAND AND STATUS REGISTER 1
	.WORD 0	:COMMAND AND STATUS REGISTER 2
	.WORD 0	:WORD COUNT REGISTER
	.WORD 0	:BUS ADDRESS REGISTER
	.WORD 0	:DESIRED TRACK AND SECTOR REGISTER
	.WORD 0	:DESIRED CYLINDER REGISTER
	.WORD 0	:ATTENTION SUMMARY/OFFSET REGISTER
	.WORD 0	:ERROR REGISTER
	.WORD 0	:STATUS REGISTER
	.WORD 0	:MESSAGE LINE A STATUS BYTE 00
	.WORD 0	:MESSAGE LINE B STATUS BYTE 00
	.WORD 0	:MESSAGE LINE A STATUS BYTE 01
	.WORD 0	:MESSAGE LINE B STATUS BYTE 01
	.WORD 0	:MESSAGE LINE A STATUS BYTE 10
	.WORD 0	:MESSAGE LINE B STATUS BYTE 10
	.WORD 0	:MESSAGE LINE A STATUS BYTE 11
	.WORD 0	:MESSAGE LINE B STATUS BYTE 11
	.WORD 0	:ECC POSITION INFORMATION
	.WORD 0	:ECC PATTERN INFORMATION
	.WORD 0	:QUEUE LINK
	.WORD 0	:MINIMUM CYLINDER
	.WORD 410.	:MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
	.BYTE 0	:MINIMUM TRACK
	.BYTE 2	:MAXIMUM TRACK
	.BYTE 0	:MINIMUM SECTOR
	.BYTE 21.	:MAXIMUM SECTOR
	.BYTE 3	:READ/WRITE RATIO
	.BYTE 0	:AUTOMATIC WRITE CHECK
	.WORD 10.	:CORRECTABLE READ ERROR THRESHOLD
	.WORD 5	:UNCORRECTABLE READ ERROR THRESHOLD
	.WORD 5	:SEEK ERROR THRESHOLD
	.WORD 177770,177777	:MAXIMUM NUMBER OF COMMANDS TO DRIVE
	.WORD 177770,177777	:MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
	.BYTE 0	:SAMPLE COMPARE FLAG
	.BYTE 0	:ECC COMPARE FLAG
	.BYTE 0	:INHIBIT ERROR CORRECTION
	.BYTE 0	:DATA PATTERN USED
	.BYTE 0	:CURRENT GENERATED DATA PATTERN
	.BYTE 0	:CURRENT RANDOMLY GENERATED COMMAND
	.WORD 0	:CURRENT RANDOMLY GENERATED CYLINDER
	.BYTE 0	:CURRENT RANDOMLY GENERATED SECTOR
	.BYTE 0	:CURRENT RANDOMLY GENERATED TRACK
	.WORD 0	:CURRENT RANDOMLY GENERATED WORD COUNT

2025	002070	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2026	002072	000			.BYTE	0		
2027	002073	000			.BYTE	0		:LAST COMMAND
2028	002074	000000			.WORD	0		:LAST CYLINDER
2029	002076	000			.BYTE	0		:LAST SECTOR
2030	002077	000			.BYTE	0		:LAST TRACK
2031	002100	000000			.WORD	0		:LAST WORD COUNT
2032	002102	000			.BYTE	0		:LAST DATA PATTERN USED
2033	002103	000			.BYTE	0		:BUFFER ALLOCATED
2034	002104	000			.BYTE	0		:RETRY COUNT
2035	002105	000			.BYTE	0		:REREAD STATUS
2036	002106	000000			.WORD	0		:DRIVE STATUS FLAGS
2037	002110	000000	000000		.WORD	0.0		:NUMBER OF ORDERS
2038	002114	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS WRITTEN
2039	002122	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS READ
2040	002130	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2041	002132	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2042	002134	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2043	002136	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2044	002140	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2045	002142	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2046	002144	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2047	002146	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2048	002147	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2049	002150	000000			.WORD	0		:ERROR STATUS INFORMATION
2050	002152	000000			.WORD	C		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2051	002154	000			.BYTE	0		
2052	002155	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2053	002156	000000			.WORD	0		:COMPARISON LENGTH
2054	002160	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2055	002162	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2056	002164	000000	000000		.WORD	0.0		:CARIAGE SERIAL NUMBER
2057	002170	000000	000000		.WORD	0.0		:FIRST EXCLUSION AREA
2058	002174	000000	000000		.WORD	0.0		:SECOND EXCLUSION AREA
2059	002200	000000	000000		.WORD	0.0		:THIRD EXCLUSION AREA
2060	002204	000000	000000		.WORD	0.0		:FOURTH EXCLUSION AREA
2061	002210	000000	000000		.WORD	0.0		:FIFTH EXCLUSION AREA

```
2062  
2063  
2064  
2065 002214 001  
2066 002215 000  
2067 002216 000000  
2068 002220 000  
2069 002221 000  
2070 002222 000  
2071 002223 000  
2072 002224 000000  
2073 002226 000000  
2074 002230 000000  
2075 002232 000000  
2076 002234 000000  
2077 002236 000000  
2078 002240 000000  
2079 002242 000000  
2080 002244 000000  
2081 002246 000000  
2082 002250 000000  
2083 002252 000000  
2084 002254 000000  
2085 002256 000000  
2086 002260 000000  
2087 002262 000000  
2088 002264 000000  
2089 002266 000000  
2090 002270 000000  
2091 002272 000000  
2092 002274 000000  
2093 002276 000000  
2094 002300 000000  
2095 002302 000000  
2096 002304 000632  
2097 002306 000  
2098 002307 002  
2099 002310 000  
2100 002311 025  
2101 002312 003  
2102 002313 000  
2103 002314 000012  
2104 002316 000005  
2105 002320 000005  
2106 002322 177770 177777  
2107 002326 177770 177777  
2108 002332 000  
2109 002333 000  
2110 002334 000  
2111 002335 000  
2112 002336 000  
2113 002337 000  
2114 002340 000000  
2115 002342 000  
2116 002343 000  
2117 002344 000000
```

.SBTTL PARAMETER BLOCK FOR DRIVE 1

PARM1: .BYTE 1 :DRIVE 1
.BYTE 0 :COMMAND
.WORD 0 :CYLINDER ADDRESS
.BYTE 0 :SECTOR ADDRESS
.BYTE 0 :TRACK ADDRESS
.BYTE 0 :OFFSET VALUE
.BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
.WORD 0 :BUS ADDRESS (BITS 0 - 15)
.WORD 0 :WORD COUNT (2'S COMPLEMENT)
.WORD 0 :PROGRAM DRIVE STATUS INFORMATION
.WORD 0 :COMMAND AND STATUS REGISTER 1
.WORD 0 :COMMAND AND STATUS REGISTER 2
.WORD 0 :WORD COUNT REGISTER
.WORD 0 :BUS ADDRESS REGISTER
.WORD 0 :DESIRED TRACK AND SECTOR REGISTER
.WORD 0 :DESIRED CYLINDER REGISTER
.WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
.WORD 0 :ERROR REGISTER
.WORD 0 :STATUS REGISTER
.WORD 0 :MESSAGE LINE A STATUS BYTE 00
.WORD 0 :MESSAGE LINE B STATUS BYTE 00
.WORD 0 :MESSAGE LINE A STATUS BYTE 01
.WORD 0 :MESSAGE LINE B STATUS BYTE 01
.WORD 0 :MESSAGE LINE A STATUS BYTE 10
.WORD 0 :MESSAGE LINE B STATUS BYTE 10
.WORD 0 :MESSAGE LINE A STATUS BYTE 11
.WORD 0 :MESSAGE LINE B STATUS BYTE 11
.WORD 0 :ECC POSITION INFORMATION
.WORD 0 :ECC PATTERN INFORMATION
.WORD 0 :QUEUE LINK
.WORD 0 :MINIMUM CYLINDER
.WORD 410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
.BYTE 0 :MINIMUM TRACK
.BYTE 2 :MAXIMUM TRACK
.BYTE 0 :MINIMUM SECTOR
.BYTE 21. :MAXIMUM SECTOR
.BYTE 3 :READ/WRITE RATIO
.WORD 0 :AUTOMATIC WRITE CHECK
.WORD 10. :CORRECTABLE READ ERROR THRESHOLD
.WORD 5 :UNCORRECTABLE READ ERROR THRESHOLD
.WORD 5 :SEEK ERROR THRESHOLD
.WORD 177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
.WORD 177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
.BYTE 0 :SAMPLE COMPARE FLAG
.BYTE 0 :ECC COMPARE FLAG
.BYTE 0 :INHIBIT ERROR CORRECTION
.BYTE 0 :DATA PATTERN USED
.BYTE 0 :CURRENT GENERATED DATA PATTERN
.BYTE 0 :CURRENT RANDOMLY GENERATED COMMAND
.WORD 0 :CURRENT RANDOMLY GENERATED CYLINDER
.BYTE 0 :CURRENT RANDOMLY GENERATED SECTOR
.BYTE 0 :CURRENT RANDOMLY GENERATED TRACK
.WORD 0 :CURRENT RANDOMLY GENERATED WORD COUNT

2118	002346	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2119	002350	000			.BYTE	0		
2120	002351	000			.BYTE	0		:LAST COMMAND
2121	002352	000000			.WORD	0		:LAST CYLINDER
2122	002354	000			.BYTE	0		:LAST SECTOR
2123	002355	000			.BYTE	0		:LAST TRACK
2124	002356	000000			.WORD	0		:LAST WORD COUNT
2125	002360	000			.BYTE	0		:LAST DATA PATTERN USED
2126	002361	000			.BYTE	0		:BUFFER ALLOCATED
2127	002362	000			.BYTE	0		:RETRY COUNT
2128	002363	000			.BYTE	0		:REREAD STATUS
2129	002364	000000			.WORD	0		:DRIVE STATUS FLAGS
2130	002366	000000	000000		.WORD	0.0		:NUMBER OF ORDERS
2131	002372	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS WRITTEN
2132	002400	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS READ
2133	002406	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2134	002410	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2135	002412	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2136	002414	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2137	002416	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2138	002420	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2139	002422	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2140	002424	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2141	002425	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2142	002426	000000			.WORD	0		:ERROR STATUS INFORMATION
2143	002430	000000			.WORD	0		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2144	002432	000			.BYTE	0		
2145	002433	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2146	002434	000000			.WORD	0		:COMPARISON LENGTH
2147	002436	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2148	002440	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2149	002442	000000	000000		.WORD	0.0		:CARIAGE SERIAL NUMBER
2150	002446	000000	000000		.WORD	0.0		:FIRST EXCLUSION AREA
2151	002452	000000	000000		.WORD	0.0		:SECOND EXCLUSION AREA
2152	002456	000000	000000		.WORD	0.0		:THIRD EXCLUSION AREA
2153	002462	000000	000000		.WORD	0.0		:FOURTH EXCLUSION AREA
2154	002466	000000	000000		.WORD	0.0		:FIFTH EXCLUSION AREA

```

2155
2156
2157
2158 002472 002
2159 002473 000
2160 002474 000000
2161 002476 000
2162 002477 000
2163 002500 000
2164 002501 000
2165 002502 000000
2166 002504 000000
2167 002506 000000
2168 002510 000000
2169 002512 000000
2170 002514 000000
2171 002516 000000
2172 002520 000000
2173 002522 000000
2174 002524 000000
2175 002526 000000
2176 002530 000000
2177 002532 000000
2178 002534 000000
2179 002536 000000
2180 002540 000000
2181 002542 000000
2182 002544 000000
2183 002546 000000
2184 002550 000000
2185 002552 000000
2186 002554 000000
2187 002556 000000
2188 002560 000000
2189 002562 000632
2190 002564 000
2191 002565 002
2192 002566 000
2193 002567 025
2194 002570 003
2195 002571 000
2196 002572 000012
2197 002574 000005
2198 002576 000005
2199 002600 177770 177777
2200 002604 177770 177777
2201 002610 000
2202 002611 000
2203 002612 000
2204 002613 000
2205 002614 000
2206 002615 000
2207 002616 000000
2208 002620 000
2209 002621 000
2210 002622 000000

```

.SBTTL PARAMETER BLOCK FOR DRIVE 2

```

PARM2: .BYTE 2 :DRIVE 2
        .BYTE 0 :COMMAND
        .WORD 0 :CYLINDER ADDRESS
        .BYTE 0 :SECTOR ADDRESS
        .BYTE 0 :TRACK ADDRESS
        .BYTE 0 :OFFSET VALUE
        .BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
        .WORD 0 :BUS ADDRESS (BITS 0 - 15)
        .WORD 0 :WORD COUNT (2'S COMPLEMENT)
        .WORD 0 :PROGRAM DRIVE STATUS INFORMATION
        .WORD 0 :COMMAND AND STATUS REGISTER 1
        .WORD 0 :COMMAND AND STATUS REGISTER 2
        .WORD 0 :WORD COUNT REGISTER
        .WORD 0 :BUS ADDRESS REGISTER
        .WORD 0 :DESIRED TRACK AND SECTOR REGISTER
        .WORD 0 :DESIRED CYLINDER REGISTER
        .WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
        .WORD 0 :ERROR REGISTER
        .WORD 0 :STATUS REGISTER
        .WORD 0 :MESSAGE LINE A STATUS BYTE 00
        .WORD 0 :MESSAGE LINE B STATUS BYTE 00
        .WORD 0 :MESSAGE LINE A STATUS BYTE 01
        .WORD 0 :MESSAGE LINE B STATUS BYTE 01
        .WORD 0 :MESSAGE LINE A STATUS BYTE 10
        .WORD 0 :MESSAGE LINE B STATUS BYTE 10
        .WORD 0 :MESSAGE LINE A STATUS BYTE 11
        .WORD 0 :MESSAGE LINE B STATUS BYTE 11
        .WORD 0 :ECC POSITION INFORMATION
        .WORD 0 :ECC PATTERN INFORMATION
        .WORD 0 :QUEUE LINK
        .WORD 0 :MINIMUM CYLINDER
        .WORD 410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
        .BYTE 0 :MINIMUM TRACK
        .BYTE 2 :MAXIMUM TRACK
        .BYTE 0 :MINIMUM SECTOR
        .BYTE 21. :MAXIMUM SECTOR
        .BYTE 3 :READ/WRITE RATIO
        .BYTE 0 :AUTOMATIC WRITE CHECK
        .WORD 10. :CORRECTABLE READ ERROR THRESHOLD
        .WORD 5 :UNCORRECTABLE READ ERROR THRESHOLD
        .WORD 5 :SEEK ERROR THRESHOLD
        .WORD 177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
        .WORD 177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
        .BYTE 0 :SAMPLE COMPARE FLAG
        .BYTE 0 :ECC COMPARE FLAG
        .BYTE 0 :INHIBIT ERROR CORRECTION
        .BYTE 0 :DATA PATTERN USED
        .BYTE 0 :CURRENT GENERATED DATA PATTERN
        .BYTE 0 :CURRENT RANDOMLY GENERATED COMMAND
        .WORD 0 :CURRENT RANDOMLY GENERATED CYLINDER
        .BYTE 0 :CURRENT RANDOMLY GENERATED SECTOR
        .BYTE 0 :CURRENT RANDOMLY GENERATED TRACK
        .WORD 0 :CURRENT RANDOMLY GENERATED WORD COUNT

```

2211	002624	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2212	002626	000			.BYTE	0		
2213	002627	000			.BYTE	0		:LAST COMMAND
2214	002630	000000			.WORD	0		:LAST CYLINDER
2215	002632	000			.BYTE	0		:LAST SECTOR
2216	002633	000			.BYTE	0		:LAST TRACK
2217	002634	000000			.WORD	0		:LAST WORD COUNT
2218	002636	000			.BYTE	0		:LAST DATA PATTERN USED
2219	002637	000			.BYTE	0		:BUFFER ALLOCATED
2220	002640	000			.BYTE	0		:RETRY COUNT
2221	002641	000			.BYTE	0		:REREAD STATUS
2222	002642	000000			.WORD	0		:DRIVE STATUS FLAGS
2223	002644	000000	000000		.WORD	0.0		:NUMBER OF ORDERS
2224	002650	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS WRITTEN
2225	002656	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS READ
2226	002664	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2227	002666	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2228	002670	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2229	002672	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2230	002674	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2231	002676	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2232	002700	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2233	002702	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2234	002703	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2235	002704	000000			.WORD	0		:ERROR STATUS INFORMATION
2236	002706	000000			.WORD	0		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2237	002710	000			.BYTE	0		
2238	002711	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2239	002712	000000			.WORD	0		:COMPARISON LENGTH
2240	002714	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2241	002716	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2242	002720	000000	000000		.WORD	0.0		:CARIAGE SERIAL NUMBER
2243	002724	000000	000000		.WORD	0.0		:FIRST EXCLUSION AREA
2244	002730	000000	000000		.WORD	0.0		:SECOND EXCLUSION AREA
2245	002734	000000	000000		.WORD	0.0		:THIRD EXCLUSION AREA
2246	002740	000000	000000		.WORD	0.0		:FOURTH EXCLUSION AREA
2247	002744	000000	000000		.WORD	0.0		:FIFTH EXCLUSION AREA

```

2248
2249
2250
2251 002750 003
2252 002751 000
2253 002752 000000
2254 002754 000
2255 002755 000
2256 002756 000
2257 002757 000
2258 002760 000000
2259 002762 000000
2260 002764 000000
2261 002766 000000
2262 002770 000000
2263 002772 000000
2264 002774 000000
2265 002776 000000
2266 003000 000000
2267 003002 000000
2268 003004 000000
2269 003006 000000
2270 003010 000000
2271 003012 000000
2272 003014 000000
2273 003016 000000
2274 003020 000000
2275 003022 000000
2276 003024 000000
2277 003026 000000
2278 003030 000000
2279 003032 000000
2280 003034 000000
2281 003036 000000
2282 003040 000632
2283 003042 000
2284 003043 002
2285 003044 000
2286 003045 025
2287 003046 003
2288 003047 000
2289 003050 000012
2290 003052 000005
2291 003054 000005
2292 003056 177770 177777
2293 003062 177770 177777
2294 003066 000
2295 003067 000
2296 003070 000
2297 003071 000
2298 003072 000
2299 003073 000
2300 003074 000000
2301 003076 000
2302 003077 000
2303 003100 000000

```

.SBTTL PARAMETER BLOCK FOR DRIVE 3

```

PARM3: .BYTE 3 :DRIVE 3
        .BYTE 0 :COMMAND
        .WORD 0 :CYLINDER ADDRESS
        .BYTE 0 :SECTOR ADDRESS
        .BYTE 0 :TRACK ADDRESS
        .BYTE 0 :OFFSET VALUE
        .BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
        .WORD 0 :BUS ADDRESS (BITS 0 - 15)
        .WORD 0 :WORD COUNT (2'S COMPLEMENT)
        .WORD 0 :PROGRAM DRIVE STATUS INFORMATION
        .WORD 0 :COMMAND AND STATUS REGISTER 1
        .WORD 0 :COMMAND AND STATUS REGISTER 2
        .WORD 0 :WORD COUNT REGISTER
        .WORD 0 :BUS ADDRESS REGISTER
        .WORD 0 :DESIRED TRACK AND SECTOR REGISTER
        .WORD 0 :DESIRED CYLINDER REGISTER
        .WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
        .WORD 0 :ERROR REGISTER
        .WORD 0 :STATUS REGISTER
        .WORD 0 :MESSAGE LINE A STATUS BYTE 00
        .WORD 0 :MESSAGE LINE B STATUS BYTE 00
        .WORD 0 :MESSAGE LINE A STATUS BYTE 01
        .WORD 0 :MESSAGE LINE B STATUS BYTE 01
        .WORD 0 :MESSAGE LINE A STATUS BYTE 10
        .WORD 0 :MESSAGE LINE B STATUS BYTE 10
        .WORD 0 :MESSAGE LINE A STATUS BYTE 11
        .WORD 0 :MESSAGE LINE B STATUS BYTE 11
        .WORD 0 :ECC POSITION INFORMATION
        .WORD 0 :ECC PATTERN INFORMATION
        .WORD 0 :QUEUE LINK
        .WORD 0 :MINIMUM CYLINDER
        .WORD 410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
        .BYTE 0 :MINIMUM TRACK
        .BYTE 2 :MAXIMUM TRACK
        .BYTE 0 :MINIMUM SECTOR
        .BYTE 21. :MAXIMUM SECTOR
        .BYTE 3 :READ/WRITE RATIO
        .BYTE 0 :AUTOMATIC WRITE CHECK
        .WORD 10. :CORRECTABLE READ ERROR THRESHOLD
        .WORD 5 :UNCORRECTABLE READ ERROR THRESHOLD
        .WORD 5 :SEEK ERROR THRESHOLD
        .WORD 177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
        .WORD 177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
        .BYTE 0 :SAMPLE COMPARE FLAG
        .BYTE 0 :ECC COMPARE FLAG
        .BYTE 0 :INHIBIT ERROR CORRECTION
        .BYTE 0 :DATA PATTERN USED
        .BYTE 0 :CURRENT GENERATED DATA PATTERN
        .BYTE 0 :CURRENT RANDOMLY GENERATED COMMAND
        .WORD 0 :CURRENT RANDOMLY GENERATED CYLINDER
        .BYTE 0 :CURRENT RANDOMLY GENERATED SECTOR
        .BYTE 0 :CURRENT RANDOMLY GENERATED TRACK
        .WORD 0 :CURRENT RANDOMLY GENERATED WORD COUNT

```

2304	003102	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2305	003104	000			.BYTE	0		
2306	003105	000			.BYTE	0		:LAST COMMAND
2307	003106	000000			.WORD	0		:LAST CYLINDER
2308	003110	000			.BYTE	0		:LAST SECTOR
2309	003111	000			.BYTE	0		:LAST TRACK
2310	003112	000000			.WORD	0		:LAST WORD COUNT
2311	003114	000			.BYTE	0		:LAST DATA PATTERN USED
2312	003115	000			.BYTE	0		:BUFFER ALLOCATED
2313	003116	000			.BYTE	0		:RETRY COUNT
2314	003117	000			.BYTE	0		:REREAD STATUS
2315	003120	000000			.WORD	0		:DRIVE STATUS FLAGS
2316	003122	000000	000000		.WORD	0,0		:NUMBER OF ORDERS
2317	003126	000000	000000	000000	.WORD	0,0,0		:NUMBER OF WORDS WRITTEN
2318	003134	000000	000000	000000	.WORD	0,0,0		:NUMBER OF WORDS READ
2319	003142	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2320	003144	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2321	003146	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2322	003150	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2323	003152	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2324	003154	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2325	003156	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2326	003160	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2327	003161	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2328	003162	000000			.WORD	0		:ERROR STATUS INFORMATION
2329	003164	000000			.WORD	0		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2330	003166	000			.BYTE	0		
2331	003167	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2332	003170	000000			.WORD	0		:COMPARISON LENGTH
2333	003172	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2334	003174	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2335	003176	000000	000000		.WORD	0,0		:CARIAGE SERIAL NUMBER
2336	003202	000000	000000		.WORD	0,0		:FIRST EXCLUSION AREA
2337	003206	000000	000000		.WORD	0,0		:SECOND EXCLUSION AREA
2338	003212	000000	000000		.WORD	0,0		:THIRD EXCLUSION AREA
2339	003216	000000	000000		.WORD	0,0		:FOURTH EXCLUSION AREA
2340	003222	000000	000000		.WORD	0,0		:FIFTH EXCLUSION AREA

2341					
2342					
2343					
2344	003226	004			
2345	003227	000			
2346	003230	000000			
2347	003232	000			
2348	003233	000			
2349	003234	000			
2350	003235	000			
2351	003236	000000			
2352	003240	000000			
2353	003242	000000			
2354	003244	000000			
2355	003246	000000			
2356	003250	000000			
2357	003252	000000			
2358	003254	000000			
2359	003256	000000			
2360	003260	000000			
2361	003262	000000			
2362	003264	000000			
2363	003266	000000			
2364	003270	000000			
2365	003272	000000			
2366	003274	000000			
2367	003276	000000			
2368	003300	000000			
2369	003302	000000			
2370	003304	000000			
2371	003306	000000			
2372	003310	000000			
2373	003312	000000			
2374	003314	000000			
2375	003316	000632			
2376	003320	000			
2377	003321	002			
2378	003322	000			
2379	003323	025			
2380	003324	003			
2381	003325	000			
2382	003326	000012			
2383	003330	000005			
2384	003332	000005			
2385	003334	177770	177777		
2386	003340	177770	177777		
2387	003344	000			
2388	003345	000			
2389	003346	000			
2390	003347	000			
2391	003350	000			
2392	003351	000			
2393	003352	000000			
2394	003354	000			
2395	003355	000			
2396	003356	000000			

.SBTTL PARAMETER BLOCK FOR DRIVE 4

PARM4:	.BYTE	4		:DRIVE 4
	.BYTE	0		:COMMAND
	.WORD	0		:CYLINDER ADDRESS
	.BYTE	0		:SECTOR ADDRESS
	.BYTE	0		:TRACK ADDRESS
	.BYTE	0		:OFFSET VALUE
	.BYTE	0		:BUS ADDRESS (BITS 16 AND 17)
	.WORD	0		:BUS ADDRESS (BITS 0 - 15)
	.WORD	0		:WORD COUNT (2'S COMPLEMENT)
	.WORD	0		:PROGRAM DRIVE STATUS INFORMATION
	.WORD	0		:COMMAND AND STATUS REGISTER 1
	.WORD	0		:COMMAND AND STATUS REGISTER 2
	.WORD	0		:WORD COUNT REGISTER
	.WORD	0		:BUS ADDRESS REGISTER
	.WORD	0		:DESIRED TRACK AND SECTOR REGISTER
	.WORD	0		:DESIRED CYLINDER REGISTER
	.WORD	0		:ATTENTION SUMMARY/OFFSET REGISTER
	.WORD	0		:ERROR REGISTER
	.WORD	0		:STATUS REGISTER
	.WORD	0		:MESSAGE LINE A STATUS BYTE 00
	.WORD	0		:MESSAGE LINE B STATUS BYTE 00
	.WORD	0		:MESSAGE LINE A STATUS BYTE 01
	.WORD	0		:MESSAGE LINE B STATUS BYTE 01
	.WORD	0		:MESSAGE LINE A STATUS BYTE 10
	.WORD	0		:MESSAGE LINE B STATUS BYTE 10
	.WORD	0		:MESSAGE LINE A STATUS BYTE 11
	.WORD	0		:MESSAGE LINE B STATUS BYTE 11
	.WORD	0		:ECC POSITION INFORMATION
	.WORD	0		:ECC PATTERN INFORMATION
	.WORD	0		:QUEUE LINK
	.WORD	0		:MINIMUM CYLINDER
	.WORD	410.		:MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
	.BYTE	0		:MINIMUM TRACK
	.BYTE	2		:MAXIMUM TRACK
	.BYTE	0		:MINIMUM SECTOR
	.BYTE	21.		:MAXIMUM SECTOR
	.BYTE	3		:READ/WRITE RATIO
	.BYTE	0		:AUTOMATIC WRITE CHECK
	.WORD	10.		:CORRECTABLE READ ERROR THRESHOLD
	.WORD	5		:UNCORRECTABLE READ ERROR THRESHOLD
	.WORD	5		:SEEK ERROR THRESHOLD
	.WORD	177770,177777		:MAXIMUM NUMBER OF COMMANDS TO DRIVE
	.WORD	177770,177777		:MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
	.BYTE	0		:SAMPLE COMPARE FLAG
	.BYTE	0		:ECC COMPARE FLAG
	.BYTE	0		:INHIBIT ERROR CORRECTION
	.BYTE	0		:DATA PATTERN USED
	.BYTE	0		:CURRENT GENERATED DATA PATTERN
	.BYTE	0		:CURRENT RANDOMLY GENERATED COMMAND
	.WORD	0		:CURRENT RANDOMLY GENERATED CYLINDER
	.BYTE	0		:CURRENT RANDOMLY GENERATED SECTOR
	.BYTE	0		:CURRENT RANDOMLY GENERATED TRACK
	.WORD	0		:CURRENT RANDOMLY GENERATED WORD COUNT

2397	003360	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2398	003362	000			.BYTE	0		
2399	003363	000			.BYTE	0		:LAST COMMAND
2400	003364	000000			.WORD	0		:LAST CYLINDER
2401	003366	000			.BYTE	0		:LAST SECTOR
2402	003367	000			.BYTE	0		:LAST TRACK
2403	003370	000000			.WORD	0		:LAST WORD COUNT
2404	003372	000			.BYTE	0		:LAST DATA PATTERN USED
2405	003373	000			.BYTE	0		:BUFFER ALLOCATED
2406	003374	000			.BYTE	0		:RETRY COUNT
2407	003375	000			.BYTE	0		:REREAD STATUS
2408	003376	000000			.WORD	0		:DRIVE STATUS FLAGS
2409	003400	000000	000000		.WORD	0.0		:NUMBER OF ORDERS
2410	003404	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS WRITTEN
2411	003412	000000	000000	000000	.WORD	0.0.0		:NUMBER OF WORDS READ
2412	003420	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2413	003422	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2414	003424	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2415	003426	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2416	003430	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2417	003432	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2418	003434	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2419	003436	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2420	003437	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2421	003440	000000			.WORD	0		:ERROR STATUS INFORMATION
2422	003442	000000			.WORD	0		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2423	003444	000			.BYTE	0		
2424	003445	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2425	003446	000000			.WORD	0		:COMPARISON LENGTH
2426	003450	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2427	003452	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2428	003454	000000	000000		.WORD	0.0		:CARIAGE SERIAL NUMBER
2429	003460	000000	000000		.WORD	0.0		:FIRST EXCLUSION AREA
2430	003464	000000	000000		.WORD	0.0		:SECOND EXCLUSION AREA
2431	003470	000000	000000		.WORD	0.0		:THIRD EXCLUSION AREA
2432	003474	000000	000000		.WORD	0.0		:FOURTH EXCLUSION AREA
2433	003500	000000	000000		.WORD	0.0		:FIFTH EXCLUSION AREA

```
2434  
2435  
2436  
2437 003504 005  
2438 003505 000  
2439 003506 000000  
2440 003510 000  
2441 003511 000  
2442 003512 000  
2443 003513 000  
2444 003514 000000  
2445 003516 000000  
2446 003520 000000  
2447 003522 000000  
2448 003524 000000  
2449 003526 000000  
2450 003530 000000  
2451 003532 000000  
2452 003534 000000  
2453 003536 000000  
2454 003540 000000  
2455 003542 000000  
2456 003544 000000  
2457 003546 000000  
2458 003550 000000  
2459 003552 000000  
2460 003554 000000  
2461 003556 000000  
2462 003560 000000  
2463 003562 000000  
2464 003564 000000  
2465 003566 000000  
2466 003570 000000  
2467 003572 000000  
2468 003574 000632  
2469 003576 000  
2470 003577 002  
2471 003600 000  
2472 003601 025  
2473 003602 003  
2474 003603 000  
2475 003604 000012  
2476 003606 000005  
2477 003610 000005  
2478 003612 177770 177777  
2479 003616 177770 177777  
2480 003622 000  
2481 003623 000  
2482 003624 000  
2483 003625 000  
2484 003626 000  
2485 003627 000  
2486 003630 000000  
2487 003632 000  
2488 003633 000  
2489 003634 000000
```

.SBTTL PARAMETER BLOCK FOR DRIVE 5

PARM5: .BYTE 5 :DRIVE 5
.BYTE 0 :COMMAND
.WORD 0 :CYLINDER ADDRESS
.BYTE 0 :SECTOR ADDRESS
.BYTE 0 :TRACK ADDRESS
.BYTE 0 :OFFSET VALUE
.BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
.WORD 0 :BUS ADDRESS (BITS 0 - 15)
.WORD 0 :WORD COUNT (2'S COMPLEMENT)
.WORD 0 :PROGRAM DRIVE STATUS INFORMATION
.WORD 0 :COMMAND AND STATUS REGISTER 1
.WORD 0 :COMMAND AND STATUS REGISTER 2
.WORD 0 :WORD COUNT REGISTER
.WORD 0 :BUS ADDRESS REGISTER
.WORD 0 :DESIRED TRACK AND SECTOR REGISTER
.WORD 0 :DESIRED CYLINDER REGISTER
.WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
.WORD 0 :ERROR REGISTER
.WORD 0 :STATUS REGISTER
.WORD 0 :MESSAGE LINE A STATUS BYTE 00
.WORD 0 :MESSAGE LINE B STATUS BYTE 00
.WORD 0 :MESSAGE LINE A STATUS BYTE 01
.WORD 0 :MESSAGE LINE B STATUS BYTE 01
.WORD 0 :MESSAGE LINE A STATUS BYTE 10
.WORD 0 :MESSAGE LINE B STATUS BYTE 10
.WORD 0 :MESSAGE LINE A STATUS BYTE 11
.WORD 0 :MESSAGE LINE B STATUS BYTE 11
.WORD 0 :ECC POSITION INFORMATION
.WORD 0 :ECC PATTERN INFORMATION
.WORD 0 :QUEUE LINK
.WORD 0 :MINIMUM CYLINDER
.WORD 410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
.BYTE 0 :MINIMUM TRACK
.BYTE 2 :MAXIMUM TRACK
.BYTE 0 :MINIMUM SECTOR
.BYTE 21. :MAXIMUM SECTOR
.BYTE 3 :READ/WRITE RATIO
.WORD 0 :AUTOMATIC WRITE CHECK
.WORD 10. :CORRECTABLE READ ERROR THRESHOLD
.WORD 5 :UNCORRECTABLE READ ERROR THRESHOLD
.WORD 5 :SEEK ERROR THRESHOLD
.WORD 177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
.WORD 177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
.BYTE 0 :SAMPLE COMPARE FLAG
.BYTE 0 :ECC COMPARE FLAG
.BYTE 0 :INHIBIT ERROR CORRECTION
.BYTE 0 :DATA PATTERN USED
.BYTE 0 :CURRENT GENERATED DATA PATTERN
.BYTE 0 :CURRENT RANDOMLY GENERATED COMMAND
.WORD 0 :CURRENT RANDOMLY GENERATED CYLINDER
.BYTE 0 :CURRENT RANDOMLY GENERATED SECTOR
.BYTE 0 :CURRENT RANDOMLY GENERATED TRACK
.WORD 0 :CURRENT RANDOMLY GENERATED WORD COUNT

2490	003636	000000			.WORD	0		:CURRENT RANDOMLY GENERATED BUS ADDRESS
2491	003640	000			.BYTE	0		
2492	003641	000			.BYTE	0		:LAST COMMAND
2493	003642	000000			.WORD	0		:LAST CYLINDER
2494	003644	000			.BYTE	0		:LAST SECTOR
2495	003645	000			.BYTE	0		:LAST TRACK
2496	003646	000000			.WORD	0		:LAST WORD COUNT
2497	003650	000			.BYTE	0		:LAST DATA PATTERN USED
2498	003651	000			.BYTE	0		:BUFFER ALLOCATED
2499	003652	000			.BYTE	0		:RETRY COUNT
2500	003653	000			.BYTE	0		:REREAD STATUS
2501	003654	000000			.WORD	0		:DRIVE STATUS FLAGS
2502	003656	000000	000000		.WORD	0,0		:NUMBER OF ORDERS
2503	003662	000000	000000	000000	.WORD	0,0,0		:NUMBER OF WORDS WRITTEN
2504	003670	000000	000000	000000	.WORD	0,0,0		:NUMBER OF WORDS READ
2505	003676	000000			.WORD	0		:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2506	003700	000000			.WORD	0		:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2507	003702	000000			.WORD	0		:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2508	003704	000000			.WORD	0		:NUMBER OF HARD DATA ERRORS
2509	003706	000000			.WORD	0		:NUMBER OF SEEK INCOMPLETES
2510	003710	000000			.WORD	0		:NUMBER OF MISPOSITIONING ERRORS
2511	003712	000000			.WORD	0		:NUMBER OF ALL OTHER ERRORS
2512	003714	000			.BYTE	0		:ASSIGNMENT COMMAND SEQUENCE
2513	003715	000			.BYTE	0		:SEEK TO PREVIOUS COMMAND FLAG
2514	003716	000000			.WORD	0		:ERROR STATUS INFORMATION
2515	003720	000000			.WORD	0		:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2516	003722	000			.BYTE	0		
2517	003723	000			.BYTE	0		:OFFSET ADDRESS FROM HEADER ADDRESS
2518	003724	000000			.WORD	0		:COMPARISON LENGTH
2519	003726	000000			.WORD	0		:BUFFER COMPARISON LENGTH
2520	003730	007777			.WORD	007777		:DRIVE SERIAL NUMBER
2521	003732	000000	000000		.WORD	0,0		:CARIAGE SERIAL NUMBER
2522	003736	000000	000000		.WORD	0,0		:FIRST EXCLUSION AREA
2523	003742	000000	000000		.WORD	0,0		:SECOND EXCLUSION AREA
2524	003746	000000	000000		.WORD	0,0		:THIRD EXCLUSION AREA
2525	003752	000000	000000		.WORD	0,0		:FOURTH EXCLUSION AREA
2526	003756	000000	000000		.WORD	0,0		:FIFTH EXCLUSION AREA

```
2527  
2528  
2529  
2530 003762 006  
2531 003763 000  
2532 003764 000000  
2533 003766 000  
2534 003767 000  
2535 003770 000  
2536 003771 000  
2537 003772 000000  
2538 003774 000000  
2539 003776 000000  
2540 004000 000000  
2541 004002 000000  
2542 004004 000000  
2543 004006 000000  
2544 004010 000000  
2545 004012 000000  
2546 004014 000000  
2547 004016 000000  
2548 004020 000000  
2549 004022 000000  
2550 004024 000000  
2551 004026 000000  
2552 004030 000000  
2553 004032 000000  
2554 004034 000000  
2555 004036 000000  
2556 004040 000000  
2557 004042 000000  
2558 004044 000000  
2559 004046 000000  
2560 004050 000000  
2561 004052 000632  
2562 004054 000  
2563 004055 002  
2564 004056 000  
2565 004057 025  
2566 004060 003  
2567 004061 000  
2568 004062 000012  
2569 004064 000005  
2570 004066 000005  
2571 004070 177770 177777  
2572 004074 177770 177777  
2573 004100 000  
2574 004101 000  
2575 004102 000  
2576 004103 000  
2577 004104 000  
2578 004105 000  
2579 004106 000000  
2580 004110 000  
2581 004111 000  
2582 004112 000000
```

.SBTTL PARAMETER BLOCK FOR DRIVE 6

PARM6: .BYTE 6 :DRIVE 6
.BYTE 0 :COMMAND
.WORD 0 :CYLINDER ADDRESS
.BYTE 0 :SECTOR ADDRESS
.BYTE 0 :TRACK ADDRESS
.BYTE 0 :OFFSET VALUE
.BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
.WORD 0 :BUS ADDRESS (BITS 0 - 15)
.WORD 0 :WORD COUNT (2'S COMPLEMENT)
.WORD 0 :PROGRAM DRIVE STATUS INFORMATION
.WORD 0 :COMMAND AND STATUS REGISTER 1
.WORD 0 :COMMAND AND STATUS REGISTER 2
.WORD 0 :WORD COUNT REGISTER
.WORD 0 :BUS ADDRESS REGISTER
.WORD 0 :DESIRED TRACK AND SECTOR REGISTER
.WORD 0 :DESIRED CYLINDER REGISTER
.WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
.WORD 0 :ERROR REGISTER
.WORD 0 :STATUS REGISTER
.WORD 0 :MESSAGE LINE A STATUS BYTE 00
.WORD 0 :MESSAGE LINE B STATUS BYTE 00
.WORD 0 :MESSAGE LINE A STATUS BYTE 01
.WORD 0 :MESSAGE LINE B STATUS BYTE 01
.WORD 0 :MESSAGE LINE A STATUS BYTE 10
.WORD 0 :MESSAGE LINE B STATUS BYTE 10
.WORD 0 :MESSAGE LINE A STATUS BYTE 11
.WORD 0 :MESSAGE LINE B STATUS BYTE 11
.WORD 0 :ECC POSITION INFORMATION
.WORD 0 :ECC PATTERN INFORMATION
.WORD 0 :QUEUE LINK
.WORD 0 :MINIMUM CYLINDER
.WORD 410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
.BYTE 0 :MINIMUM TRACK
.BYTE 2 :MAXIMUM TRACK
.BYTE 0 :MINIMUM SECTOR
.BYTE 21. :MAXIMUM SECTOR
.BYTE 3 :READ/WRITE RATIO
.BYTE 0 :AUTOMATIC WRITE CHECK
.WORD 10. :CORRECTABLE READ ERROR THRESHOLD
.WORD 5 :UNCORRECTABLE READ ERROR THRESHOLD
.WORD 5 :SEEK ERROR THRESHOLD
.WORD 177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
.WORD 177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
.BYTE 0 :SAMPLE COMPARE FLAG
.BYTE 0 :ECC COMPARE FLAG
.BYTE 0 :INHIBIT ERROR CORRECTION
.BYTE 0 :DATA PATTERN USED
.BYTE 0 :CURRENT GENERATED DATA PATTERN
.BYTE 0 :CURRENT RANDOMLY GENERATED COMMAND
.WORD 0 :CURRENT RANDOMLY GENERATED CYLINDER
.BYTE 0 :CURRENT RANDOMLY GENERATED SECTOR
.BYTE 0 :CURRENT RANDOMLY GENERATED TRACK
.WORD 0 :CURRENT RANDOMLY GENERATED WORD COUNT

Address	Offset	Value	Unit	Description
2583	004114	000000	.WORD	0 :CURRENT RANDOMLY GENERATED BUS ADDRESS
2584	004116	000	.BYTE	0
2585	004117	000	.BYTE	0 :LAST COMMAND
2586	004120	000000	.WORD	0 :LAST CYLINDER
2587	004122	000	.BYTE	0 :LAST SECTOR
2588	004123	000	.BYTE	0 :LAST TRACK
2589	004124	000000	.WORD	0 :LAST WORD COUNT
2590	004126	000	.BYTE	0 :LAST DATA PATTERN USED
2591	004127	000	.BYTE	0 :BUFFER ALLOCATED
2592	004130	000	.BYTE	0 :RETRY COUNT
2593	004131	000	.BYTE	0 :REREAD STATUS
2594	004132	000000	.WORD	0 :DRIVE STATUS FLAGS
2595	004134	000000	.WORD	0.0 :NUMBER OF ORDERS
2596	004140	000000	.WORD	0.0.0 :NUMBER OF WORDS WRITTEN
2597	004146	000000	.WORD	0.0.0 :NUMBER OF WORDS READ
2598	004154	000000	.WORD	0 :NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2599	004156	000000	.WORD	0 :NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2600	004160	000000	.WORD	0 :NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2601	004162	000000	.WORD	0 :NUMBER OF HARD DATA ERRORS
2602	004164	000000	.WORD	0 :NUMBER OF SEEK INCOMPLETES
2603	004166	000000	.WORD	0 :NUMBER OF MISPOSITIONING ERRORS
2604	004170	000000	.WORD	0 :NUMBER OF ALL OTHER ERRORS
2605	004172	000	.BYTE	0 :ASSIGNMENT COMMAND SEQUENCE
2606	004173	000	.BYTE	0 :SEEK TO PREVIOUS COMMAND FLAG
2607	004174	000000	.WORD	0 :ERROR STATUS INFORMATION
2608	004176	000000	.WORD	0 :HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2609	004200	000	.BYTE	0
2610	004201	000	.BYTE	0 :OFFSET ADDRESS FROM HEADER ADDRESS
2611	004202	000000	.WORD	0 :COMPARISON LENGTH
2612	004204	000000	.WORD	0 :BUFFER COMPARISON LENGTH
2613	004206	007777	.WORD	007777 :DRIVE SERIAL NUMBER
2614	004210	000000	.WORD	0.0 :CARIAGE SERIAL NUMBER
2615	004214	000000	.WORD	0.0 :FIRST EXCLUSION AREA
2616	004220	000000	.WORD	0.0 :SECOND EXCLUSION AREA
2617	004224	000000	.WORD	0.0 :THIRD EXCLUSION AREA
2618	004230	000000	.WORD	0.0 :FOURTH EXCLUSION AREA
2619	004234	000000	.WORD	0.0 :FIFTH EXCLUSION AREA

```
2620  
2621  
2622  
2623 004240 007  
2624 004241 000  
2625 004242 000000  
2626 004244 000  
2627 004245 000  
2628 004246 000  
2629 004247 000  
2630 004250 000000  
2631 004252 000000  
2632 004254 000000  
2633 004256 000000  
2634 004260 000000  
2635 004262 000000  
2636 004264 000000  
2637 004266 000000  
2638 004270 000000  
2639 004272 000000  
2640 004274 000000  
2641 004276 000000  
2642 004300 000000  
2643 004302 000000  
2644 004304 000000  
2645 004306 000000  
2646 004310 000000  
2647 004312 000000  
2648 004314 000000  
2649 004316 000000  
2650 004320 000000  
2651 004322 000000  
2652 004324 000000  
2653 004326 000000  
2654 004330 000632  
2655 004332 000  
2656 004333 002  
2657 004334 000  
2658 004335 025  
2659 004336 003  
2660 004337 000  
2661 004340 000012  
2662 004342 000005  
2663 004344 000005  
2664 004346 177770 177777  
2665 004352 177770 177777  
2666 004356 000  
2667 004357 000  
2668 004360 000  
2669 004361 000  
2670 004362 000  
2671 004363 000  
2672 004364 000000  
2673 004366 000  
2674 004367 000  
2675 004370 000000
```

.SBTTL PARAMETER BLOCK FOR DRIVE 7

PARM7: .BYTE 7 :DRIVE 7
.BYTE 0 :COMMAND
.WORD 0 :CYLINDER ADDRESS
.BYTE 0 :SECTOR ADDRESS
.BYTE 0 :TRACK ADDRESS
.BYTE 0 :OFFSET VALUE
.BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
.WORD 0 :BUS ADDRESS (BITS 0 - 15)
.WORD 0 :WORD COUNT (2'S COMPLEMENT)
.WORD 0 :PROGRAM DRIVE STATUS INFORMATION
.WORD 0 :COMMAND AND STATUS REGISTER 1
.WORD 0 :COMMAND AND STATUS REGISTER 2
.WORD 0 :WORD COUNT REGISTER
.WORD 0 :BUS ADDRESS REGISTER
.WORD 0 :DESIRED TRACK AND SECTOR REGISTER
.WORD 0 :DESIRED CYLINDER REGISTER
.WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
.WORD 0 :ERROR REGISTER
.WORD 0 :STATUS REGISTER
.WORD 0 :MESSAGE LINE A STATUS BYTE 00
.WORD 0 :MESSAGE LINE B STATUS BYTE 00
.WORD 0 :MESSAGE LINE A STATUS BYTE 01
.WORD 0 :MESSAGE LINE B STATUS BYTE 01
.WORD 0 :MESSAGE LINE A STATUS BYTE 10
.WORD 0 :MESSAGE LINE B STATUS BYTE 10
.WORD 0 :MESSAGE LINE A STATUS BYTE 11
.WORD 0 :MESSAGE LINE B STATUS BYTE 11
.WORD 0 :ECC POSITION INFORMATION
.WORD 0 :ECC PATTERN INFORMATION
.WORD 0 :QUEUE LINK
.WORD 0 :MINIMUM CYLINDER
410. :MAXIMUM CYLINDER (DEFAULT TO RK06 MAX CYL)
.BYTE 0 :MINIMUM TRACK
2 :MAXIMUM TRACK
.BYTE 0 :MINIMUM SECTOR
21. :MAXIMUM SECTOR
3 :READ/WRITE RATIO
.BYTE 0 :AUTOMATIC WRITE CHECK
10. :CORRECTABLE READ ERROR THRESHOLD
5 :UNCORRECTABLE READ ERROR THRESHOLD
5 :SEEK ERROR THRESHOLD
177770,177777 :MAXIMUM NUMBER OF COMMANDS TO DRIVE
177770,177777 :MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
0 :SAMPLE COMPARE FLAG
0 :ECC COMPARE FLAG
0 :INHIBIT ERROR CORRECTION
0 :DATA PATTERN USED
0 :CURRENT GENERATED DATA PATTERN
0 :CURRENT RANDOMLY GENERATED COMMAND
0 :CURRENT RANDOMLY GENERATED CYLINDER
0 :CURRENT RANDOMLY GENERATED SECTOR
0 :CURRENT RANDOMLY GENERATED TRACK
0 :CURRENT RANDOMLY GENERATED WORD COUNT

2676	004372	000000			.WORD	0	:CURRENT RANDOMLY GENERATED BUS ADDRESS
2677	004374	000			.BYTE	0	
2678	004375	000			.BYTE	0	:LAST COMMAND
2679	004376	000000			.WORD	0	:LAST CYLINDER
2680	004400	000			.BYTE	0	:LAST SECTOR
2681	004401	000			.BYTE	0	:LAST TRACK
2682	004402	000000			.WORD	0	:LAST WORD COUNT
2683	004404	000			.BYTE	0	:LAST DATA PATTERN USED
2684	004405	000			.BYTE	0	:BUFFER ALLOCATED
2685	004406	000			.BYTE	0	:RETRY COUNT
2686	004407	000			.BYTE	0	:REREAD STATUS
2687	004410	000000			.WORD	0	:DRIVE STATUS FLAGS
2688	004412	000000	000000		.WORD	0,0	:NUMBER OF ORDERS
2689	004416	000000	000000	000000	.WORD	0,0,0	:NUMBER OF WORDS WRITTEN
2690	004424	000000	000000	000000	.WORD	0,0,0	:NUMBER OF WORDS READ
2691	004432	000000			.WORD	0	:NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
2692	004434	000000			.WORD	0	:NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
2693	004436	000000			.WORD	0	:NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
2694	004440	000000			.WORD	0	:NUMBER OF HARD DATA ERRORS
2695	004442	000000			.WORD	0	:NUMBER OF SEEK INCOMPLETES
2696	004444	000000			.WORD	0	:NUMBER OF MISPOSITIONING ERRORS
2697	004446	000000			.WORD	0	:NUMBER OF ALL OTHER ERRORS
2698	004450	000			.BYTE	0	:ASSIGNMENT COMMAND SEQUENCE
2699	004451	000			.BYTE	0	:SEEK TO PREVIOUS COMMAND FLAG
2700	004452	000000			.WORD	0	:ERROR STATUS INFORMATION
2701	004454	000000			.WORD	0	:HEADER ADDRESS OF FIRST MISCOMPARE ERROR
2702	004456	000			.BYTE	0	
2703	004457	000			.BYTE	0	:OFFSET ADDRESS FROM HEADER ADDRESS
2704	004460	000000			.WORD	0	:COMPARISON LENGTH
2705	004462	000000			.WORD	0	:BUFFER COMPARISON LENGTH
2706	004464	007777			.WORD	007777	:DRIVE SERIAL NUMBER
2707	004466	000000	000000		.WORD	0,0	:CARIAGE SERIAL NUMBER
2708	004472	000000	000000		.WORD	0,0	:FIRST EXCLUSION AREA
2709	004476	000000	000000		.WORD	0,0	:SECOND EXCLUSION AREA
2710	004502	000000	000000		.WORD	0,0	:THIRD EXCLUSION AREA
2711	004506	000000	000000		.WORD	0,0	:FOURTH EXCLUSION AREA
2712	004512	000000	000000		.WORD	0,0	:FIFTH EXCLUSION AREA

```
2713 .SBTTL --- RETRY PARAMETER BLOCK ---
2714
2715 .SBTTL PARAMETER BLOCK 10 FOR DRIVE
2716
2717 004516 000 PARM10: .BYTE 0 :DRIVE NUMBER
2718 004517 000 .BYTE 0 :COMMAND
2719 004520 000000 .WORD 0 :CYLINDER ADDRESS
2720 004522 000 .BYTE 0 :SECTOR ADDRESS
2721 004523 000 .BYTE 0 :TRACK ADDRESS
2722 004524 000 .BYTE 0 :OFFSET VALUE
2723 004525 000 .BYTE 0 :BUS ADDRESS (BITS 16 AND 17)
2724 004526 000000 .WORD 0 :BUS ADDRESS (BITS 0 - 15)
2725 004530 000000 .WORD 0 :WORD COUNT (2'S COMPLEMENT)
2726 004532 000000 .WORD 0 :PROGRAM DRIVE STATUS INFORMATION
2727 004534 000000 .WORD 0 :COMMAND AND STATUS REGISTER 1
2728 004536 000000 .WORD 0 :COMMAND AND STATUS REGISTER 2
2729 004540 000000 .WORD 0 :WORD COUNT REGISTER
2730 004542 000000 .WORD 0 :BUS ADDRESS REGISTER
2731 004544 000000 .WORD 0 :DESIRED TRACK AND SECTOR REGISTER
2732 004546 000000 .WORD 0 :DESIRED CYLINDER REGISTER
2733 004550 000000 .WORD 0 :ATTENTION SUMMARY/OFFSET REGISTER
2734 004552 000000 .WORD 0 :ERROR REGISTER
2735 004554 000000 .WORD 0 :STATUS REGISTER
2736 004556 000000 .WORD 0 :MESSAGE LINE A STATUS BYTE 00
2737 004560 000000 .WORD 0 :MESSAGE LINE B STATUS BYTE 00
2738 004562 000000 .WORD 0 :MESSAGE LINE A STATUS BYTE 01
2739 004564 000000 .WORD 0 :MESSAGE LINE B STATUS BYTE 01
2740 004566 000000 .WORD 0 :MESSAGE LINE A STATUS BYTE 10
2741 004570 000000 .WORD 0 :MESSAGE LINE B STATUS BYTE 10
2742 004572 000000 .WORD 0 :MESSAGE LINE A STATUS BYTE 11
2743 004574 000000 .WORD 0 :MESSAGE LINE B STATUS BYTE 11
2744 004576 000000 .WORD 0 :ECC POSITION INFORMATION
2745 004600 000000 .WORD 0 :ECC PATTERN INFORMATION
```


2802	004752	151322	.WORD	151322
2803	004754	151322	.WORD	151322
2804	004756	026455	.WORD	026455
2805	004760	026455	.WORD	026455
2806	004762	026455	.WORD	026455
2807	004764	151322	.WORD	151322
2808	004766	151322	.WORD	151322
2809	004770	151322	.WORD	151322
2810	004772	026455	.WORD	026455
2811	004774	026455	.WORD	026455
2812	004776	026455	.WORD	026455
2813	005000	026455	.WORD	026455
2814	005002	151322	.WORD	151322
2815	005004	151322	.WORD	151322
2816	005006	151322	.WORD	151322
2817	005010	151322	.WORD	151322
2818	005012	026455	.WORD	026455
2819	005014	026455	.WORD	026455
2820	005016	026455	.WORD	026455
2821	005020	151322	.WORD	151322
2822	005022	151322	.WORD	151322
2823	005024	151322	.WORD	151322
2824	005026	026455	.WORD	026455
2825	005030	026455	.WORD	026455
2826	005032	151322	.WORD	151322
2827	005034	151322	.WORD	151322
2828	005036	026455	.WORD	026455
2829	005040	151322	.WORD	151322
2830				
2831	005042	007417	PAT2: .WORD	007417
2832	005044	170360	.WORD	170360
2833	005046	007417	.WORD	007417
2834	005050	007417	.WORD	007417
2835	005052	170360	.WORD	170360
2836	005054	170360	.WORD	170360
2837	005056	007417	.WORD	007417
2838	005060	007417	.WORD	007417
2839	005062	007417	.WORD	007417
2840	005064	170360	.WORD	170360
2841	005066	170360	.WORD	170360
2842	005070	170360	.WORD	170360
2843	005072	007417	.WORD	007417
2844	005074	007417	.WORD	007417
2845	005076	007417	.WORD	007417
2846	005100	007417	.WORD	007417
2847	005102	170360	.WORD	170360
2848	005104	170360	.WORD	170360
2849	005106	170360	.WORD	170360
2850	005110	170360	.WORD	170360
2851	005112	007417	.WORD	007417
2852	005114	007417	.WORD	007417
2853	005116	007417	.WORD	007417
2854	005120	170360	.WORD	170360
2855	005122	170360	.WORD	170360
2856	005124	170360	.WORD	170360
2857	005126	007417	.WORD	007417

2858	005130	007417	.WORD	007417
2859	005132	170360	.WORD	170360
2860	005134	170360	.WORD	170360
2861	005136	007417	.WORD	007417
2862	005140	170360	.WORD	170360
2863				
2864	005142	052525	PAT3: .WORD	052525
2865	005144	125252	.WORD	125252
2866	005146	052525	.WORD	052525
2867	005150	052525	.WORD	052525
2868	005152	125252	.WORD	125252
2869	005154	125252	.WORD	125252
2870	005156	052525	.WORD	052525
2871	005160	052525	.WORD	052525
2872	005162	052525	.WORD	052525
2873	005164	125252	.WORD	125252
2874	005166	125252	.WORD	125252
2875	005170	125252	.WORD	125252
2876	005172	052525	.WORD	052525
2877	005174	052525	.WORD	052525
2878	005176	052525	.WORD	052525
2879	005200	052525	.WORD	052525
2880	005202	125252	.WORD	125252
2881	005204	125252	.WORD	125252
2882	005206	125252	.WORD	125252
2883	005210	125252	.WORD	125252
2884	005212	052525	.WORD	052525
2885	005214	052525	.WORD	052525
2886	005216	052525	.WORD	052525
2887	005220	125252	.WORD	125252
2888	005222	125252	.WORD	125252
2889	005224	125252	.WORD	125252
2890	005226	052525	.WORD	052525
2891	005230	052525	.WORD	052525
2892	005232	125252	.WORD	125252
2893	005234	125252	.WORD	125252
2894	005236	052525	.WORD	052525
2895	005240	125252	.WORD	125252
2896				
2897	005242	000000	PAT4: .WORD	000000
2898	005244	010421	.WORD	010421
2899	005246	021042	.WORD	021042
2900	005250	031463	.WORD	031463
2901	005252	042104	.WORD	042104
2902	005254	052525	.WORD	052525
2903	005256	063146	.WORD	063146
2904	005260	073567	.WORD	073567
2905	005262	104210	.WORD	104210
2906	005264	114631	.WORD	114631
2907	005266	125252	.WORD	125252
2908	005270	135673	.WORD	135673
2909	005272	146314	.WORD	146314
2910	005274	156735	.WORD	156735
2911	005276	167356	.WORD	167356
2912	005300	177777	.WORD	177777
2913	005302	177777	.WORD	177777

2914	005304	167356	.WORD	167356
2915	005306	156735	.WORD	156735
2916	005310	146314	.WORD	146314
2917	005312	135673	.WORD	135673
2918	005314	125252	.WORD	125252
2919	005316	114631	.WORD	114631
2920	005320	104210	.WORD	104210
2921	005322	073567	.WORD	073567
2922	005324	063146	.WORD	063146
2923	005326	052525	.WORD	052525
2924	005330	042104	.WORD	042104
2925	005332	031463	.WORD	031463
2926	005334	021042	.WORD	021042
2927	005336	010421	.WORD	010421
2928	005340	000000	.WORD	000000
2929				
2930	005342	000000	PAT5: .WORD	000000
2931	005344	177777	.WORD	177777
2932	005346	000000	.WORD	000000
2933	005350	000000	.WORD	000000
2934	005352	177777	.WORD	177777
2935	005354	177777	.WORD	177777
2936	005356	000000	.WORD	000000
2937	005360	000000	.WORD	000000
2938	005362	000000	.WORD	000000
2939	005364	177777	.WORD	177777
2940	005366	177777	.WORD	177777
2941	005370	177777	.WORD	177777
2942	005372	000000	.WORD	000000
2943	005374	000000	.WORD	000000
2944	005376	000000	.WORD	000000
2945	005400	000000	.WORD	000000
2946	005402	177777	.WORD	177777
2947	005404	177777	.WORD	177777
2948	005406	177777	.WORD	177777
2949	005410	177777	.WORD	177777
2950	005412	000000	.WORD	000000
2951	005414	000000	.WORD	000000
2952	005416	000000	.WORD	000000
2953	005420	177777	.WORD	177777
2954	005422	177777	.WORD	177777
2955	005424	177777	.WORD	177777
2956	005426	000000	.WORD	000000
2957	005430	000000	.WORD	000000
2958	005432	177777	.WORD	177777
2959	005434	177777	.WORD	177777
2960	005436	000000	.WORD	000000
2961	005440	177777	.WORD	177777
2962				
2963	005442	000001	PAT6: .WORD	000001
2964	005444	000003	.WORD	000003
2965	005446	000007	.WORD	000007
2966	005450	000017	.WORD	000017
2967	005452	000037	.WORD	000037
2968	005454	000077	.WORD	000077
2969	005456	000177	.WORD	000177

2970	005460	000377	.WORD	000377
2971	005462	000777	.WORD	000777
2972	005464	001777	.WORD	001777
2973	005466	003777	.WORD	003777
2974	005470	007777	.WORD	007777
2975	005472	017777	.WORD	017777
2976	005474	037777	.WORD	037777
2977	005476	077777	.WORD	077777
2978	005500	177777	.WORD	177777
2979	005502	177776	.WORD	177776
2980	005504	177774	.WORD	177774
2981	005506	177770	.WORD	177770
2982	005510	177760	.WORD	177760
2983	005512	177740	.WORD	177740
2984	005514	177700	.WORD	177700
2985	005516	177600	.WORD	177600
2986	005520	177400	.WORD	177400
2987	005522	177000	.WORD	177000
2988	005524	176000	.WORD	176000
2989	005526	174000	.WORD	174000
2990	005530	170000	.WORD	170000
2991	005532	160000	.WORD	160000
2992	005534	140000	.WORD	140000
2993	005536	100000	.WORD	100000
2994	005540	000000	.WORD	000000
2995				
2996	005542	000001	PAT7: .WORD	000001
2997	005544	000002	.WORD	000002
2998	005546	000004	.WORD	000004
2999	005550	000010	.WORD	000010
3000	005552	000020	.WORD	000020
3001	005554	000040	.WORD	000040
3002	005556	000100	.WORD	000100
3003	005560	000200	.WORD	000200
3004	005562	000400	.WORD	000400
3005	005564	001000	.WORD	001000
3006	005566	002000	.WORD	002000
3007	005570	004000	.WORD	004000
3008	005572	010000	.WORD	010000
3009	005574	020000	.WORD	020000
3010	005576	040000	.WORD	040000
3011	005600	100000	.WORD	100000
3012	005602	100000	.WORD	100000
3013	005604	040000	.WORD	040000
3014	005606	020000	.WORD	020000
3015	005610	010000	.WORD	010000
3016	005612	004000	.WORD	004000
3017	005614	002000	.WORD	002000
3018	005616	001000	.WORD	001000
3019	005620	000400	.WORD	000400
3020	005622	000200	.WORD	000200
3021	005624	000100	.WORD	000100
3022	005626	000040	.WORD	000040
3023	005630	000020	.WORD	000020
3024	005632	000010	.WORD	000010
3025	005634	000004	.WORD	000004

3026	005636	000002	.WORD	000002
3027	005640	000001	.WORD	000001
3028				
3029	005642	177776	PAT10: .WORD	177776
3030	005644	177775	.WORD	177775
3031	005646	177773	.WORD	177773
3032	005650	177767	.WORD	177767
3033	005652	177757	.WORD	177757
3034	005654	177737	.WORD	177737
3035	005656	177677	.WORD	177677
3036	005660	177577	.WORD	177577
3037	005662	177377	.WORD	177377
3038	005664	176777	.WORD	176777
3039	005666	175777	.WORD	175777
3040	005670	173777	.WORD	173777
3041	005672	167777	.WORD	167777
3042	005674	157777	.WORD	157777
3043	005676	137777	.WORD	137777
3044	005700	077777	.WORD	077777
3045	005702	077777	.WORD	077777
3046	005704	137777	.WORD	137777
3047	005706	157777	.WORD	157777
3048	005710	167777	.WORD	167777
3049	005712	173777	.WORD	173777
3050	005714	175777	.WORD	175777
3051	005716	176777	.WORD	176777
3052	005720	177377	.WORD	177377
3053	005722	177577	.WORD	177577
3054	005724	177677	.WORD	177677
3055	005726	177737	.WORD	177737
3056	005730	177757	.WORD	177757
3057	005732	177767	.WORD	177767
3058	005734	177773	.WORD	177773
3059	005736	177775	.WORD	177775
3060	005740	177776	.WORD	177776
3061				
3062	005742	172666	PAT11: .WORD	172666
3063	005744	155555	.WORD	155555
3064	005746	172666	.WORD	172666
3065	005750	155555	.WORD	155555
3066	005752	172666	.WORD	172666
3067	005754	155555	.WORD	155555
3068	005756	172666	.WORD	172666
3069	005760	155555	.WORD	155555
3070	005762	172666	.WORD	172666
3071	005764	155555	.WORD	155555
3072	005766	172666	.WORD	172666
3073	005770	155555	.WORD	155555
3074	005772	172666	.WORD	172666
3075	005774	155555	.WORD	155555
3076	005776	172666	.WORD	172666
3077	006000	155555	.WORD	155555
3078	006002	172666	.WORD	172666
3079	006004	155555	.WORD	155555
3080	006006	172666	.WORD	172666
3081	006010	155555	.WORD	155555

3082	006012	172666	.WORD	172666
3083	006014	155555	.WORD	155555
3084	006016	172666	.WORD	172666
3085	006020	155555	.WORD	155555
3086	006022	172666	.WORD	172666
3087	006024	155555	.WORD	155555
3088	006026	172666	.WORD	172666
3089	006030	155555	.WORD	155555
3090	006032	172666	.WORD	172666
3091	006034	155555	.WORD	155555
3092	006036	172666	.WORD	172666
3093	006040	155555	.WORD	155555
3094				
3095	006042	153333	PAT12: .WORD	153333
3096	006044	066667	.WORD	066667
3097	006046	153333	.WORD	153333
3098	006050	066667	.WORD	066667
3099	006052	153333	.WORD	153333
3100	006054	066667	.WORD	066667
3101	006056	153333	.WORD	153333
3102	006060	066667	.WORD	066667
3103	006062	153333	.WORD	153333
3104	006064	066667	.WORD	066667
3105	006066	153333	.WORD	153333
3106	006070	066667	.WORD	066667
3107	006072	153333	.WORD	153333
3108	006074	066667	.WORD	066667
3109	006076	153333	.WORD	153333
3110	006100	066667	.WORD	066667
3111	006102	153333	.WORD	153333
3112	006104	066667	.WORD	066667
3113	006106	153333	.WORD	153333
3114	006110	066667	.WORD	066667
3115	006112	153333	.WORD	153333
3116	006114	066667	.WORD	066667
3117	006116	153333	.WORD	153333
3118	006120	066667	.WORD	066667
3119	006122	153333	.WORD	153333
3120	006124	066667	.WORD	066667
3121	006126	153333	.WORD	153333
3122	006130	066667	.WORD	066667
3123	006132	153333	.WORD	153333
3124	006134	066667	.WORD	066667
3125	006136	153333	.WORD	153333
3126	006140	066667	.WORD	066667
3127				
3128	006142	000000	PAT13: .WORD	000000
3129	006144	177777	.WORD	177777
3130	006146	177777	.WORD	177777
3131	006150	177777	.WORD	177777
3132	006152	177777	.WORD	177777
3133	006154	177777	.WORD	177777
3134	006156	177777	.WORD	177777
3135	006160	177777	.WORD	177777
3136	006162	177777	.WORD	177777
3137	006164	177777	.WORD	177777

3138	006166	177777	.WORD	177777
3139	006170	177777	.WORD	177777
3140	006172	177777	.WORD	177777
3141	006174	177777	.WORD	177777
3142	006176	177777	.WORD	177777
3143	006200	177777	.WORD	177777
3144	006202	177777	.WORD	177777
3145	006204	177777	.WORD	177777
3146	006206	177777	.WORD	177777
3147	006210	177777	.WORD	177777
3148	006212	177777	.WORD	177777
3149	006214	177777	.WORD	177777
3150	006216	177777	.WORD	177777
3151	006220	177777	.WORD	177777
3152	006222	177777	.WORD	177777
3153	006224	177777	.WORD	177777
3154	006226	177777	.WORD	177777
3155	006230	177777	.WORD	177777
3156	006232	177777	.WORD	177777
3157	006234	177777	.WORD	177777
3158	006236	177777	.WORD	177777
3159	006240	177777	.WORD	177777
3160				
3161	006242	177777		
3162	006244	000000	PAT14: .WORD	177777
3163	006246	000000	.WORD	000000
3164	006250	000000	.WORD	000000
3165	006252	000000	.WORD	000000
3166	006254	000000	.WORD	000000
3167	006256	000000	.WORD	000000
3168	006260	000000	.WORD	000000
3169	006262	000000	.WORD	000000
3170	006264	000000	.WORD	000000
3171	006266	000000	.WORD	000000
3172	006270	000000	.WORD	000000
3173	006272	000000	.WORD	000000
3174	006274	000000	.WORD	000000
3175	006276	000000	.WORD	000000
3176	006300	000000	.WORD	000000
3177	006302	000000	.WORD	000000
3178	006304	000000	.WORD	000000
3179	006306	000000	.WORD	000000
3180	006310	000000	.WORD	000000
3181	006312	000000	.WORD	000000
3182	006314	000000	.WORD	000000
3183	006316	000000	.WORD	000000
3184	006320	000000	.WORD	000000
3185	006322	000000	.WORD	000000
3186	006324	000000	.WORD	000000
3187	006326	000000	.WORD	000000
3188	006330	000000	.WORD	000000
3189	006332	000000	.WORD	000000
3190	006334	000000	.WORD	000000
3191	006336	000000	.WORD	000000
3192	006340	000000	.WORD	000000
3193				

3194	006342	172304	PAT15:	.WORD	172304
3195	006344	172304		.WORD	172304
3196	006346	172304		.WORD	172304
3197	006350	172304		.WORD	172304
3198	006352	172304		.WORD	172304
3199	006354	172304		.WORD	172304
3200	006356	172304		.WORD	172304
3201	006360	172304		.WORD	172304
3202	006362	172304		.WORD	172304
3203	006364	172304		.WORD	172304
3204	006366	172304		.WORD	172304
3205	006370	172304		.WORD	172304
3206	006372	172304		.WORD	172304
3207	006374	172304		.WORD	172304
3208	006376	172304		.WORD	172304
3209	006400	172304		.WORD	172304
3210	006402	172304		.WORD	172304
3211	006404	172304		.WORD	172304
3212	006406	172304		.WORD	172304
3213	006410	172304		.WORD	172304
3214	006412	172304		.WORD	172304
3215	006414	172304		.WORD	172304
3216	006416	172304		.WORD	172304
3217	006420	172304		.WORD	172304
3218	006422	172304		.WORD	172304
3219	006424	172304		.WORD	172304
3220	006426	172304		.WORD	172304
3221	006430	172304		.WORD	172304
3222	006432	172304		.WORD	172304
3223	006434	172304		.WORD	172304
3224	006436	172304		.WORD	172304
3225	006440	172304		.WORD	172304
3226					
3227	006442	070627	PAT16:	.WORD	070627
3228	006444	113431		.WORD	113431
3229	006446	014561		.WORD	014561
3230	006450	070627		.WORD	070627
3231	006452	113431		.WORD	113431
3232	006454	014561		.WORD	014561
3233	006456	070627		.WORD	070627
3234	006460	113431		.WORD	113431
3235	006462	014561		.WORD	014561
3236	006464	070627		.WORD	070627
3237	006466	113431		.WORD	113431
3238	006470	014561		.WORD	014561
3239	006472	070627		.WORD	070627
3240	006474	113431		.WORD	113431
3241	006476	014561		.WORD	014561
3242	006500	070627		.WORD	070627
3243	006502	113431		.WORD	113431
3244	006504	014561		.WORD	014561
3245	006506	070627		.WORD	070627
3246	006510	113431		.WORD	113431
3247	006512	014561		.WORD	014561
3248	006514	070627		.WORD	070627
3249	006516	113431		.WORD	113431

3250	006520	014561	.WORD	014561
3251	006522	070627	.WORD	070627
3252	006524	113431	.WORD	113431
3253	006526	014561	.WORD	014561
3254	006530	070627	.WORD	070627
3255	006532	113431	.WORD	113431
3256	006534	014561	.WORD	014561
3257	006536	070627	.WORD	070627
3258	006540	113431	.WORD	113431
3259				
3260	006542	133467	PAT17: .WORD	133467

3261	006544	133467	.WORD	133467
3262	006546	133467	.WORD	133467
3263	006550	133467	.WORD	133467
3264	006552	133467	.WORD	133467
3265	006554	133467	.WORD	133467
3266	006556	133467	.WORD	133467
3267	006560	133467	.WORD	133467
3268	006562	133467	.WORD	133467
3269	006564	133467	.WORD	133467
3270	006566	133467	.WORD	133467
3271	006570	133467	.WORD	133467
3272	006572	133467	.WORD	133467
3273	006574	133467	.WORD	133467
3274	006576	133467	.WORD	133467
3275	006600	133467	.WORD	133467
3276	006602	133467	.WORD	133467
3277	006604	133467	.WORD	133467
3278	006606	133467	.WORD	133467
3279	006610	133467	.WORD	133467
3280	006612	133467	.WORD	133467
3281	006614	133467	.WORD	133467
3282	006616	133467	.WORD	133467
3283	006620	133467	.WORD	133467
3284	006622	133467	.WORD	133467
3285	006624	133467	.WORD	133467
3286	006626	133467	.WORD	133467
3287	006630	133467	.WORD	133467
3288	006632	133467	.WORD	133467
3289	006634	133467	.WORD	133467
3290	006636	133467	.WORD	133467
3291	006640	133467	.WORD	133467

```

3292      .SBTTL  PROGRAM SETUP
3293
3294 006642 112737 000001 001361 PARSRT: MOVB  #1,FLAG      ;SET FLAG FOR PARAMETER ENTRY
3295 006650 000412                BR          PRGSRT      ;START PROGRAM
3296
3297 006652 112737 177777 001361 RESTRT: MOVB  #-1,FLAG     ;SET FLAG FOR RETRY
3298 006660 000406                BR          PRGSRT      ;START PROGRAM
3299
3300 006662 105037 001361          START1: CLRB  FLAG        ;SET FOR ENTRY INPUT START
3301 006666 000403                BR          PRGSRT      ;START PROGRAM
3302
3303 006670 112737 000002 001361 START:  MOVB  #2,FLAG     ;SET FOR AUTO SIZE PGM SIZE
3304
3305 006676 012737 000340 177776 PRGSRT: MOV   #PR7,PS      ;LOCK OUT ALL INTERRUPTS
3306      .SBTTL  INITIALIZE THE COMMON TAGS
3307      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3308 006704 012706 001100          MOV   #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3309 006710 005026                CLR   (R6)+          ;;CLEAR MEMORY LOCATION
3310 006712 022706 001140          CMP   #SWR,R6      ;;DONE?
3311 006716 001374                BNE   -6             ;;LOOP BACK IF NO
3312 006720 012706 001100          MOV   #STACK,SP     ;;SETUP THE STACK POINTER
3313      ;;INITIALIZE A FEW VECTORS
3314 006724 012737 056160 000034  MOV   #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3315 006732 012737 000340 000036  MOV   #340,@#TRAPVEC+2;LEVEL 7
3316 006740 012737 055046 000024  MOV   #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3317 006746 012737 000340 000026  MOV   #340,@#PWRVEC+2 ;LEVEL 7
3318      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3319      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
3320 006754 013746 000004          MOV   @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
3321 006760 012737 007014 000004  MOV   #64,@#ERRVEC   ;;SET UP ERROR VECTOR
3322 006766 012737 177570 001140  MOV   #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
3323 006774 012737 177570 001142  MOV   #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
3324 007002 022777 177777 172130  CMP   #-1,@SWR      ;;TRY TO REFERENCE HARDWARE SWR
3325 007010 001012                BNE   66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3326      ;;AND THE HARDWARE SWR IS NOT = -1
3327 007012 000403                BR    65$           ;;BRANCH IF NO TIMEOUT
3328 007014 012716 007022          64$:  MOV   #65$,(SP)   ;;SET UP FOR TRAP RETURN
3329 007020 000002                RTI
3330 007022 012737 000176 001140  65$:  MOV   #SWREG,SWR    ;;POINT TO SOFTWARE SWR
3331 007030 012737 000174 001142  MOV   #DISPREG,DISPLAY
3332 007036 012637 000004          66$:  MOV   (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
3333
3334 007042 005037 001176          CLR   $PASS         ;;CLEAR PASS COUNT
3335 007046 132737 000200 001211  BITB  #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
3336 007054 001403                BEQ   67$           ;;YES,USE NON-APT SWITCH
3337 007056 012737 001212 001140  MOV   #$$SWREG,SWR  ;;NO,USE APT SWITCH REGISTER
3338 007064          67$:
3339      .SBTTL  TYPE PROGRAM NAME
3340      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
3341 007064 005227 177777          INC   #-1           ;;FIRST TIME?
3342 007070 001040                BNE   68$           ;;BRANCH IF NO
3343 007072 104401 007100          TYPE  ,69$        ;;TYPE ASCIZ STRING
3344 007076 000435                BR    68$           ;;GET OVER THE ASCIZ
3345      ;;69$: .ASCIZ <CRLF>'RK611/RK06-RK07 PERFORMANCE EXERCISER: MAINDEC CZR6P-D'<CRLF>
3346 007172          68$:
3347 007172 012737 000000 000036  MOV   #PRO,TRAPVEC+2 ;SET TRAP VECTOR TO PRIORITY 0

```


3460	007734	005020		5\$:	CLR	(R0)+		:CLEAR TRACK EXCLUSION ENTRY
3461	007736	005301			DEC	R1		:DECREMENT COUNT
3462	007740	001375			BNE	5\$:CHECK IF WHOLE BLOCK CLEARED
3463	007742	005304			DEC	R4		:DECREMENT COUNT
3464	007744	001357			BNE	4\$:TEST IF DONE
3465	007746	122737	000001	001361	CMPB	#1,FLAG		:CHECK IF PARAMETER START
3466	007754	001402			BEQ	10\$:YES, ASK FOR PARAMETERS
3467	007756	000137	011110		JMP	S1\$:USE DEFAULT VALUES
3468								
3469	007762	104401	057557		10\$:	TYPE	,SYS000	:TYPE "CPU ID"
3470	007766	004037	020516		JSR	R0,TSTDEF		:CHECK FOR DEFAULT PARAMETERS
3471	007772	010022			11\$:COMMA DETECTED
3472	007774	010040			13\$:CARRIAGE RETURN DETECTED
3473	007776	011110			S1\$:CONTROL Z <^Z> DETECTED
3474	010000	007762			10\$:CONTROL C <^C> DETECTED
3475	010002	010346			MOV	R3,-(SP)		:LOAD BUFFER ADDRESS ON STACK
3476	010004	004737	054406		JSR	PC,OCTBIN		:CONVERT TO BINARY
3477	010010	010022			11\$:ERROR RETURN
3478	010012	012604			MOV	(SP)+,R4		:STORE INPUT VALUE
3479	010014	022704	000400		CMP	#400,R4		:CHECK IF 0-377
3480	010020	101010			BHI	14\$:YES, LOAD CPU ID
3481	010022				11\$:			
3482	010022	010337	010030		MOV	R3,64\$:LOAD BUFFER ADDRESS
3483	010026	104401			TYPE			:TYPE RECEIVED INPUT
3484	010030	000000			64\$:	.WORD	0	:BUFFER ADDRESS
3485	010032	104401	001164		TYPE	,SQUES		:TYPE QUESTION MARK
3486	010036	000751			BR	10\$:TRY AGAIN
3487								
3488	010040	005004			13\$:	CLR	R4	:CLEAR CPU ID (R4)
3489	010042	110437	001364		14\$:	MOVB	R4,CPUID	:LOAD CPU ID
3490								
3491	010046	013737	055044	001674	15\$:	MOV	\$LSTAD,FBLKT+2	:LOAD LAST ADDRESS IN FIRST
3492								:FREE BLOCK ENTRY
3493	010054	104401	057567		TYPE	,SYS001		:TYPE "OVERLAY LOADER"
3494	010060	004037	020516		JSR	R0,TSTDEF		:CHECK FOR DEFAULT PARAMETERS
3495	010064	010134			18\$:COMMA DETECTED
3496	010066	010102			16\$:CARRIAGE RETURN DETECTED
3497	010070	011122			S2\$:CONTROL Z <^Z> DETECTED
3498	010072	007762			10\$:CONTROL C <^C> DETECTED
3499	010074	122713	000131		CMPB	#'Y,(R3)		:CHECK IF YES
3500	010100	001424			BEQ	19\$:YES, INDICATE THAT LOADER HAS BEEN OVERLAYED
3501	010102	105037	001365		16\$:	CLRB	OVLYLD	:INDICATE THAT LOADER HAS NOT BEEN OVERLAYED
3502	010106	105737	000041		TSTB	41		:SEE WHO LOADED THE PROGRAM
3503	010112	001004			BNE	17\$:BRANCH IF XXDP
3504	010114	162737	000300	001674	SUB	#96.*2.,FBLKT+2		:SUBTRACT ABS LOADER SIZE
3505	010122	000416			BR	20\$:ASK FOR MAXIMUM TRANSFER SIZE
3506								
3507	010124	162737	006000	001674	17\$:	SUB	#1536.*2.,FBLKT+2	:SUBTRACT XXDP LOADER SIZE
3508	010132	000412			BR	20\$:ASK FOR TRANSFER SIZE
3509								
3510	010134				18\$:			
3511	010134	010337	010142		MOV	R3,65\$:LOAD BUFFER ADDRESS
3512	010140	104401			TYPE			:TYPE RECEIVED INPUT
3513	010142	000000			65\$:	.WORD	0	:BUFFER ADDRESS
3514	010144	104401	001164		TYPE	,SQUES		:TYPE QUESTION MARK
3515	010150	000736			BR	15\$:TRY AGAIN

3516										
3517	010152	112737	177777	001365	19\$:	MOVB	#-1,OVLYLD		:INDICATE THAT LOADER HAS BEEN OVERLAYED	
3518										
3519	010160	104401	057607		20\$:	TYPE	,SYS002		:TYPE 'MAXIMUM TRANSFER'	
3520	010164	013746	001674			MOV	FBLKT+2,-(SP)		:STORE LAST USABLE MEMORY LOCATION	
3521	010170	163716	001672			SUB	FBLKT,(SP)		:CALCULATE MEMORY LEFT	
3522	010174	022716	027000			CMP	#256.*23.*2.,(SP)		:CHECK IF GREATER THAN OF EQUAL TO 5888	
3523	010200	101402				BLOS	21\$:YES, USE 5888	
3524	010202	006216				ASR	(SP)		:DETERMINE NUMBER OF WORDS	
3525	010204	000402				BR	22\$:TYPE VALUE	
3526										
3527	010206	012716	013400		21\$:	MOV	#256.*23.,(SP)		:LOAD 5888	
3528										
3529	010212	012637	001570		22\$:	MOV	(SP)+,T MPCNT		:SAVE TRANSFER COUNT	
3530	010216	013746	001570			MOV	T MPCNT,-(SP)		:SAVE TRANSFER COUNT FOR TYPE OUT	
3531	010222	004737	053224			JSR	PC,BIN OCT		:CONVERT TO OCTAL	
3532	010226	104401	057641			TYPE	,SYS003			
3533	010232	004037	020516			JSR	R0,TSTDEF		:CHECK FOR DEFAULT PARAMETERS	
3534	010236	010300				24\$:COMMA DETECTED	
3535	010240	010316				26\$:CARRIAGE RETURN DETECTED	
3536	010242	011152				S3\$:CONTROL Z <^Z> DETECTED	
3537	010244	007762				10\$:CONTROL C <^C> DETECTED	
3538	010246	010346				MOV	R3,-(SP)		:STORE BUFFER ADDRESS ON STACK	
3539	010250	004737	054406			JSR	PC,OCTBIN		:CONVERT TO BINARY	
3540	010254	010276				23\$:ERROR RETURN	
3541	010256	012604				MOV	(SP)+,R4		:STORE INPUT	
3542	010260	001407				BEQ	24\$:CHECK IF ZERO	
3543	010262	020437	001570			CMP	R4,T MPCNT		:CHECK IF LEGAL ENTRY	
3544	010266	101004				BHI	24\$:NO, TRY AGAIN	
3545	010270	010437	001354			MOV	R4,MAXBUF		:LOAD MAXIMUM TRANSFER	
3546	010274	000422				BR	27\$:GET NO. OF SOFT COMPARES	
3547										
3548	010276	005726			23\$:	TST	(SP)+		:ADJUST STACK	
3549	010300				24\$:					
3550	010300	010337	010306			MOV	R3,66\$:LOAD BUFFER ADDRESS	
3551	010304	104401				TYPE			:TYPE RECEIVED INPUT	
3552	010306	000000			66\$:	.WORD	0		:BUFFER ADDRESS	
3553	010310	104401	001164			TYPE	,SQUES		:TYPE QUESTION MARK	
3554	010314	000721				BR	20\$:TRY AGAIN	
3555										
3556	010316	013737	001570	001354	26\$:	MOV	T MPCNT,MAXBUF		:LOAD MAXIMUM TRANSFER	
3557	010324	022737	013400	001354		CMP	#256.*23.,MAXBUF		:CHECK IF LESS THAN OR EQUAL TO 23 SECTORS	
3558	010332	103003				BHIS	27\$:YES, USE CALCULATED VALUE	
3559	010334	012737	013400	001354		MOV	#256.*23.,MAXBUF		:USE 23 SECTORS	
3560										
3561	010342	104401	057645		27\$:	TYPE	,SYS004		:TYPE 'NO. OF SOFTWARE COMPARISONS'	
3562	010346	004037	020516			JSR	R0,TSTDEF		:CHECK FOR DEFAULT PARAMETERS	
3563	010352	010410				29\$:COMMA DETECTED	
3564	010354	010426				30\$:CARRIAGE RETURN DETECTED	
3565	010356	011210				S4\$:CONTROL Z <^Z> DETECTED	
3566	010360	007762				10\$:CONTROL C <^C> DETECTED	
3567	010362	010346				MOV	R3,-(SP)		:STORE PARAMETER	
3568	010364	004737	054542			JSR	PC,DEC BIN		:CONVERT TO BINARY	
3569	010370	010410				29\$:ERROR RETURN	
3570	010372	023716	001354			CMP	MAXBUF,(SP)		:CHECK IF LEGAL INPUT	
3571	010376	103403				BLO	28\$:TYPE VALUE AND RETRY	

3572	010400	012637	001362		MOV	(SP)+,SOFCMP	:LOAD NUMBER OF SOFTWARE COMPARES
3573	010404	000413			BR	31\$:DETERMINE RK06 UNIBUS ADDRESS
3574							
3575	010406	005726		28\$:	TST	(SP)+	:ADJUST STACK
3576	010410			29\$:			
3577	010410	010337	010416		MOV	R3,67\$:LOAD BUFFER ADDRESS
3578	010414	104401			TYPE		:TYPE RECEIVED INPUT
3579	010416	000000		67\$:	.WORD	0	:BUFFER ADDRESS
3580	010420	104401	001164		TYPE	\$QUES	:TYPE QUESTION MARK
3581	010424	000746			BR	27\$:TRY AGAIN
3582							
3583	010426	012737	000003	001362	30\$:	MOV	#3,SOFCMP
3584							:LOAD DEFAULT VALUE
3585	010434	105737	001374	31\$:	TSTB	CLKFLG	:CHECK IF CLOCK ON SYSTEM
3586	010440	001434			BEQ	41\$:NO, CONTINUE
3587	010442	104401	060040		TYPE	,SYS009	:TYPE "STATISTIC INTERVAL="
3588	010446	004037	020516		JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
3589	010452	010510			34\$:COMMA DETECTED
3590	010454	010526			35\$:CARRIAGE RETURN DETECTED
3591	010456	011216			S5\$:CONTROL Z <^Z> DETECTED
3592	010460	007762			10\$:CONTROL C <^C> DETECTED
3593	010462	010346			MOV	R3,-(SP)	:STORE NUMBER FOR CONVERSION
3594	010464	004737	054542		JSR	PC,DECBIN	:CONVERT TO BINARY
3595	010470	010510			34\$:ILLEGAL NUMBER
3596	010472	012604			MOV	(SP)+,R4	:STORE NUMBER
3597	010474	022704	000377		CMP	#255.,R4	:CHECK IF GREATER THAN 255
3598	010500	103403			BLO	34\$:NO, TRY AGAIN
3599	010502	110437	001360		MOVB	R4,PERINV	:LOAD PERFORMANCE INTERVAL
3600	010506	000411			BR	41\$:CONTINUE
3601							
3602	010510			34\$:			
3603	010510	010337	010516		MOV	R3,68\$:LOAD BUFFER ADDRESS
3604	010514	104401			TYPE		:TYPE RECEIVED INPUT
3605	010516	000000		68\$:	.WORD	0	:BUFFER ADDRESS
3606	010520	104401	001164		TYPE	\$QUES	:TYPE QUESTION MARK
3607	010524	000743			BR	31\$:TRY AGAIN
3608							
3609	010526	105037	001360	35\$:	CLRB	PERINV	:SET UP FOR NO INTERVAL STATISTICS
3610							
3611	010532	104401	057677	41\$:	TYPE	,SYS005	:TYPE "RK06 BUS ADD."
3612	010536	004037	020516		JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
3613	010542	010656			44\$:COMMA DETECTED
3614	010544	010674			45\$:CARRIAGE RETURN DETECTED
3615	010546	011222			S6\$:CONTROL Z <^Z> DETECTED
3616	010550	007762			10\$:CONTROL C <^C> DETECTED
3617	010552	010346			MOV	R3,-(SP)	:STORE PARAMETER
3618	010554	004737	054406		JSR	PC,OCTBIN	:CONVERT TO BINARY
3619	010560	010656			44\$:ERROR RETURN
3620	010562	012604			MOV	(SP)+,R4	:STORE INPUT
3621	010564	022704	160000		CMP	#160000,R4	:CHECK IF I/O PAGE
3622	010570	101032			BHI	44\$:NO, TRY AGAIN
3623	010572	012737	010632	000004	MOV	#42\$,ERRVEC	:LOAD ERRVEC TO NON-EXISTENT MEMORY
3624	010600	013737	000340	000006	MOV	PR7,ERRVEC+2	: PRIORITY 7
3625	010606	005714			TST	(R4)	:CHECK FOR NON-EXISTENT MEMORY
3626	010610	012737	000000	000006	MOV	#<HALT>,ERRVEC+2	:REINSTATE TRAP CATCHER
3627	010616	012737	000006	000004	MOV	#ERRVEC+2,ERRVEC	

3628	010624	010437	001446			MOV	R4,RKBAS	:LEGAL I/O ADDRESS, LOAD RK06 BASE
3629	010630	000424				BR	48\$:GET VECTOR ADDRESS
3630								
3631	010632	012737	000000	000006	42\$:	MOV	#<HALT>,ERRVEC+2	:REINSTATE TRAP CATCHER
3632	010640	012737	000006	000004		MOV	#ERRVEC+2,ERRVEC	
3633	010646	062706	000004			ADD	#4,SP	:ADJUST STACK
3634	010652	104401	057745			TYPE	,SYS007	:TYPE 'NON-EXISTENT MEMORY'
3635	010656				44\$:			
3636	010656	010337	010664			MOV	R3,69\$:LOAD BUFFER ADDRESS
3637	010662	104401				TYPE		:TYPE RECEIVED INPUT
3638	010664	000000			69\$:	.WORD	0	:BUFFER ADDRESS
3639	010666	104401	001164			TYPE	,SQUES	:TYPE QUESTION MARK
3640	010672	000717				BR	41\$:TRY AGAIN
3641								
3642	010674	012737	177440	001446	45\$:	MOV	#177440,RKBAS	:LOAD DEFAULT BUS ADDRESS
3643								
3644	010702	104401	057722		48\$:	TYPE	,SYS006	:TYPE 'RK06 VEC ADD'
3645	010706	004037	020516			JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
3646	010712	010750				50\$:COMMA DETECTED
3647	010714	010766				51\$:CARRIAGE RETURN DETECTED
3648	010716	011230				57\$:CONTROL Z <^Z> DETECTED
3649	010720	007762				10\$:CONTROL C <^C> DETECTED
3650	010722	010346				MOV	R3,-(SP)	:STORE PARAMETER
3651	010724	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
3652	010730	010750				50\$:ERROR RETURN
3653	010732	022716	001000			CMP	#1000,(SP)	:CHECK IF VECTOR SPACE
3654	010736	101403				BLOS	49\$:NO, TRY AGAIN
3655	010740	012637	001450			MOV	(SP)+,RKVEC	:LOAD VECTOR ADDRESS
3656	010744	000413				BR	53\$:GET PRIORITY
3657								
3658	010746	005726			49\$:	TST	(SP)+	:ADJUST STACK
3659	010750				50\$:			
3660	010750	010337	010756			MOV	R3,70\$:LOAD BUFFER ADDRESS
3661	010754	104401				TYPE		:TYPE RECEIVED INPUT
3662	010756	000000			70\$:	.WORD	0	:BUFFER ADDRESS
3663	010760	104401	001164			TYPE	,SQUES	:TYPE QUESTION MARK
3664	010764	000746				BR	48\$:TRY AGAIN
3665								
3666	010766	012737	000210	001450	51\$:	MOV	#210,RKVEC	:LOAD DEFAULT RK06 VECTOR
3667	010774	104401	060064		53\$:	TYPE	,SYS010	:TYPE 'RK06 PRIOR='
3668	011000	004037	020516			JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
3669	011004	011062				56\$:COMMA DETECTED
3670	011006	011100				57\$:CARRIAGE RETURN DETECTED
3671	011010	011236				58\$:CONTROL Z <^Z> DETECTED
3672	011012	007762				10\$:CONTROL C <^C> DETECTED
3673	011014	010346				MOV	R3,-(SP)	:STORE PARAMETER
3674	011016	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
3675	011022	011062				56\$:ERROR RETURN
3676	011024	022716	000007			CMP	#7,(SP)	:CHECK IF OCTAL
3677	011030	103413				BLO	55\$	
3678	011032	022716	000004			CMP	#4,(SP)	
3679	011036	101010				BHI	55\$	
3680	011040	006316				ASL	(SP)	:SHIFT 5 BITS LEFT
3681	011042	006316				ASL	(SP)	
3682	011044	006316				ASL	(SP)	
3683	011046	006316				ASL	(SP)	

3684	011050	006316				ASL	(SP)	
3685	011052	012637	001452			MOV	(SP)+,RKPRI	:LOAD PRIORITY
3686	011056	000472				BR	REST1	:LOAD RK06 VECTOR AND TTY VECTOR
3687								
3688	011060	005726			55\$:	TST	(SP)+	:ADJUST STACK
3689	011062				56\$:			
3690	011062	010337	011070			MOV	R3,71\$:LOAD BUFFER ADDRESS
3691	011066	104401				TYPE		:TYPE RECEIVED INPUT
3692	011070	000000			71\$:	.WORD	0	:BUFFER ADDRESS
3693	011072	104401	001164			TYPE	,SQUES	:TYPE QUESTION MARK
3694	011076	000736				BR	53\$:TRY AGAIN
3695								
3696	011100	012737	000240	001452	57\$:	MOV	#PR5,RKPRI	:LOAD PRIORITY
3697	011106	000456				BR	REST1	:LOAD RK06 VECTOR AND TTY VECTOR
3698								
3699	011110	105037	001364		S1\$:	CLRB	CPUID	:LOAD DEFAULT CPU ID
3700	011114	013737	055044	001674		MOV	\$LSTAD,FBLKT+2	:LOAD LAST ADDRESS
3701	011122	105037	001365		S2\$:	CLRB	OVLYLD	:INDICATE THAT THE LOADER IS NOT OVERLAYED
3702	011126	105737	000041			TSTB	41	:SEE WHO LOADED PROGRAM
3703	011132	001004				BNE	1\$:BRANCH IF XXDP
3704	011134	162737	000300	001674		SUB	#96.*2.,FBLKT+2	:SUBTRACT ABS LOADER SIZE
3705	011142	000403				BR	S3\$:GO DETERMINE MAX BUFFER SIZE
3706								
3707	011144	162737	006000	001674	1\$:	SUB	#1536.*2.,FBLKT+2	:SUBTRACT XXDP LOADER SIZE
3708	011152	013746	001674		S3\$:	MOV	FBLKT+2,-(SP)	:STORE MAXIMUM ADDRESS
3709	011156	163716	001672			SUB	FBLKT,(SP)	:DETERMINE MEMORY AVAILIABLE
3710	011162	022716	027000			CMP	#256.*23.*2.,(SP)	:CHECK IF ENOUGH ROOM
3711	011166	101005				BHI	1\$:NO,LOAD MEMORY AVILIABLE
3712	011170	005726				TST	(SP)+	:ADJUST STACK
3713	011172	012737	013400	001354		MOV	#256.*23.,MAXBUF	:LOAD MAXIMUM BUFFER SIZE
3714	011200	000403				BR	S4\$:LOAD NUMBER OF SOFTWARE COMPARES
3715								
3716	011202	006216			1\$:	ASR	(SP)	:CHANGE TO WORDS
3717	011204	012637	001354			MOV	(SP)+,MAXBUF	:LOAD MAXIMUM WORDS TRANSFERRED
3718	011210	012737	000003	001362	S4\$:	MOV	#3,SOF CMP	:LOAD NO. OF SOFTWARE COMPARES
3719	011216	105037	001360		S5\$:	CLRB	PERINV	:SET UP FOR NO INTERVAL STATISTICS
3720	011222	012737	177440	001446	S6\$:	MOV	#177440,RKBAS	:LOAD DEFAULT BUS ADDRESS
3721	011230	012737	000210	001450	S7\$:	MOV	#210,RKVEC	:LOAD DEFAULT VECTOR ADDRESS
3722	011236	012737	000240	001452	S8\$:	MOV	#PR5,RKPRI	:LOAD RK06 PRIORITY
3723								
3724	011244	013704	001450		REST1:	MOV	RKVEC,R4	:STORE VECTOR ADDRESS
3725	011250	012724	044444			MOV	#I.INTR,(R4)+	:LOAD RK06 VECTOR ADDRESS
3726	011254	013714	001452			MOV	RKPRI,(R4)	: AND INTERRUPT PRIORITY
3727	011260	105737	001361		REST2:	TSTB	FLAG	
3728	011264	100022				BPL	99\$	
3729	011266	122737	000013	000041		CMPB	#13,@#41	:LOAD FROM XXDP ?
3730	011274	001016				BNE	99\$:BRANCH IF NOT
3731	011276	104401	056224			TYPE	,XXDPMG	
3732	011302	000000				HALT		
3733	011304	122737	000013	000041		CMPB	#13,@#41	:DID THEY CHANGE THE PACK?
3734	011312	001007				BNE	99\$:CONTINUE IF THEY DID
3735	011314	112737	000000	056463		MOVB	#0,OPR001	:LOAD DRIVE TO BE DROPPED
3736	011322	010446				MOV	R4,-(SP)	:SAVE R4
3737	011324	010546				MOV	R5,-(SP)	:SAVE R5
3738	011326	004737	013162			JSR	PC,DROP	:GO DROP DRIVE 0
3739	011332				99\$:			

3740	011332	004737	043642			JSR	PC,CLKINT	:INITIALIZE CLOCK VECTOR ADDRESS
3741	011336	104401	057772			TYPE	,SYS008	:TYPE 'READY TO BEGIN TESTING'
3742	011342	005227	177777			INC	#-1	:TYPE HELP MESSAGE ON FIRST PASS
3743	011346	001002				BNE	5\$	
3744	011350	104401	066404			TYPE	,HELP	
3745	011354	004737	054002		5\$:	JSR	PC,\$TKINT	:SET UP TTY INTERRUPT
3746	011360	013737	001452	000036		MOV	RKPRI,TRAPVEC+2	:ALLOW L AND P CLOCK INTERRUPTS
3747	011366	005037	177776			CLR	PS	:ALLOW RK06 INTERRUPTS
3748								
3749	011372	122737	000002	001361		CMPB	#2,FLAG	:SEE IF AUTO SIZE
3750	011400	001401				BEQ	1\$:BR IF YES
3751	011402	000406				BR	MAIN	:ELSE GOTO MAIN
3752	011404	012746	011420		1\$:	MOV	#MAIN,-(SP)	:SETUP STACK TO RETRUN TO MAIN
3753	011410	010446				MOV	R4,-(SP)	:SAVE R4
3754	011412	010546				MOV	R5,-(SP)	:SAVE R5
3755	011414	000137	012666			JMP	WRTEPK+10	:WRITE ALL DRIVES ON BUS & RET TO MAIN
3756								

```

3757          .SBTTL  MAIN IDLE LOOP
3758
3759 011420 022706 001100      MAIN:  CMP      #STACK,SP      ;CHECK IF STACK PROBLEM
3760 011424 001401              BEQ      1$          ;NO, CONTINUE
3761 011426 000000              HALT                    ; *** PROGRAM PROBLEM
3762 011430 004737 052116      1$:   JSR      PC,C.OPT      ;CALL COMMAND OPTIMIZER
3763 011434 004737 044270      JSR      PC,W.WTCH      ;CALL WATCH DOG TIMER
3764 011440 005737 001564      TST      ERCONT        ;CHECK IF IN SPECIAL ERROR SEQUENCE
3765 011444 100765              BMI      MAIN         ;YES, DO NOT PRINT INTERVAL STATISTICS OR
3766                                ; ADD DRIVES TO TESTING SEQUENCE
3767 011446 032777 060000 167464 BIT      #SW14!SW13,@SWR ;CHECK IF LOOP ON OPERATION OR
3768                                ; INHIBIT PRINT OUT
3769 011454 001022              BNE      11$         ;YES, DO NOT PRINT OUT INTERVAL STATISTICS
3770 011456 105737 001633      TSTB     STATIS        ;CHECK FOR INTERVAL STATISTICS
3771 011462 001417              BEQ      11$         ;NO, CONTINUE SCAN
3772 011464 005004              CLR      R4          ;START WITH DRIVE 0
3773 011466 016405 001524      2$:   MOV      PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
3774 011472 032765 000001 000014 BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IS IN USE
3775 011500 001402              BEQ      3$          ;NO, LOOK AT NEXT PARAMETER BLOCK
3776 011502 004737 013260      JSR      PC,STAT      ;GET STATISTICS
3777 011506 005724              TST      (R4)+        ;INCREMENT DRIVE INDEX
3778 011510 022704 000020      3$:   CMP      #20,R4     ;CHECK IF ALL DRIVES SCANNED
3779 011514 101364              BHI      2$          ;NO, DO NEXT DRIVE
3780 011516 105037 001633      CLRB     STATIS        ;RESET FLAG
3781
3782 011522 004037 051644      11$:  JSR      R0,Q.POP      ;GET FIRST DRIVE FROM DRIVE
3783 011526 001314              AVAILQ                    ; AVAILABLE QUEUE
3784 011530 012605              MOV      (SP)+,R5      ;STORE PARAMETER BLOCK ADDRESS
3785 011532 001732              BEQ      MAIN         ;IF QUEUE EMPTY WAIT
3786 011534 032765 002000 000014 BIT      #DRPDRV,P.PRST(R5) ;CHECK IF DROP DRIVE
3787 011542 001403              BEQ      12$         ;NO, GENERATE NEW COMMAND
3788 011544 004737 013162      JSR      PC,DROP      ;DROP DRIVE
3789 011550 000723              BR       MAIN         ;GET NEXT AVAILABLE DRIVE
3790
3791 011552 032777 040000 167360 12$:  BIT      #SW14,@SWR    ;CHECK IF LOOP ON OPERATION
3792 011560 001002              BNE      15$         ;YES, DO NOT CLEAR SEEK TO PREVIOUS
3793                                ; CYLINDER FLAG
3794 011562 105065 000211      15$:  CLRB     P.SEEK(R5)   ;CLEAR SEEK TO PREVIOUS CYLINDER
3795 011566 004737 021336      JSR      PC,GENERT    ;GENERATE NEW COMMAND
3796 011572 105765 000211      TSTB     P.SEEK(R5)   ;CHECK IF SEEK TO PREVIOUS CYLINDER
3797 011576 001310              BNE      MAIN         ;YES, GET NEXT AVAILIABLE DRIVE
3798 011600 013737 001452 177776 MOV      RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
3799 011606 005737 001320      TST      BWAITQ      ;CHECK IF BUFFER WAIT QUEUE EMPTY
3800 011612 001013              BNE      25$         ;NO, ENQUEUE PARAMETER BLOCK
3801 011614 004037 022616      JSR      R0,BUFFAL    ;ALLOCATE BUFFER
3802 011620 011642              25$:  JSR      R0,BUFFAL    ;NOT ENOUGH ROOM RETURN
3803 011622 005037 177776      CLR      PS          ;ALLOW RK06 INTERRUPTS
3804 011626 004737 023234      JSR      PC,LDDATA    ;LOAD BUFFER
3805 011632 004037 051564      JSR      R0,Q.PUSH    ;PUT PARAMETER BLOCK ON COMMAND
3806 011636 001324              CINITQ                    ; INITIATION QUEUE
3807 011640 000667              BR       MAIN         ;GET NEXT AVAILABLE DRIVE
3808
3809 011642 004037 051564      25$:  JSR      R0,Q.PUSH    ;PUT PARAMETER BLOCK ON BUFFER
3810 011646 001320              BWAITQ                    ; WAIT QUEUE
3811 011650 005037 177776      CLR      PS          ;ALLOW RK06 INTERRUPTS
3812 011654 000661              BR       MAIN         ;GET NEXT AVAILABLE DRIVE

```

.SBTTL OPERATOR COMMAND DECODER

```
*****
*
* THE LEGAL COMMANDS ARE:
*   TN-  INITIATE TESTING ON DRIVE N
*   PN-  CHANGE PARAMETER AND INITIATE TEST ON
*         DRIVE N
*   DN-  DROP DRIVE FROM TESTING SEQUENCE
*   SN-  DEMAND PERFORMANCE SUMMARY ON DRIVE N
*   WN-  WRITE RANDOM DATA PATTERNS ON DRIVE N
*         AND INITIATE TESTING.
*
*CALL JSR PC,OPRCMD
*      RETURN
*
* COMMAND          ROUTINE
*-----
*   TN             TESTDV
*   PN             PARMDV
*   DN             DROPDV
*   SN             STATDV
*   WN             WRTEPK
*****
```

3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841 011656 010446
3842 011660 104401 056626
3843 011664 104403
3844 011666 012604
3845 011670 116437 000001 056463
3846
3847 011676 122737 000101 056463
3848 011704 001421
3849 011706 122737 000060 056463
3850 011714 101004
3851 011716 122737 000067 056463
3852 011724 103006
3853 011726 104401 056446 2\$:
3854 011732 104401 056466
3855 011736 012604
3856 011740 000207
3857
3858 011742 142737 000370 056463 3\$:
3859 011750 010546 4\$:
3860 011752 122714 000124
3861 011756 001427
3862 011760 122714 000120
3863 011764 001512
3864 011766 122714 000104
3865 011772 001553
3866 011774 122714 000123
3867 012000 001002
3868 012002 000137 012502

```
OPRCMD: MOV R4, -(SP) ;STORE R4 ON STACK
          TYPE ,OPR007 ;TYPE 'PLEASE ENTER COMMAND'
          RDLIN ;READ LINE FROM TTY
          MOV (SP)+,R4 ;STORE ADDRESS OF INPUT STRING
          MOV 1(R4),OPR001 ;MOVE SECOND CHARACTER TO DRIVE
          ; REQUEST STORAGE
          CMPB #'A',OPR001 ;CHECK IF ALL DRIVES
          BEQ 4$ ;YES, PROCESS ALL DRIVES
          CMPB #'0',OPR001 ;CHECK IF LEGAL DRIVE NUMBER
          BHI 2$ ;NO, PRINT ERROR
          CMPB #'7',OPR001 ;CHECK IF 0-7
          BHIS 3$ ;YES, PROCESS DRIVE COMMAND
          2$: TYPE ,OPR000 ;TYPE 'DRIVE NUMBER N ILLEGAL?'
              TYPE ,OPR002
          MOV (SP)+,R4 ;RESTORE R4
          RTS PC ;RETURN
          3$: BICB #370,OPR001 ;CLEAR UNIMPORTANT BITS
              MOV R5, -(SP) ;STORE R5 ON STACK
              CMPB #'T',(R4) ;CHECK IF TEST DRIVE
              BEQ TESTDV ;YES, GO TO TEST DRIVE
              CMPB #'P',(R4) ;CHECK IF TEST AND CHANGE PARAMETERS
              BEQ PARMDV ;YES, GO TEST AND CHANGE PARAMETERS
              CMPB #'D',(R4) ;CHECK IF DROP DRIVE
              BEQ DROPDV ;YES, GO DROP DRIVE
              CMPB #'S',(R4) ;CHECK IF DEMAND STATISTICS
              BNE 5$
              JMP STATDV ;YES, GO TO DEMAND STATISTICS
```

3869							
3870	012006	122714	000127	5\$:	CMPB	#'W,(R4)	:CHECK IF WRITE PACK AND TEST
3871	012012	001002			BNE	6\$	
3872	012014	000137	012656		JMP	WRTEPK	:YES, GO WRITE PACK AND TEST
3873							
3874	012020	111437	056440	6\$:	MOVB	(R4),ILLCMD	:STORE COMMAND FOR PRINT OUT
3875	012024	104401	056407		TYPE	,ILLCOM	:TYPE ERROR MESSAGE
3876	012030	012605			MOV	(SP)+,R5	:RESTORE R5
3877	012032	012604			MOV	(SP)+,R4	:RESTORE R4
3878	012034	000207			RTS	PC	:RETURN

```

3879          .SBTTL TEST DRIVE COMMAND
3880
3881 012036 122737 000101 056463 TESTDV: CMPB #'A,OPR001 ;CHECK IF ALL DRIVES ARE TO BE TESTED
3882 012044 001027          BNE 10$ ;NO, DO SPECIFIC DRIVE
3883 012046 105037 001632          CLR R4 ;CLEAR DRIVE COUNT
3884 012052 005004          CLR R4 ;CLEAR DRIVE COUNT
3885 012054 016405 001524 1$: MOV PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
3886 012060 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
3887 012066 001004          BNE 2$ ;YES, GO TO NEXT DRIVE
3888 012070 004737 013536          JSR PC,TEST ;GO TEST DRIVE
3889 012074 105237 001632          INCB DRVCNT ;INCREMENT DRIVE COUNT
3890 012100 005724          2$: TST (R4)+ ;ADDRESS NEXT DRIVE (ADD 2)
3891 012102 022704 000020          CMP #20,R4 ;CHECK IF FINISHED
3892 012106 001362          BNE 1$ ;NO, SET UP NEXT DRIVE
3893 012110 105737 001632          TSTB DRVCNT ;CHECK IF ANY DRIVE IN ASSIGNMENT SEQUENCE
3894 012114 001033          BNE 15$ ;YES, RETURN
3895 012116 104401 057005          TYPE ,OPR013 ;TYPE "ALL DRIVES CURRENTLY UNDER TEST?"
3896 012122 000430          BR 15$ ;RETURN
3897
3898 012124 113704 056463          10$: MOVB OPR001,R4 ;STORE DRIVE NUMBER
3899 012130 006304          ASL R4 ;MULTIPLY DRIVE NUMBER BY 2
3900 012132 016405 001524          MOV PBLKT(R4),R5 ;GET PARAMETER ADDRESS
3901 012136 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE ASSIGNED
3902 012144 001003          BNE 11$ ;YES, PRINT DRIVE ALREADY ASSIGNED
3903 012146 004737 013536          JSR PC,TEST ;GO TEST DRIVE
3904 012152 000414          BR 15$ ;RETURN
3905
3906 012154 013737 001452 177776 11$: MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS
3907 012162 152737 000060 056463 BISB #60,OPR001 ;MAKE NUMBER ASCIZ
3908 012170 104401 056446          TYPE ,OPR000 ;TYPE "DRIVE N ALREADY ASSIGNED?"
3909 012174 104401 056514          TYPE ,OPR004
3910 012200 005037 177776          CLR PS ;ALLOW RK06 INTERRUPTS
3911 012204 012605          15$: MOV (SP)+,R5 ;RESTORE R5
3912 012206 012604          MOV (SP)+,R4 ;RESTORE R4
3913 012210 000207          RTS PC ;RETURN
3914
3915          .SBTTL CHANGE PARAMETERS COMMAND
3916
3917 012212 122737 000101 056463 PARMDV: CMPB #'A,OPR001 ;CHECK IF PARAMETERS FOR ALL DRIVES
3918 012220 001020          BNE 10$ ;NO, DO SPECIFIC DRIVE
3919 012222 005004          CLR R4 ;CLEAR DRIVE COUNT
3920 012224 016405 001524 1$: MOV PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
3921 012230 004737 016346          JSR PC,PARM ;ASK FOR PARAMETERS
3922 012234 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
3923 012242 001002          BNE 2$ ;YES, GO TO NEXT DRIVE
3924 012244 004737 013536          JSR PC,TEST ;GO TEST DRIVE
3925 012250 005724          2$: TST (R4)+ ;ADDRESS NEXT DRIVE (ADD 2)
3926 012252 022704 000020          CMP #20,R4 ;CHECK IF FINISHED
3927 012256 001362          BNE 1$ ;NO, SET UP NEXT DRIVE
3928 012260 000415          BR 15$ ;RETURN
3929
3930 012262 113704 056463          10$: MOVB OPR001,R4 ;STORE DRIVE NUMBER
3931 012266 006304          ASL R4 ;MULTIPLY DRIVE NUMBER BY 2
3932 012270 016405 001524          MOV PBLKT(R4),R5 ;GET PARAMETER ADDRESS
3933 012274 004737 016346          JSR PC,PARM ;GET NEW PARAMETERS
3934 012300 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE

```

```
3935 012306 001002          BNE      15$          ;YES, RETURN
3936 012310 004737 013536   JSR      PC,TEST     ;GO TEST DRIVE
3937 012314 012605          MOV      (SP)+,R5    ;RESTORE R5
3938 012316 012604          MOV      (SP)+,R4    ;RESTORE R4
3939 012320 000207          RTS       PC         ;RETURN
3940
3941          .SBTTL  DROP DRIVE COMMAND
3942
3943 012322 122737 000101 056463 DROPDV: CMPB   #'A,OPR001    ;CHECK IF ALL DRIVES ARE TO BE DROPPED
3944 012330 001030          BNE      10$        ;NO, DO SPECIFIC DRIVE
3945 012332 105037 001632   CLR     DRVCNT      ;CLEAR DRIVE COUNT
3946 012336 005004          CLR     R4          ;CLEAR DRIVE COUNT
3947 012340 016405 001524   1$:  MOV     PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
3948 012344 032765 000001 000014 BIT     #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
3949 012352 001405          BEQ     2$          ;NO, GO TO NEXT DRIVE
3950 012354 052765 002000 000014 BIS     #DRPDRV,P.PRST(R5) ;INDICATE DRIVE IS TO BE DROPPED
3951 012362 105237 001632   INCB   DRVCNT      ;INCREMENT DRIVE COUNT
3952 012366 005724          TST     (R4)+       ;ADDRESS NEXT DRIVE (ADD 2)
3953 012370 022704 000020   CMP     #20,R4     ;CHECK IF FINISHED
3954 012374 001361          BNE     1$          ;NO, SET UP NEXT DRIVE
3955 012376 105737 001632   TSTB   DRVCNT      ;CHECK IF ANY DRIVES AVAILIABLE
3956 012402 001034          BNE     15$        ;YES, RETURN
3957 012404 104401 057047   TYPE   ,OPR014     ;TYPE 'NO DRIVES IN USE?'
3958 012410 000431          BR      15$        ;RETURN
3959
3960 012412 113704 056463   10$:  MOV     OPR001,R4   ;STORE DRIVE NUMBER
3961 012416 006304          ASL     R4          ;MULTIPLY DRIVE NUMBER BY 2
3962 012420 016405 001524   MOV     PBLKT(R4),R5 ;GET PARAMETER ADDRESS
3963 012424 032765 000001 000014 BIT     #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE ASSIGNED
3964 012432 001404          BEQ     11$        ;NO, PRINT DRIVE N NOT ASSIGNED
3965 012434 052765 002000 000014 BIS     #DRPDRV,P.PRST(R5) ;INDICATE DRIVE IS TO BE DROPPED
3966 012442 000414          BR      15$        ;RETURN
3967
3968 012444 013737 001452 177776 11$:  MOV     RKPRI,PS    ;LOCK OUT RK06 INTERRUPTS
3969 012452 152737 000060 056463 BISB   #60,OPR001   ;MAKE NUMBER ASCII
3970 012460 104401 056446   TYPE   ,OPR000     ;TYPE 'DRIVE N NOT ASSIGNED?'
3971 012464 104401 056540   TYPE   ,OPR005
3972 012470 005037 177776   CLR     PS         ;ALLOW RK06 INTERRUPTS
3973 012474 012605          15$:  MOV     (SP)+,R5    ;RESTORE R5
3974 012476 012604          MOV     (SP)+,R4    ;RESTORE R4
3975 012500 000207          RTS       PC         ;RETURN
3976
3977          .SBTTL  GET DRIVE STATISTICS COMMAND
3978
3979 012502 122737 000101 056463 STATDV: CMPB   #'A,OPR001    ;CHECK IF STATISTICS FOR ALL DRIVES
3980 012510 001027          BNE     10$        ;NO, DP SPECIFIC DRIVE
3981 012512 105037 001632   CLR     DRVCNT      ;CLEAR DRIVE COUNT
3982 012516 005004          CLR     R4          ;CLEAR DRIVE COUNT
3983 012520 016405 001524   1$:  MOV     PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
3984 012524 032765 000001 000014 BIT     #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
3985 012532 001404          BEQ     2$          ;NO, GO TO NEXT DRIVE
3986 012534 004737 013260   JSR     PC,STAT     ;GATHER STATISTICS
3987 012540 105237 001632   INCB   DRVCNT      ;INCREMENT DRIVE COUNT
3988 012544 005724          TST     (R4)+       ;ADDRESS NEXT DRIVE (ADD 2)
3989 012546 022704 000020   CMP     #20,R4     ;CHECK IF FINISHED
3990 012552 001362          BNE     1$          ;NO, SET UP NEXT DRIVE
```

```

3991 012554 105737 001632      TSTB  DRVCNT      ;CHECK IF ANY DRIVES AVAILIABLE
3992 012560 001033      BNE   15$        ;YES, RETURN
3993 012562 104401 057047      TYPE  ,OPR014   ;TYPE 'NO DRIVES IN USE?'
3994 012566 000430      BR    15$        ;RETURN
3995
3996 012570 113704 056463      10$:  MOVB  OPR001,R4 ;STORE DRIVE NUMBER
3997 012574 006304      ASL  R4          ;MULTIPLY DRIVE NUMBER BY 2
3998 012576 016405 001524      MOV  PBLKT(R4),R5 ;GET PARAMETER ADDRESS
3999 012602 032765 000001 000014      BIT  #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE ASSIGNED
4000 012610 001403      BEQ  11$        ;PRINT DRIVE N NOT ASSIGNED
4001 012612 004737 013260      JSR  PC,STAT    ;GO GATHER DRIVE STATISTICS
4002 012616 000414      BR   15$        ;RETURN
4003
4004 012620 013737 001452 177776 11$:  MOV  RKPRI,PS    ;LOCK OUT RK06 INTERRUPTS
4005 012626 152737 000060 056463      BISB #60,OPR001 ;MAKE NUMBER ASCII
4006 012634 104401 056446      TYPE ,OPR000   ;TYPE 'DRIVE N NOT ASSIGNED?'
4007 012640 104401 056540      TYPE ,OPR005
4008 012644 005037 177776      CLR  PS        ;ALLOW RK06 INTERRUPTS
4009 012650 012605      15$:  MOV  (SP)+,R5   ;RESTORE R5
4010 012652 012604      MOV  (SP)+,R4   ;RESTORE R4
4011 012654 000207      RTS  PC        ;RETURN
4012
4013      .SBTTL WRITE PACK AND TEST COMMAND
4014
4015 012656 122737 000101 056463 WRTEPK: CMPB  #'A,OPR001 ;CHECK IF ALL DRIVES ARE TO BE TESTED
4016 012664 001055      BNE  10$        ;NO, DO SPECIFIC DRIVE
4017 012666 105037 001632      CLRB DRVCNT     ;CLEAR DRIVE COUNT
4018 012672 005004      CLR  R4        ;CLEAR DRIVE COUNT
4019 012674 016405 001524      1$:  MOV  PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
4020 012700 032765 000001 000014      BIT  #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
4021 012706 001032      BNE  2$        ;YES, GO TO NEXT DRIVE
4022 012710 105765 000000      TSTB P.DRVN(R5)
4023 012714 001020      BNE  97$
4024 012716 122737 000013 000041      CMPB #13,@#41   ;LOAD FROM XXDP ?
4025 012724 001014      BNE  97$        ;BRANCH IF NOT
4026 012726 104401 056224      TYPE ,XXDPMG
4027 012732 000000      HALT
4028 012734 122737 000013 000041      CMPB #13,@#41   ;DID THEY CHANGE THE PACK?
4029 012742 001005      BNE  97$        ;CONTINUE IF THEY DID
4030 012744 152765 000001 000210      BISB #BIT0,P.ASSN(R5) ;SET UP FOR NO STATS
4031 012752 004737 013162      JSR  PC,DROP    ;GO DROP DRIVE 0
4032 012756 112765 000020 000210 97$:  MOVB #BIT4,P.ASSN(R5) ;SET FLAG FOR WRITE PACK
4033 012764 004737 013536      JSR  PC,TEST    ;GO ASSIGN DRIVE
4034 012770 105237 001632      INCB DRVCNT     ;INCREMENT DRIVE COUNT
4035 012774 005724      2$:  TST  (R4)+     ;ADDRESS NEXT DRIVE (ADD 2)
4036 012776 022704 000020      CMP  #20,R4    ;CHECK IF FINISHED
4037 013002 001334      BNE  1$        ;NO, SET UP NEXT COMMAND
4038 013004 105737 001632      TSTB DRVCNT    ;CHECK IF ANY DRIVES NOT IN USE
4039 013010 001461      BEQ  15$       ;NO, RETURN
4040 013012 104401 057005      TYPE ,OPR013   ;TYPE 'ALL DRIVES CURRENTLY UNDER TEST'
4041 013016 000456      BR   15$       ;RETURN
4042
4043 013020 113704 056463      10$:  MOVB  OPR001,R4 ;STORE DRIVE NUMBER
4044 013024 006304      ASL  R4          ;MULTIPLY DRIVE NUMBER BY 2
4045 013026 016405 001524      MOV  PBLKT(R4),R5 ;GET PARAMETER ADDRESS
4046 013032 032765 000001 000014      BIT  #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE ASSIGNED

```



```
4047 013040 001031          BNE      11$           ;YES, PRINT DRIVE ALREADY ASSIGNED
4048 013042 105765 000000    TSTB    P.DRVN(R5)
4049 013046 001020          BNE      99$
4050 013050 122737 000013 000041  CMPB    #13,@#41      ;LOAD FROM XXDP ?
4051 013056 001014          BNE      99$           ;BRANCH IF NOT
4052 013060 104401 056224    TYPE    ,XXDPMG
4053 013064 000000          HALT
4054 013066 122737 000013 000041  CMPB    #13,@#41      ;DID THEY CHANGE THE PACK?
4055 013074 001005          BNE      99$           ;CONTINUE IF THEY DID
4056 013076 152765 000001 000210  BISB    #BIT0,P.ASSN(R5);SET UP FOR NO STAT
4057 013104 004737 013162          JSR     PC,DROP       ;GO DROP DRIVE 0
4058 013110 112765 000020 000210 99$:   MOVB    #BIT4,P.ASSN(R5);SET FLAG FOR WRITE PACK
4059 013116 004737 013536          JSR     PC,TEST       ;GO ASSIGN DRIVE
4060 013122 000414          BR      15$           ;RETURN
4061
4062 013124 013737 001452 177776 11$:   MOV     RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
4063 013132 152737 000060 056463  BISB    #60,OPR001    ;MAKE NUMBER ASCII
4064 013140 104401 056446          TYPE    ,OPR000      ;TYPE 'DRIVE N ALREADY ASSIGNED?'
4065 013144 104401 056514          TYPE    ,OPR004
4066 013150 005037 177776          CLR     PS           ;ALLOW RK06 INTERRUPTS
4067 013154 012605          MOV     (SP)+,R5     ;RESTORE R5
4068 013156 012604          MOV     (SP)+,R4     ;RESTORE R4
4069 013160 000207          RTS     PC           ;RETURN
4070
```

```

4071          .SBTTL  DROP DRIVE AND GATHER STATISTICS ROUTINES
4072
4073 013162 013746 177776          DROP:  MOV    PS,-(SP)          ;STORE PSW
4074 013166 013737 001452 177776  MOV    RKPRI,PS          ;LOCK OUT TTY AND RK06 INTERRUPTS
4075 013174 005065 000014          CLR    P.PRST(R5)        ;CLEAR PROGRAM STAUUS REGISTER
4076 013200 005065 000212          CLR    P.ERR(R5)        ;CLEAR ERROR FLAGS
4077 013204 116537 000000 056755  MOVB   P.DRVN(R5),OPR011 ;LOAD DRIVE NUMBER
4078 013212 152737 000060 056755  BISB   #60,OPR011        ;MAKE IT ASCII
4079 013220 104401 056736          TYPE  ,OPR010           ;TYPE "DRIVE N HAS BEEN DROPPED
4080 013224 104401 056560          TYPE  ,OPR006           ;FROM TEST SEQUENCE"
4081 013230 132765 000001 000210  BITB   #BIT0,P.ASSN(R5) ;SEE IF SIZING FOR DRIVE
4082 013236 001404          BEQ    2$              ;BR IF NOT
4083 013240 105065 000210          CLRB  P.ASSN(R5)        ;CLEAR ASSINGMENT CODE BITS
4084 013244 000137 013530          JMP    STAT01
4085 013250 105065 000210 2$:   CLRB  P.ASSN(R5)        ;CLEAR ASSINGMENT CODE BITS
4086 013254 000137 013272          JMP    STAT00
4087
4088 013260 013746 177776          STAT:  MOV    PS,-(SP)          ;STORE PSW
4089 013264 013737 001452 177776  MOV    RKPRI,PS          ;LOCK OUT TTY AND RK06 INTERRUPTS
4090 013272 104401 057124          STAT00: TYPE ,DRVSTT          ;**DRIVE STATISTICS
4091 013276 116537 000000 056755  MOVB   P.DRVN(R5),OPR011 ;LOAD DRIVE NUMBER FOR PRINT OUT
4092 013304 152737 000060 056755  BISB   #60,OPR011        ;MAKE IT ASCII
4093 013312 104401 056736          TYPE  ,OPR010           ;TYPE "DRIVE N"
4094 013316 004737 027774          JSR    PC,PRISER        ;PRINT DRIVE AND PACK SERIAL NUMBERS
4095 013322 004737 044046          JSR    PC,PRITIM        ;PRINT TIME IF CLOCK AVAILABLE
4096 013326 104401 057202          TYPE  ,STT001           ;TYPE "ORDERS PERFORMED"
4097 013332 010546          MOV    R5,-(SP)         ;GET PARAMETER BLOCK ADDRESS
4098 013334 062716 000152          ADD    #P.NODR,(SP)     ;CALCULATE ADDRESS FOR PRINT OUT
4099 013340 004737 055470          JSR    PC,$DB2D         ;CONVERT TO ASCII
4100 013344 104401 057224          TYPE  ,STT002           ;TYPE "WORDS WRITTEN*65K"
4101 013350 010546          MOV    R5,-(SP)         ;GET PARAMETER BLOCK ADDRESS
4102 013352 062716 000160          ADD    #P.NWRT+2,(SP)  ;CALCULATE ADDRESS FOR PRINT OUT
4103 013356 004737 055470          JSR    PC,$DB2D         ;CONVERT TO ASCII
4104 013362 104401 057251          TYPE  ,STT003           ;TYPE "WORDS READ*65K"
4105 013366 010546          MOV    R5,-(SP)         ;GET PARAMETER BLOCK ADDRESS
4106 013370 062716 000166          ADD    #P.NRD+2,(SP)   ;CALCULATE ADDRESS FOR PRINT OUT
4107 013374 004737 055470          JSR    PC,$DB2D         ;CONVERT TO ASCII
4108 013400 104401 057273          TYPE  ,STT004           ;TYPE "SOFT DATA ERRORS"
4109          .ECC
4110 013404 016546 000176          MOV    P.SECC(R5),-(SP) ;STORE SOFT ECC ERROR COUNT
4111 013410 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL
4112 013414 104401 057326          TYPE  ,STT005           ;TYPE "REREAD"
4113 013420 016546 000172          MOV    P.SRRD(R5),-(SP) ;SAVE SOFT REREAD CORRECTABLE
4114 013424 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL
4115 013430 104401 057342          TYPE  ,STT006           ;TYPE "OFFSET"
4116 013434 016546 000174          MOV    P.SOFF(R5),-(SP) ;SAVE SOFT OFFSET CORRECTABLE
4117 013440 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL
4118 013444 104401 057356          TYPE  ,STT007           ;TYPE "HARD DATA ERRORS"
4119 013450 016546 000200          MOV    P.HARD(R5),-(SP) ;SAVE HARD DATA ERRORS
4120 013454 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL
4121 013460 104401 057402          TYPE  ,STT008           ;TYPE "SEEK INCOMPLETES"
4122 013464 016546 000202          MOV    P.NSKI(R5),-(SP) ;TYPE NUMBER OF SEEK INCOMPLETES
4123 013470 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL
4124 013474 104401 057426          TYPE  ,STT009           ;TYPE "OPERATION INCOMPLETES"
4125 013500 016546 000204          MOV    P.NOPI(R5),-(SP) ;SAVE NUMBER OF OPERATION INCOMPLETES
4126 013504 004737 055426          JSR    PC,$SB2D         ;CONVERT TO DECIMAL

```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 L 7 PAGE 92
DROP DRIVE AND GATHER STATISTICS ROUTINES

SEQ 0089

4127 013510 104401 057474
4128 013514 016546 000206
4129 013520 004737 055426
4130 013524 104401 001165
4131 013530 012637 177776
4132 013534 000207
4133

STAT01: TYPE ,STT010 :TYPE "OTHER ERRORS"
 MOV P.NER(R5),-(SP) :SAVE NUMBER OF OTHER ERRORS
 JSR PC,\$SB2D :CONVERT TO DECIMAL
 TYPE ,\$CRLF :TYPE <CR><LF>
 MOV (SP)+,PS :RESTORE PSW
 RTS PC :RETURN

```
4134  
4135  
4136 013536 116546 000210  
4137 013542 010446  
4138 013544 010346  
4139 013546 010503  
4140 013550 062703 000122  
4141  
4142 013554 010504  
4143 013556 062704 000232  
4144  
4145 013562 005023  
4146 013564 020304  
4147 013566 001375  
4148 013570 116504 000000  
4149 013574 156437 001514 001513  
4150 013602 012603  
4151 013604 012604  
4152 013606 112665 000210  
4153  
4154 013612 105065 000117  
4155 013616 005037 001500  
4156 013622 152765 000001 000210  
4157 013630 052765 000001 000014  
4158 013636 112765 000141 000001  
4159 013644 112765 000141 000123  
4160 013652 004037 051564  
4161 013656 001324  
4162 013660 000207
```

.SBTTL INITIATE DRIVE ASSIGNMENT SEQUENCE

TEST: MOV P.ASSN(R5),-(SP) :SAVE ASSIGNMENT CODE
 MOV R4,-(SP) :STORE R4 ON STACK
 MOV R3,-(SP) :STORE R3 ON STACK
 MOV R5,R3 :LOAD PARAMETER BLOCK ADDRESS INTO R3
 ADD #P.RDPT,R3 :CALCULATE BEGINNING OF STATISTICS AND
 : ERROR AREA
 MOV R5,R4 :LOAD PARAMETER BLOCK ADDRESS INTO R4
 ADD #P.EXAR,R4 :CALCULATE END OF STATISTICS AND
 : ERROR AREA
1\$: CLR (R3)+ :CLEAR STATISTIC AND ERROR AREA
 CMP R3,R4 :CHECK IF FINISHED
 BNE 1\$:NO, CONTINUE
 MOVB P.DRVN(R5),R4 :STORE DRIVE NUMBER
 BISB INTMSK(R4),O.OVER :SET UP FOR OVERLAPPING SEEKS
 MOV (SP)+,R3 :RESTORE R3
 MOV (SP)+,R4 :RESTORE R4
 MOVB (SP)+,P.ASSN(R5) :RESTORE ASSIGNMENT CODE

TEST1: CLRB P.ECMP(R5) :CLEAR ECC COMPARE FLAG
 CLR OPTFLG :CLEAR CMND OPTIMIZER FLAG
 BISB #BIT0,P.ASSN(R5) :SET READ STATUS COMMAND FLAG
 BIS #DRVUSE,P.PRST(R5) :SET DRIVE IN USE
 MOVB #RDSTAT,P.CMND(R5) :READ ALL DRIVE STATUS
 MOVB #RDSTAT,P.RCMD(R5)
 JSR R0,Q.PUSH :ENQUEUE PARAMETER BLOCK IN
 : COMMAND INITIATION QUEUE
 RTS PC :RETURN

```

4163      .SBTTL  DRIVE ASSIGNMENT SEQUENCE
4164
4165 013662 132765 000001 000210 ASNORM: BITB  #BIT0,P.ASSN(R5) ;CHECK IF READ STATUS
4166 013670 001467          BEQ    ASN4$      ;NO, CHECK IF START SPINDLE
4167 013672 005737 027334          TST    NEWFLG     ;SEE IF DTYE ERROR ON ASSIGNING
4168 013676 001404          BEQ    1$         ;BR IF NO
4169 013700 005037 027334          CLR    NEWFLG     ;ELSE CLEAR DTYE ON ASSIGN FLAG
4170 013704 000137 013612          JMP    TEST1      ;& REPEAT SELECT FOR DRIVE THAT HAD DTYE
4171 013710 142765 000001 000210 1$:  BICB  #BIT0,P.ASSN(R5) ;CLEAR READ STATUS COMMAND ISSUED
4172 013716 016565 000054 000224          MOV    P.A11(R5),P.SERL(R5) ;STORE SERIAL NUMBER
4173 013724 042765 100007 000224          BIC    #^C<M.SER>,P.SERL(R5) ;KEEP SERIAL NUMBER
4174 013732 006265 000224          ASR    P.SERL(R5)   ;SHIFT 3 BITS RIGHT
4175 013736 006265 000224          ASR    P.SERL(R5)
4176 013742 006265 000224          ASR    P.SERL(R5)
4177 013746 032765 000200 000036          BIT    #DRDY,P.DS(R5) ;CHECK IF DRIVE READY
4178 013754 001415          BEQ    ASN2$      ;NO, ISSUE START SPINDLE
4179 013756 116565 000123 000135 ASN1$: MOVB  P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
4180 013764 112765 000103 000001          MOVB  #PACK,P.CMND(R5) ;ISSUE PACK ACKNOWLEDGE
4181 013772 112765 000103 000123          MOVB  #PACK,P.RCMD(R5)
4182 014000 152765 000004 000210          BISB  #BIT2,P.ASSN(R5) ;SET PACK ACKNOWLEDGE FLAG
4183 014006 000414          BR    ASN3$      ;ISSUE PACK ACKNOWLEDGE AND WAIT
4184
4185 014010 116565 000123 000135 ASN2$: MOVB  P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
4186 014016 112765 000111 000001          MOVB  #SRTSPL,P.CMND(R5) ;ISSUE START SPINDLE
4187 014024 112765 000111 000123          MOVB  #SRTSPL,P.RCMD(R5)
4188 014032 152765 000002 000210          BISB  #BIT1,P.ASSN(R5) ;SET START SPINDLE FLAG
4189 014040 004037 051564          JSR    R0,Q.PUSH  ;ENQUEUE PARAMETER BLOCK IN
4190 014044 001324          CINITQ ; COMMAND INITIATION QUEUE
4191 014046 000207          RTS    PC        ;RETURN
4192
4193 014050 132765 000002 000210 ASN4$: BITB  #BIT1,P.ASSN(R5) ;CHECK IF START SPINDLE
4194 014056 001414          BEQ    15$       ;NO, CHECK IF PACK ACKNOWLEDGE
4195 014060 142765 000002 000210          BICB  #BIT1,P.ASSN(R5) ;CLEAR START SPINDLE FLAG
4196 014066 032765 000200 000036          BIT    #DRDY,P.DS(R5) ;CHECK IF DRIVE READY
4197 014074 001330          BNE    ASN1$     ;YES, ISSUE PACK ACKNOWLEDGE
4198 014076 052765 002000 000212          BIS    #BIT10,P.ERR(R5) ;SET DRIVE NOT READY AFTER START SPINDLE
4199 014104 000137 025470          JMP    ABNORM    ;REPORT ERROR
4200
4201 014110 132765 000004 000210 15$:  BITB  #BIT2,P.ASSN(R5) ;CHECK IF PACK ACKNOWLEDGE
4202 014116 001431          BEQ    18$       ;NO, CHECK IF RECAL
4203 014120 142765 000004 000210          BICB  #BIT2,P.ASSN(R5) ;CLEAR PACK ACKNOWLEDGE FLAG
4204 014126 032765 000100 000036          BIT    #VV,P.DS(R5)   ;CHECK IF VOLUME VALID IS SET
4205 014134 001005          BNE    16$       ;YES, CHECK IF PACK IS TO BE WRITTEN
4206 014136 052765 004000 000212          BIS    #BIT11,P.ERR(R5) ;SET VOLUME VALID NOT SET AFTER PACK
4207          ; ACKNOWLEDGE
4208 014144 000137 025470          JMP    ABNORM    ;REPORT ERROR
4209
4210 014150 116565 000123 000135 16$:  MOVB  P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
4211 014156 112765 000113 000001          MOVB  #RECAL,P.CMND(R5) ;ISSUE RECAL
4212 014164 112765 000113 000123          MOVB  #RECAL,P.RCMD(R5)
4213 014172 152765 000100 000210          BISB  #BIT6,P.ASSN(R5) ;SET RECAL ISSUED
4214 014200 000717          BR    ASN3$
4215
4216 014202 132765 000100 000210 18$:  BITB  #BIT6,P.ASSN(R5) ;CHECK IF RECAL
4217 014210 001457          BEQ    23$       ;NO, CHECK IF READ PACK SERIAL NUMBER
4218 014212 142765 000100 000210          BICB  #BIT6,P.ASSN(R5) ;RESET RECAL ISSUED

```

4219	014220	152765	000010	000210	BISB	#BIT3,P.ASSN(R5)	;SET ASSIGNMENT CODE FOR PACK
4220							; SERIAL NUMBER READ
4221	014226	012765	001000	000004	MOV	#1000,P.SECT(R5)	; TRACK 2 SECTOR 0
4222	014234	116565	000123	000135	MOVB	P.RCMD(R5),P.LCMD(R5)	;STORE PREVIOUS COMMAND
4223	014242	112765	000121	000001	20\$: MOVB	#RDDATA,P.CMND(R5)	;LOAD READ COMMAND FOR PACK
4224	014250	112765	000121	000123	MOVB	#RDDATA,P.RCMD(R5)	; SERIAL NUMBER
4225	014256	012765	177774	000012	MOV	#-4,P.WC(R5)	;LOAD WORD COUNT FOR THE FIRST FOUR WORDS
4226	014264	004737	027252		JSR	PC,GETNUM	;GET CORRECT END CYL NUM
4227	014270	013765	027332	000002	MOV	LSTCYL,P.CYLN(R5)	;SET CYLINDER TO 632/1456
4228	014276	013765	027332	000124	MOV	LSTCYL,P.RCYL(R5)	
4229	014304	016565	000004	000126	MOV	P.SECT(R5),P.RSEC(R5)	;STORE TRACK AND SECTOR
4230	014312	005737	001320		TST	BWAITQ	;CHECK IF BUFFER WAIT QUEUE IS EMPTY
4231	014316	001010			BNE	22\$;NO, ENQUEUE PARAMETER BLOCK IN BUFFER
4232							; WAIT QUEUE
4233	014320	004037	022616		JSR	RO,BUFFAL	;ALLOCATE BUFFER
4234	014324	014340			22\$;NOT ENOUGH ROOM RETURN
4235	014326	005037	177776		CLR	PS	;ALLOW RK06 INTERRUPTS
4236	014332	004737	023234		JSR	PC,LDDATA	;CLEAR BUFFER
4237	014336	000640			BR	ASN3\$;ISSUE COMMAND
4238							
4239	014340	004037	051564		22\$: JSR	RO,Q.PUSH	;PUT PARAMETER BLOCK ON BUFFER
4240	014344	001320			BWAITQ		; WAIT QUEUE
4241	014346	000207			RTS	PC	;RETURN
4242							
4243	014350	032777	040000	164562	23\$: BIT	#SW14,@SWR	;CHECK IS CYCLE ON OPERATION
4244	014356	001542			BEQ	43\$;NO, CONTINUE WITH ASSIGNMENT PROCESS
4245	014360	105765	000211		TSTB	P.SEEK(R5)	;CHECK IF SEEK TO PREVIOUS POSITION
4246	014364	001444			BEQ	28\$;NO, SERVICE INTERRUPT
4247	014366	105065	000211		CLRB	P.SEEK(R5)	;CLEAR FLAG
4248	014372	116565	000123	000001	MOVB	P.RCMD(R5),P.CMND(R5)	;REISSUE LAST COMMAND
4249	014400	016565	000124	000002	MOV	P.RCYL(R5),P.CYLN(R5)	;LOAD CYLINDER ADDRESS
4250	014406	016565	000126	000004	MOV	P.RSEC(R5),P.SECT(R5)	;LOAD PREVIOUS TRACK AND SECTOR
4251	014414	016565	000130	000012	MOV	P.RWC(R5),P.WC(R5)	;LOAD PREVIOUS WORD COUNT
4252	014422	116565	000122	000121	MOVB	P.RDPT(R5),P.DPAT(R5)	;LOAD PREVIOUS PATTERN
4253	014430	132765	000010	000210	BITB	#BIT3,P.ASSN(R5)	;CHECK IF READ PACK SERIAL NUMBER
4254	014436	001003			BNE	25\$;YES, DO NOT SET WRITE BEFORE WRITE CHECK
4255	014440	052765	000200	000014	BIS	#W.WCK,P.PRST(R5)	;SET WRITE BEFORE WRITE CHECK
4256	014446	005737	001320		25\$: TST	BWAITQ	;CHECK IF ANY DRIVE WAITING FOR BUFFER
4257	014452	001332			BNE	22\$;YES, ENQUEUE PARAMETER BLOCK
4258	014454	004037	022616		JSR	RO,BUFFAL	;GO ALLOCATE BUFFER
4259	014460	014340			22\$;NOT ENOUGH ROOM RETURN
4260	014462	005037	177776		CLR	PS	;ALLOW RK06 INTERRUPTS
4261	014466	004737	023234		JSR	PC,LDDATA	;GO LOAD DATA
4262	014472	000137	014040		JMP	ASN3\$;GO ISSUE COMMAND
4263							
4264	014476	004037	025210		28\$: JSR	RO,CHKADD	;CHECK SECTOR, TRACK, CYLINDER,
4265							; WORD COUNT, AND BUS ADDRESS
4266	014502	015410			60\$;ERROR RETURN
4267	014504	132765	000010	000210	BITB	#BIT3,P.ASSN(R5)	;CHECK IF READ PACK SERIAL NUMBER
4268	014512	001424			BEQ	35\$;NO, RELEASE BUFFER
4269	014514	010446			MOV	R4,-(SP)	;STORE R4 ON STACK
4270	014516	016504	000010		MOV	P.BALO(R5),R4	;LOAD R4 TO GET PACK SERIAL NUMBER
4271	014522	012465	000226		MOV	(R4)+,P.PKSR(R5)	;GET PACK SERIAL NUMBER
4272	014526	012465	000230		MOV	(R4)+,P.PKSR+2(R5)	
4273	014532	005724			TST	(R4)+	;ADJUST R4 TO ADDRESS ALLIGNMENT INDICATION
4274	014534	005714			TST	(R4)	;CHECK IF DATA PACK

4275	014536	001423				BEQ	37\$:YES, CONTINUE
4276	014540	012604			30\$:	MOV	(SP)+,R4	:RESTORE R4
4277	014542	004737	023020			JSR	PC,BUFREL	:RELEASE BUFFER
4278	014546	104401	057071			TYPE	,OPR015	:TYPE 'ALLIGNMENT PACT IN DRIVE''
4279	014552	004737	013162			JSR	PC,DROP	:GO DROP DRIVE
4280	014556	004737	022520			JSR	PC,GETBUF	:GO GET BUFFER FOR NEXT COMMAND
4281	014562	000207				RTS	PC	:RETURN
4282								
4283	014564	032765	000200	000014	35\$:	BIT	#W,WCK,P.PRST(R5)	:CHECK IF WRITE CHECK HAS BEEN ISSUED
4284	014572	001406				BEQ	38\$:YES, GO RELEASE BUFFER
4285	014574	042765	000200	000014		BIC	#W,WCK,P.PRST(R5)	:CLEAR WRITE BEFORE WRITE CHECK
4286	014602	000137	014040			JMP	ASN3\$:GO ISSUE COMMAND
4287								
4288	014606	012604			37\$:	MOV	(SP)+,R4	:RESTORE R4
4289	014610	004737	023020		38\$:	JSR	PC,BUFREL	:RELEASE BUFFER
4290	014614	032765	002000	000014		BIT	#DRPDRV,P.PRST(R5)	:CHECK IF DROP DRIVE
4291	014622	001402				BEQ	40\$:NO, SEEK TO PREVIOUS POSITION
4292	014624	000137	016216			JMP	WV1\$:GO DROP DRIVE
4293								
4294	014630	112765	177777	000211	40\$:	MOVB	#-1,P.SEEK(R5)	:SET FLAG
4295	014636	016565	000136	000002		MOV	P.LCYL(R5),P.CYLN(R5)	:LOAD LAST CYLINDER
4296	014644	016565	000140	000004		MOV	P.LSEC(R5),P.SECT(R5)	:LOAD LAST SECTOR AND TRACK
4297	014652	112765	000117	000001		MOVB	#SEEK,P.CMND(R5)	:ISSUE SEEK
4298	014660	000137	014040			JMP	ASN3\$:GO SEEK TO PREVIOUS POSITION
4299								
4300	014664	132765	000010	000210	43\$:	BITB	#BIT3,P.ASSN(R5)	:CHECK IF READ PACK SERIAL NUMBER
4301	014672	001002				BNE	45\$:YES, SET UP FOR READ
4302	014674	000137	015412			JMP	WRTVER	:CHECK IF WRITE PACK
4303								
4304	014700	132765	000200	000210	45\$:	BITB	#BIT7,P.ASSN(R5)	:CHECK IF RECAL AND RETRY
4305								: READ OF PACK SERIAL NUMBER
4306	014706	001427				BEQ	48\$:NO, SERVICE READ OF PACK SERIAL NUMBER
4307	014710	142765	000200	000210		BICB	#BIT7,P.ASSN(R5)	:RESET RECAL FOR REREAD OF PACK SERIAL NO.
4308	014716	116565	000123	000135		MOVB	P.RCMD(R5),P.LCMD(R5)	:STORE LAST COMMAND
4309	014724	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	:STORE LAST CYLINDER
4310	014732	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	:STORE LAST SECTOR AND TRACK
4311	014740	016565	000130	000142		MOV	P.RWC(R5),P.LWC(R5)	:STORE LAST WORD COUNT
4312	014746	062765	000002	000126		ADD	#2,P.RSEC(R5)	:TRY NEXT 16-BIT FORMAT SECTOR
4313	014754	016565	000126	000004		MOV	P.RSEC(R5),P.SECT(R5)	:STORE SECTOR TO BE READ
4314	014762	000137	014242			JMP	20\$:GO ISSUE READ OF PACK SERIAL NUMBER
4315								
4316	014766	142765	000010	000210	48\$:	BICB	#BIT3,P.ASSN(R5)	:CLEAR FLAG
4317	014774	004037	025210			JSR	R0,CHKADD	:CHECK IF DATA TRANSFER CORRECT
4318	015000	015410				60\$:ERROR RETURN
4319	015002	010446				MOV	R4,-(SP)	:STORE R4
4320	015004	016504	000132			MOV	P.RBAL(R5),R4	:LOAD R4 TO ADDRESS SERIAL NUMBER
4321	015010	012465	000226			MOV	(R4)+,P.PKSR(R5)	:LOAD LEAST SIGNIFICANT BITS
4322								: OF SERIAL NUMBER
4323	015014	012465	000230			MOV	(R4)+,P.PKSR+2(R5)	:LOAD MOST SIGNIFICANT BITS
4324								: OF SERIAL NUMBER
4325	015020	005724				TST	(R4)+	:ADJUST R4 TO ADDRESS ALLIGNMENT INDICATION
4326	015022	005714				TST	(R4)	:CHECK IF ALLIGNMENT PACK
4327	015024	001245				BNE	30\$:YES, DROP DRIVE
4328	015026	012604				MOV	(SP)+,R4	:RESTORE R4
4329	015030	004737	023020			JSR	PC,BUFREL	:RELEASE BUFFER
4330	015034	132765	000020	000210		BITB	#BIT4,P.ASSN(R5)	:CHECK IF WRITE PACK

```

4331 015042 001042          BNE      55$      ;YES, SET UP PATTERN AND ALLOCATE BUFFER
4332 015044 105765 000076    TSTB     P.RATE(R5) ;CHECK IF READ ONLY MODE
4333 015050 001413          BEQ      52$      ;YES, START TESTING
4334 015052 032765 004000 000036 BIT      #WRL,P.DS(R5) ;CHECK IF WRITE LOCKED
4335 015060 001407          BEQ      52$      ;NO, START TESTING
4336
4337 015062 104401 056760    50$:     TYPE     ,OPR012 ;TYPE 'DRIVE WRITE LOCKED'
4338 015066 004737 013162    JSR     PC,DROP   ;DROP DRIVE FROM TEST SEQUENCE
4339 015072 004737 022520    JSR     PC,GETBUF ;GET BUFFER FOR NEXT COMMAND
4340 015076 000207          RTS      PC       ;RETURN
4341
4342 015100 116537 000000 056755 52$:     MOVB     P.DRVN(R5),OPR011 ;LOAD DRIVE NUMBER
4343 015106 152737 000060 056755    BISB     #60,OPR011 ;MAKE IT ASCII
4344 015114 104401 056736    TYPE     ,OPR010 ;TYPE 'DRIVE NUMBER N UNDER TEST'
4345 015120 104401 056501    TYPE     ,OPR003
4346 015124 004737 027774    JSR     PC,PRISER ;PRINT DRIVE AND PACK SERIAL NUMBERS
4347 015130 004737 044046    JSR     PC,PRITIM ;PRINT TIME
4348 015134 104401 001165    TYPE     ,$CRLF
4349 015140 004037 051564    JSR     RO,Q.PUSH ;PUT PARAMETER BLOCK IN DRIVE
4350 015144 001314          AVAILQ   ; AVAILIABLE QUEUE
4351 015146 000207          RTS      PC       ;RETURN
4352
4353          .SBTTL  WRITE WHOLE PACK WITH RANDOM DATA
4354
4355 015150 032765 004000 000036 55$:     BIT      #WRL,P.DS(R5) ;CHECK IF WRITE LOCKED
4356 015156 001341          BNE     50$      ;YES, DROP DRIVE
4357 015160 005037 177776    CLR     PS       ;ALLOW RK06 INTERRUPTS
4358 015164 116565 000123 000135 MOVB     P.RCMD(R5),P.LCMD(R5) ;STORE LAST COMMAND
4359 015172 016565 000124 000136 MOV      P.RCYL(R5),P.LCYL(R5) ;STORE LAST CYLINDER
4360 015200 016565 000126 000140 MOV      P.RSEC(R5),P.LSEC(R5) ;STORE LAST TRACK AND SECTOR
4361 015206 016565 000130 000142 MOV      P.RWC(R5),P.LWC(R5) ;STORE LAST WORD COUNT
4362 015214 112765 000131 000001 MOVB     #WRTCHK,P.CMND(R5) ;LOAD WRITE COMMAND
4363 015222 112765 000131 000123 MOVB     #WRTCHK,P.RCMD(R5)
4364 015230 052765 000200 000014 BIS      #W.WCK,P.PRST(R5) ;SET WRITE BEFORE WRITE CHECK
4365 015236 005065 000002    CLR     P.CYLN(R5) ;CLEAR CYLINDER ADDRESS
4366 015242 005065 000124    CLR     P.RCYL(R5)
4367 015246 005065 000004    CLR     P.SECT(R5) ;CLEAR TRACK AND SECTOR
4368 015252 005065 000126    CLR     P.RSEC(R5)
4369 015256 105065 000121    CLRB   P.DPAT(R5) ;LOAD RANDOM PATTERN
4370 015262 013765 001354 000012 MOV      MAXBUF,P.WC(R5) ;LOAD WORD COUNT
4371 015270 042765 000377 000012 BIC     #377,P.WC(R5) ;MAKE IT AN EVEN NUMBER OF SECTORS
4372 015276 001003          BNE     57$      ;IF NOT ZERO USE THIS AS WORD COUNT
4373 015300 012765 000400 000012 MOV      #256,P.WC(R5) ;TRANSFER AT LEAST 256 WORDS
4374 015306 062765 000002 000152 57$:     ADD     #2,P.NODR(R5) ;INCREMENT NUMBER OF ORDERS
4375 015314 066565 000012 000156 ADD     P.WC(R5),P.NWRT(R5) ;ADD NUMBER OF WORDS WRITTEN
4376 015322 005565 000160    ADC     P.NWRT+2(R5)
4377 015326 066565 000012 000164 ADD     P.WC(R5),P.NRD(R5) ;ADD NUMBER OF WORDS READ
4378 015334 005565 000166    ADC     P.NRD+2(R5)
4379 015340 005465 000012    NEG     P.WC(R5) ;MAKE THE WORD COUNT NEGITIVE
4380 015344 013737 001452 177776 MOV      RKPRI,PS ;LOCK RK06 INTERRUPTS
4381 015352 005737 001320    TST     BWAITQ   ;CHECK IF COMMANDS WAITING FOR BUFFER
4382 015356 001011          BNE     59$      ;YES, ENQUEUE COMMAND IN BUFFER WAIT QUEUE
4383 015360 004037 022616    JSR     RO,BUFFAL ;GO ALLOCATE BUFFER
4384 015364 015402          59$      ;NOT ENOUGH ROOM, ENQUEUE IN BUFFER WAIT QUEUE
4385 015366 005037 177776    CLR     PS       ;ALLOW RK06 INTERRUPTS
4386 015372 004737 023234    JSR     PC,LDDATA ;LOAD BUFFER

```


CZR6PD0 RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 PAGE 98
WRITE WHOLE PACK WITH RANDOM DATA

SEQ 0095

4387	015376	000137	014040		JMP	ASN3\$:GO ISSUE COMMAND
4388								
4389	015402	004037	051564		59\$: JSR	R0,Q.PUSH		:PUT PARAMETER BLOCK IN BUFFER
4390	015406	001320			BWAITQ			: WAIT QUEUE
4391	015410	000207			60\$: RTS	PC		:RETURN
4392								
4393	015412	132765	000020	000210	WRTVER: BITB	#BIT4,P.ASSN(R5)		:CHECK IF WRITE PACK IN PROGRESS
4394	015420	001001			BNE	1\$:YES, CONTINUE
4395	015422	000000			HALT			: **** PROGRAM PROBLEM
4396	015424	032765	000200	000014	1\$: BIT	#W.WCK,P.PRST(R5)		:CHECK IF WRITE CHECK ISSUED

4397	015432	001410				BEQ	WV0\$:YES, CHECK FOR DROP DRIVE
4398	015434	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:CLEAR ISSUE WRITE BEFORE WRITE CHECK
4399	015442	112765	000131	000001		MOVB	#WRTCHK,P.CMND(R5)	:RESTORE WRITE CHK CMD
4400	015450	000137	014040			JMP	ASN3\$:GO ISSUE WRITE CHECK
4401								
4402	015454	032765	002000	000014	WV0\$:	BIT	#DRPDRV,P.PRST(R5)	:CHECK IF DROP DRIVE
4403	015462	001402				BEQ	10\$:NO, CONTINUE
4404	015464	000137	016206			JMP	30\$:YES, TERMINATE PACK WRITING
4405								
4406	015470	132765	000040	000210	10\$:	BITB	#BIT5,P.ASSN(R5)	:CHECK IF END OF PACK
4407	015476	001402				BEQ	15\$:NO, CONTINUE
4408	015500	000137	016230			JMP	WV2\$:YES, RETURN BUFFER
4409	015504	052765	000200	000014	15\$:	BIS	#W.WCK,P.PRST(R5)	:SET ISSUE WRITE BEFORE WRT CHK
4410	015512	122765	000140	000001		CMPB	#RELEAS,P.CMND(R5)	:SEE IF LAST CMD WAS RELEASE
4411	015520	001452				BEQ	3\$:BR IF YES, ELSE FILL PARAM BLOCK
4412	015522	112737	177777	001506		MOVB	#-1,I.ISRL	:INDICATE THAT AN INTERRUPT HAS BEEN ISSUED
4413	015530	012762	000300	000000		MOV	#INTR,RKCS1(R2)	:GO SCAN FOR DRIVE INTERRUPTS
4414	015536	005037	177776			CLR	PS	:ALLOW RK06 INTERRUPTS
4415	015542	116565	000123	000135		MOVB	P.RCMD(R5),P.LCMD(R5)	:STORE PREVIOUS COMMAND
4416	015550	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	:STORE PREVIOUS CYLINDER
4417	015556	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	:STORE PREVIOUS TRACK AND SECTOR
4418	015564	116565	000122	000144		MOVB	P.RDPT(R5),P.LDPT(R5)	:STORE PREVIOUS DATA PATTERN
4419	015572	016565	000130	000142		MOV	P.RWC(R5),P.LWC(R5)	:STORE PREVIOUS WORD COUNT
4420	015600	016565	000026	000004		MOV	P.DTS(R5),P.SECT(R5)	:LOAD NEW SECTOR AND TRACK
4421	015606	016565	000026	000126		MOV	P.DTS(R5),P.RSEC(R5)	
4422	015614	016565	000030	000002		MOV	P.DCYL(R5),P.CYLN(R5)	:LOAD NEW CYLINDER
4423	015622	016565	000030	000124		MOV	P.DCYL(R5),P.RCYL(R5)	
4424	015630	016565	000132	000010		MOV	P.RBAL(R5),P.BALO(R5)	:REINITIALIZE BUS ADDRESS FOR POSSIBLE
4425								:BAD SECTOR RECOVERY ON LAST COMMAND
4426	015636	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:SEE IF LAST CMD WAS WRT CHK
4427	015644	001404				BEQ	4\$:BR IF YES
4428	015646	112765	000131	000001	3\$:	MOVB	#WRTCHK,P.CMND(R5)	:ELSE SET TO DO WRT DATA (W.WCK=1)
4429	015654	000410				BR	5\$	
4430	015656	042765	000200	000014	4\$:	BIC	#W.WCK,P.PRST(R5)	
4431	015664	112765	000140	000001		MOVB	#RELEAS,P.CMND(R5)	:RELEAS AFTER WRT CHK
4432	015672	000137	014040			JMP	ASN3\$:GO RELEASE
4433	015676	016546	000002		5\$:	MOV	P.CYLN(R5),-(SP)	:STORE CYLINDER ADDRESS ON STACK
4434	015702	006316				ASL	(SP)	:MULTIPLY CYLINDER BY 3
4435	015704	066516	000002			ADD	P.CYLN(R5),(SP)	
4436	015710	005046				CLR	-(SP)	:MAKE ROOM ON STACK FOR TRACK
4437	015712	116516	000005			MOVB	P.TRCK(R5),(SP)	:STORE TRACK ON STACK
4438	015716	061666	000002			ADD	(SP),2(SP)	:ADD TRACK
4439	015722	012716	000026			MOV	#22,(SP)	:STORE 22 ON STACK (MULTIPLIER)
4440	015726	004737	055210			JSR	PC,\$MULT	:CALCULATE NUMBER OF SECTORS TRANSFERRED
4441	015732	012616				MOV	(SP)+,(SP)	:THROW AWAY MOST SIGNIFICANT BITS
4442	015734	005046				CLR	-(SP)	:CLEAR LOCATION FOR SECTOR
4443	015736	116516	000004			MOVB	P.SECT(R5),(SP)	:STORE SECTOR COUNT
4444	015742	061666	000002			ADD	(SP),2(SP)	:DETERMINE TOTAL NUMBER OF SECTORS TRANSFERRED
4445	015746	004737	027252			JSR	PC,GETNUM	:GET CORRECT LAST CYL NUM
4446	015752	013716	027336			MOV	HOLD,(SP)	:STORE NUMBER OF SECTORS ON STACK
4447	015756	166616	000002			SUB	2(SP),(SP)	:DETERMINE NUMBER OF SECTORS LEFT ON
4448								:ON PACK TO BE WRITTEN
4449	015762	016546	000130			MOV	P.RWC(R5),-(SP)	:STORE WORD COUNT
4450	015766	011665	000012			MOV	(SP),P.WC(R5)	
4451	015772	005416				NEG	(SP)	:MAKE IT POSITIVE
4452	015774	116616	000001			MOVB	1(SP),(SP)	:KEEP SECTOR COUNT

4453	016000	105066	000001		CLRB	1(SP)	:CLEAR MOST SIGNIFICANT BITS
4454	016004	021666	000002		CMP	(SP),2(SP)	:CHECK IF PACK OVERRUN
4455	016010	103413			BLO	20\$:NO, GO ISSUE COMMAND
4456	016012	001407			BEQ	19\$:CHECK IF NO NEW WORD COUNT NEEDED
4457	016014	005016			CLR	(SP)	:MAKE ROOM FOR NEW WORD COUNT
4458	016016	116666	000002	000001	MOVB	2(SP),1(SP)	:MOVE IN NUMBER OF SECTORS LEFT
4459	016024	005416			NEG	(SP)	:MAKE WORD COUNT NEGATIVE
4460	016026	011665	000012		MOV	(SP),P.WC(R5)	:LOAD NEW WORD COUNT
4461	016032	152765	000040	000210	BISB	#BIT5,P.ASSN(R5)	:SET END OF PACK TRANSFER
4462							
4463	016040	062706	000006		ADD	#6,SP	:ADJUST STACK
4464	016044	016546	000012		MOV	P.WC(R5),-(SP)	:STORE PRESENT WORD COUNT
4465	016050	005416			NEG	(SP)	:MAKE IT POSITIVE
4466	016052	062765	000002	000152	ADD	#2,P.NODR(R5)	:INCREMENT NUMBER OF ORDERS
4467	016060	061665	000156		ADD	(SP),P.NWRT(R5)	:ADD NUMBER OF WORDS WRITTEN
4468	016064	005565	000160		ADC	P.NWRT+2(R5)	
4469	016070	062665	000164		ADD	(SP)+,P.NRD(R5)	:ADD NUMBER OF WORDS READ
4470	016074	005565	000166		ADC	P.NRD+2(R5)	
4471	016100	105265	000122		INCB	P.RDPT(R5)	:GENERATE NEXT DATA PATTERN TO BE WRITTEN
4472	016104	142765	000360	000122	BICB	#360,P.RDPT(R5)	:KEEP 4 LEAST SIGNIFICANT BITS
4473	016112	116565	000122	000121	MOVB	P.RDPT(R5),P.DPAT(R5)	:STORE DATA PATTERN USED
4474	016120	013737	001452	177776	MOV	RKPRI,PS	:LOCK OUT RK06 INTERRUPTS
4475	016126	132765	000040	000210	BITB	#BIT5,P.ASSN(R5)	:CHECK IF WRITE TO END OF PACK
4476	016134	001014			BNE	25\$:YES, RELEASE BUFFER
4477	016136	005737	001320		TST	BWAITQ	:CHECK IF BUFFER WAIT QUEUE EMPTY
4478	016142	001011			BNE	25\$:NO, RELEASE BUFFER
4479	016144	016565	000012	000130	MOV	P.WC(R5),P.RWC(R5)	:STORE CURRENT GENERATED WORD COUNT
4480	016152	005037	177776		CLR	PS	:ALLOW RK06 INTERRUPTS
4481	016156	004737	023234		JSR	PC,LDDATA	:LOAD NEW RANDOM DATA
4482	016162	000137	014040		JMP	ASN3\$:GO ISSUE COMMAND
4483							
4484	016166	004737	023020		JSR	PC,BUFREL	:RELEASE BUFFER FOR OTHER DRIVES
4485	016172	004037	051564		JSR	R0,Q.PUSH	:PUT PARAMETER BLOCK ON BUFFER WAIT QUEUE
4486	016176	001320			BWAITQ		
4487	016200	004737	022520		JSR	PC,GETBUF	:GET NEW BUFFER
4488	016204	000207			RTS	PC	:RETURN
4489							
4490	016206	105065	000210		30\$: CLRB	P.ASSN(R5)	:CLEAR ASSIGNMENT CODE
4491	016212	004737	023020		JSR	PC,BUFREL	:RELEASE BUFFER
4492	016216	104401	056656		WV1\$: TYPE	,OPR08	:TYPE 'OPERATOR INITIATED DROP DRIVE'
4493	016222	004737	013162		JSR	PC,DROP	:GO DROP DRIVE
4494	016226	000444			BR	WV5\$:GO ALLOCATE BUFFER
4495							
4496	016230	105065	000210		WV2\$: CLRB	P.ASSN(R5)	:CLEAR ASSIGNMENT FLAGS
4497	016234	116537	000000	056755	MOVB	P.DRVN(R5),OPR011	:GET DRIVE NUMBER
4498	016242	152737	000060	056755	BISB	#60,OPR011	:MAKE IT ASCII
4499	016250	104401	056736		TYPE	,OPR010	:TYPE 'DRIVE NUMBER N UNDER TEST'
4500	016254	104401	056501		TYPE	,OPR003	
4501	016260	004737	027774		JSR	PC,PRISER	:PRINT DRIVE AND PACK SERIAL NUMBERS
4502	016264	004737	044046		JSR	PC,PRITIM	:PRINT TIME
4503	016270	104401	001165		TYPE	,\$CRLF	
4504	016274	010346			MOV	R3,-(SP)	:STORE R3 ON STACK
4505	016276	010446			MOV	R4,-(SP)	:STORE R4 ON STACK
4506	016300	010503			MOV	R5,R3	:LOAD PARAMETER BLOCK BASE
4507	016302	062703	000152		ADD	#P.NODR,R3	:CALCULATE BEGINNING OF STATISTICS
4508	016306	010504			MOV	R5,R4	:LOAD PARAMETER BLOCK BASE

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 H 8 PAGE 101
WRITE WHOLE PACK WITH RANDOM DATA

SEQ 0098

4509 016310 062704 000210
4510 016314 005023
4511 016316 020304
4512 016320 001375
4513 016322 012604
4514 016324 012603
4515 016326 004737 023020
4516 016332 004037 051564
4517 016336 001314
4518 016340 004737 022520
4519 016344 000207

7\$: ADD #P,ASSN,R4 ;CALCULATE END OF STATISTICS
CLR (R3)+ ;CLEAR STATISTICS BEFORE STARTING TEST
CMP R3,R4 ;CHECK IF FINISHED
BNE 7\$;NO, CONTINUE
MOV (SP)+,R4 ;RESTORE R4
MOV (SP)+,R3 ;RESTORE R3
JSR PC,BUFREL ;RELEASE BUFFER
JSR R0,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN DRIVE
AVAILQ ; AVAILIABLE QUEUE
WV5\$: JSR PC,GETBUF ;GET NEW BUFFER
WV9\$: RTS PC ;RETURN

```

4520          .SBTTL  ALTER DRIVE PARAMETERS
4521
4522 016346 010446          PARM:  MOV    R4,-(SP)      ;STORE R4 ON STACK
4523 016350 010346          MOV    R3,-(SP)      ;STORE R3 ON STACK
4524 016352 116537 000000 060131 POS:  MOVVB  P.DRVN(R5),PARC01 ;LOAD DRIVE NUMBER FOR PRINT OUT
4525 016360 152737 000060 060131      BISB   #60,PAR001    ;MAKE IT ASCII
4526 016366 104401 060105          TYPE  ,PAR000      ;TYPE 'PARAMETERS FOR DRIVE N'
4527 016372 004737 020566          JSR   PC,DTYPE     ;INPUT DRIVE TYPE
4528 016376 104401 060136          2$:   TYPE  ,PAR002      ;TYPE 'CYLINDER MAX,MIN'
4529 016402 004037 020516          JSR   R0,TSTDEF   ;CHECK FOR DEFAULT PARAMETERS
4530 016406 016524          5$      ;COMMA DETECTED
4531 016410 016540          6$      ;CARRIAGE RETURN DETECTED
4532 016412 020356          P1$     ;CONTROL Z <^Z> DETECTED
4533 016414 016352          POS     ;CONTROL C <^C> DETECTED
4534 016416 010346          MOV    R3,-(SP)      ;STORE BUFFER ADDRESS
4535 016420 004737 054406          JSR   PC,OCTBIN    ;CONVERT TO BINARY
4536 016424 016506          4$      ;ERROR RETURN
4537 016426 012637 001602          MOV   (SP)+,PARCYL  ;STORE MAX CYLINDER ENTRY
4538 016432 004737 027252          JSR   PC,GETNUM    ;GET CORRECT LAST CYL NUM
4539 016436 023737 027332 001602          CMP   LSTCYL,PARCYL ;CHECK IF LESS OR EQUAL TO
4540          ; MAXIMUM CYLINDER ADDRESS
4541 016444 103420          BLO   4$          ;NO, TRY AGAIN
4542 016446 010304          MOV   R3,R4        ;STORE R3 IN R4 FOR LINE SCAN
4543 016450 105714          64$:  TSTB  (R4)        ;CHECK FOR CARRIAGE RETURN
4544 016452 001435          BEQ   7$          ;YES, EXIT
4545 016454 122427 000054          CMPB  (R4)+,#' ,   ;CHECK FOR COMMA
4546 016460 001373          BNE   64$        ;NO, GO TEST NEXT CHARACTER
4547 016462 010446          3$:   MOV   R4,-(SP)      ;STORE BUFFER ADDRESS
4548 016464 004737 054406          JSR   PC,OCTBIN    ;CONVERT TO BINARY
4549 016470 016506          4$      ;ERROR RETURN
4550 016472 012637 001604          MOV   (SP)+,PARCYL+2 ;STORE MIN CYLINDER ENTRY
4551 016476 023737 001602 001604          CMP   PARCYL,PARCYL+2 ;CHECK IF LEGAL ENTRY
4552 016504 103022          BHIS  8$          ;YES, LOAD MAX AND MIN CYLINDER
4553 016506          4$:
4554 016506 010337 016514          MOV   R3,65$      ;LOAD BUFFER ADDRESS
4555 016512 104401          TYPE  ;TYPE RECEIVED INPUT
4556 016514 000000          65$:  .WORD  0          ;BUFFER ADDRESS
4557 016516 104401 001164          TYPE  ,SQUES      ;TYPE QUESTION MARK
4558 016522 000725          BR    2$          ;TRY AGAIN
4559
4560 016524 013737 027332 001602 5$:   MOV   LSTCYL,PARCYL ;LOAD MAX CYLINDER = 632/1456
4561 016532 010304          MOV   R3,R4        ;STORE R3 IN R4 FOR LINE SCAN
4562 016534 105724          TSTB  (R4)+        ;ADJUST R4
4563 016536 000751          BR    3$          ;GET MINIMUM CYLINDER
4564
4565 016540 013737 027332 001602 6$:   MOV   LSTCYL,PARCYL ;LOAD MAX CYLINDER = 632/1456
4566 016546 005037 001604          7$:   CLR   PARCYL+2    ;LOAD MIN CYLINDER = 0
4567 016552 013765 001602 000070 8$:   MOV   PARCYL,P.MXCL(R5) ;LOAD MAX CYLINDER
4568 016560 013765 001604 000066          MOV   PARCYL+2,P.MNCL(R5) ;LOAD MIN CYLINDER
4569
4570 016566 104401 060161          10$:  TYPE  ,PAR003      ;TYPE 'TRACK MAX,MIN'
4571 016572 004037 020516          JSR   R0,TSTDEF   ;CHECK FOR DEFAULT PARAMETERS
4572 016576 016710          13$     ;COMMA DETECTED
4573 016600 016724          14$     ;CARRIAGE RETURN DETECTED
4574 016602 020370          P2$     ;CONTROL Z <^Z> DETECTED
4575 016604 016352          POS     ;CONTROL C <^C> DETECTED

```

4576	016606	010346				MOV	R3,-(SP)	:STORE BUFFER ADDRESS
4577	016610	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
4578	016614	016672				12\$:ERROR RETURN
4579	016616	012637	001606			MOV	(SP)+,PARTRK	:STORE MAX TRACK ENTRY
4580	016622	022737	000002	001606		CMP	#D.MXTR,PARTRK	:CHECK IF LESS OR EQUAL TO
4581								: MAXIMUM TRACK ADDRESS
4582	016630	103420				BLO	12\$:NO, TRY AGAIN
4583	016632	010304				MOV	R3,R4	:STORE R3 IN R4 FOR LINE SCAN
4584	016634	105714			66\$:	TSTB	(R4)	:CHECK FOR CARRIAGE RETURN
4585	016636	001435				BEQ	15\$:YES, EXIT
4586	016640	122427	000054			CMPB	(R4)+, #' ,	:CHECK FOR COMMA
4587	016644	001373				BNE	66\$:NO, GO TEST NEXT CHARACTER
4588	016646	010446			11\$:	MOV	R4,-(SP)	:STORE BUFFER ADDRESS
4589	016650	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
4590	016654	016672				12\$:ERROR RETURN
4591	016656	012637	001610			MOV	(SP)+,PARTRK+2	:STORE MIN TRACK ENTRY
4592	016662	023737	001606	001610		CMP	PARTRK,PARTRK+2	:CHECK IF LEGAL ENTRY
4593	016670	103022				BHIS	16\$:YES, LOAD MAX AND MIN TRACK
4594	016672				12\$:			
4595	016672	010337	016700			MOV	R3,67\$:LOAD BUFFER ADDRESS
4596	016676	104401				TYPE		:TYPE RECEIVED INPUT
4597	016700	000000			67\$:	.WORD	0	:BUFFER ADDRESS
4598	016702	104401	001164			TYPE	.SQUES	:TYPE QUESTION MARK
4599	016706	000727				BR	10\$:TRY AGAIN
4600								
4601	016710	012737	000002	001606	13\$:	MOV	#D.MXTR,PARTRK	:LOAD MAX TRACK = 2
4602	016716	010304				MOV	R3,R4	:STORE R3 IN R4 FOR LINE SCAN
4603	016720	105724				TSTB	(R4)+	:ADJUST R4
4604	016722	000751				BR	11\$:GET MINIMUM TRACK
4605								
4606	016724	012737	000002	001606	14\$:	MOV	#D.MXTR,PARTRK	:LOAD MAX TRACK = 2
4607	016732	005037	001610		15\$:	CLR	PARTRK+2	:LOAD MIN TRACK = 0
4608	016736	113765	001606	000073	16\$:	MOVB	PARTRK,P.MXTR(R5)	:LOAD MAX TRACK
4609	016744	113765	001610	000072		MOVB	PARTRK+2,P.MNTR(R5)	:LOAD MIN TRACK
4610								
4611	016752	104401	060201		18\$:	TYPE	.PAR004	:TYPE "SECTOR MAX,MIN"
4612	016756	004037	020516			JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
4613	016762	017074				21\$:COMMA DETECTED
4614	016764	017110				22\$:CARRIAGE RETURN DETECTED
4615	016766	020402				P3\$:CONTROL Z <^Z> DETECTED
4616	016770	016352				P0\$:CONTROL C <^C> DETECTED
4617	016772	010346				MOV	R3,-(CP)	:STORE BUFFER ADDRESS
4618	016774	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
4619	017000	017056				20\$:ERROR RETURN
4620	017002	012637	001612			MOV	(SP)+,PARSEC	:STORE MAX SECTOR ENTRY
4621	017006	022737	000025	001612		CMP	#D.MXSC,PARSEC	:CHECK IF LESS OR EQUAL TO
4622								: MAXIMUM SECTOR ADDRESS
4623	017014	103420				BLO	20\$:NO, TRY AGAIN
4624	017016	010304				MOV	R3,R4	:STORE R3 IN R4 FOR LINE SCAN
4625	017020	105714			68\$:	TSTB	(R4)	:CHECK FOR CARRIAGE RETURN
4626	017022	001435				BEQ	23\$:YES, EXIT
4627	017024	122427	000054			CMPB	(R4)+, #' ,	:CHECK FOR COMMA
4628	017030	001373				BNE	68\$:NO, GO TEST NEXT CHARACTER
4629	017032	010446			19\$:	MOV	R4,-(SP)	:STORE BUFFER ADDRESS
4630	017034	004737	054406			JSR	PC,OCTBIN	:CONVERT TO BINARY
4631	017040	017056				20\$:ERROR RETURN

```

4632 017042 012637 001614      MOV      (SP)+,PARSEC+2  ;STORE MIN SECTOR ENTRY
4633 017046 023737 001612 001614  CMP      PARSEC,PARSEC+2 ;CHECK IF LEGAL ENTRY
4634 017054 103022      BHS      24$           ;YES, LOAD MAX AND MIN SECTOR
4635 017056      20$:
4636 017056 010337 017064      MOV      R3,69$       ;LOAD BUFFER ADDRESS
4637 017062 104401      TYPE     ;TYPE RECEIVED INPUT
4638 017064 000000 69$:      .WORD   0           ;BUFFER ADDRESS
4639 017066 104401 001164      TYPE     $QUES       ;TYPE QUESTION MARK
4640 017072 000727      BR       18$         ;TRY AGAIN
4641
4642 017074 012737 000025 001612 21$:      MOV      #D.MXSC,PARSEC ;LOAD MAX SECTOR = 21
4643 017102 010304      MOV      R3,R4       ;STORE R3 IN R4 FOR LINE SCAN
4644 017104 105724      TSTB    (R4)+        ;ADJUST R4
4645 017106 000751      BR       19$         ;GET MINIMUM SECTOR
4646
4647 017110 012737 000025 001612 22$:      MOV      #D.MXSC,PARSEC ;LOAD MAX SECTOR = 21
4648 017116 005037 001614 23$:      CLR      PARSEC+2    ;LOAD MIN SECTOR = 0
4649 017122 113765 001612 000075 24$:      MOVVB   PARSEC,P.MXSC(R5) ;LOAD MAX SECTOR
4650 017130 113765 001614 000074      MOVVB   PARSEC+2,P.MNSC(R5) ;LOAD MIN SECTOR
4651
4652 017136 104401 060222 26$:      TYPE     ,PAR005     ;TYPE 'READ/WRITE RATIO ='
4653 017142 004037 020516      JSR     R0,TSTDEF    ;CHECK FOR DEFAULT PARAMETERS
4654 017146 017176      27$:      ;COMMA DETECTED
4655 017150 017222      29$:      ;CARRIAGE RETURN DETECTED
4656 017152 020414      P4$     ;CONTROL Z <^Z> DETECTED
4657 017154 016352      POS     ;CONTROL C <^C> DETECTED
4658 017156 010346      MOV     R3,-(SP)     ;STORE BUFFER ADDRESS
4659 017160 004737 054406      JSR     PC,OCTBIN    ;CONVERT BINARY
4660 017164 017176      27$:      ;ERROR RETURN
4661 017166 012604      MOV     (SP)+,R4     ;STORE INPUT
4662 017170 022704 000010      CMP     #10,R4      ;CHECK IF 0-7
4663 017174 101007      BHI     28$         ;YES, LOAD READ/WRITE RATIO
4664 017176      27$:
4665 017176 010337 017204      MOV     R3,70$       ;LOAD BUFFER ADDRESS
4666 017202 104401      TYPE     ;TYPE RECEIVED INPUT
4667 017204 000000 70$:      .WORD   0           ;BUFFER ADDRESS
4668 017206 104401 001164      TYPE     $QUES       ;TYPE QUESTION MARK
4669 017212 000751      BR       26$         ;TRY AGAIN
4670
4671 017214 110465 000076 28$:      MOVVB   R4,P.RATE(R5) ;LOAD READ/WRITE RATIO
4672 017220 000403      BR       30$         ;ASK 'WRITE CHECK AFTER WRITE ='
4673
4674 017222 112765 000003 000076 29$:      MOVVB   #D.RATE,P.RATE(R5) ;LOAD DEFAULT READ/WRITE
4675
4676 017230 104401 060245 30$:      TYPE     ,PAR006     ;TYPE 'WRITE CHECK AFTER WRITE ='
4677 017234 004037 020516      JSR     R0,TSTDEF    ;CHECK FOR DEFAULT PARAMETERS
4678 017240 017266      31$:      ;COMMA DETECTED
4679 017242 017304      32$:      ;CARRIAGE RETURN DETECTED
4680 017244 020422      P5$     ;CONTROL Z <^Z> DETECTED
4681 017246 016352      POS     ;CONTROL C <^C> DETECTED
4682 017250 122713 000131      CMPB   #'Y,(R3)     ;CHECK IF YES
4683 017254 001013      BNE     32$         ;NO, LOAD DEFAULT
4684 017256 112765 177777 000077      MOVVB   #-1,P.AWCK(R5) ;SET WRITE CHECK AFTER WRITE
4685 017264 000411      BR       34$         ;GET CORRECTABLE READ ERROR
4686      ; THRESHOLD
4687

```

4688	017266			31\$:	MOV	R3,71\$:LOAD BUFFER ADDRESS
4689	017266	010337	017274		TYPE		:TYPE RECEIVED INPUT
4690	017272	104401			.WORD	0	:BUFFER ADDRESS
4691	017274	000000		71\$:	TYPE	\$QUES	:TYPE QUESTION MARK
4692	017276	104401	001164		BR	30\$:TRY AGAIN
4693	017302	000752					
4694							
4695	017304	105065	000077	32\$:	CLRB	P.AWCK(R5)	:CLEAR WRITE CHECK AFTER WRITE
4696							
4697	017310	104401	060277	34\$:	TYPE	.PAR007	:TYPE 'CORRECTABLE READ ERROR
4698							: THRESHOLD
4699	017314	004037	020516		JSR	RO,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
4700	017320	017346			35\$:COMMA DETECTED
4701	017322	017364			36\$:CARRIAGE RETURN DETECTED
4702	017324	020426			P6\$:CONTROL Z <^Z> DETECTED
4703	017326	016352			P0\$:CONTROL C <^C> DETECTED
4704	017330	010346			MOV	R3,-(SP)	:STORE BUFFER ADDRESS
4705	017332	004737	054542		JSR	PC,DECBIN	:CONVERT TO BINARY
4706	017336	017346			35\$:ERROR RETURN
4707	017340	012665	000100		MOV	(SP)+,P.CERT(R5)	:LOAD CORRECTABLE READ ERROR
4708							: THRESHOLD
4709	017344	100012			BPL	38\$:IF NOT NEGATIVE, GET UNCORRECTABLE
4710							: READ ERROR THRESHOLD
4711							
4712	017346			35\$:	MOV	R3,72\$:LOAD BUFFER ADDRESS
4713	017346	010337	017354		TYPE		:TYPE RECEIVED INPUT
4714	017352	104401			.WORD	0	:BUFFER ADDRESS
4715	017354	000000		72\$:	TYPE	\$QUES	:TYPE QUESTION MARK
4716	017356	104401	001164		BR	34\$:TRY AGAIN
4717	017362	000752					
4718							
4719	017364	012765	000012 000100	36\$:	MOV	#D.CERT,P.CERT(R5)	:LOAD DEFAULT CORRECTABLE READ
4720							: ERROR THRESHOLD
4721							
4722	017372	104401	060342	38\$:	TYPE	.PAR008	:TYPE 'UNCORRECTABLE READ ERROR
4723							: THRESHOLD ="
4724	017376	004037	020516		JSR	RO,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
4725	017402	017430			39\$:COMMA DETECTED
4726	017404	017446			40\$:CARRIAGE RETURN DETECTED
4727	017406	020434			P7\$:CONTROL Z <^Z> DETECTED
4728	017410	016352			P0\$:CONTROL C <^C> DETECTED
4729	017412	010346			MOV	R3,-(SP)	:STORE PUFFER ADDRESS
4730	017414	004737	054542		JSR	PC,DECBIN	:CONVERT TO BINARY
4731	017420	017430			39\$:ERROR RETURN
4732	017422	012665	000102		MOV	(SP)+,P.UERT(R5)	:LOAD UNCORRECTABLE READ ERROR
4733							: THRESHOLD
4734	017426	100012			BPL	42\$:IF NOT NEGATIVE, GET SEEK
4735							: ERROR THRESHOLD
4736	017430			39\$:	MOV	R3,73\$:LOAD BUFFER ADDRESS
4737	017430	010337	017436		TYPE		:TYPE RECEIVED INPUT
4738	017434	104401			.WORD	0	:BUFFER ADDRESS
4739	017436	000000		73\$:	TYPE	\$QUES	:TYPE QUESTION MARK
4740	017440	104401	001164		BR	38\$:TRY AGAIN
4741	017444	000752					
4742							
4743	017446	012765	000005 000102	40\$:	MOV	#D.UERT,P.UERT(R5)	:LOAD DEFAULT UNCORRECTABLE READ


```
4744                                     : ERROR THRESHOLD
4745
4746 017454 104401 060407          42$: TYPE ,PAR009          :TYPE ''SEEK ERROR THRESHOLD ='
4747 017460 004037 020516          JSR  RO,TSTDEF          :CHECK FOR DEFAULT PARAMETERS
4748 017464 017512                  43$                      :COMMA DETECTED
4749 017466 017530                  44$                      :CARRIAGE RETURN DETECTED
4750 017470 020442                  P8$                      :CONTROL Z <^Z> DETECTED
4751 017472 016352                  P0$                      :CONTROL C <^C> DETECTED
4752 017474 010346                  MOV  R3,-(SP)           :STORE BUFFER ADDRESS
4753 017476 004737 054542          JSR  PC,DECBIN         :CONVERT TO BINARY
4754 017502 017512                  43$                      :ERROR RETURN
4755 017504 012665 000104          MOV  (SP)+,P.SKET(R5)  :LOAD SEEK ERROR THRESHOLD
4756 017510 100012                  BPL  46$               :CHECK IF INHIBIT ERROR CORRECTION
4757                                     : AND RETRY
4758 017512          43$:
4759 017512 010337 017520          MOV  R3,74$           :LOAD BUFFER ADDRESS
4760 017516 104401                  TYPE                      :TYPE RECEIVED INPUT
4761 017520 000000          74$: .WORD 0          :BUFFER ADDRESS
4762 017522 104401 001164          TYPE ,SQUES           :TYPE QUESTION MARK
4763 017526 000752                  BR  42$               :TRY AGAIN
4764
4765 017530 012765 000005 000104 44$: MOV  #D.SKET,P.SKET(R5) ;LOAD DEFAULT SEEK ERROR THRESHOLD
4766
4767 017536 104401 060436          46$: TYPE ,PAR010          :TYPE ''INHIBIT ERROR CORRECTION
4768                                     : AND RETRY
4769 017542 004037 020516          JSR  RO,TSTDEF          :CHECK FOR DEFAULT PARAMETERS
4770 017546 017574                  47$                      :COMMA DETECTED
4771 017550 017612                  48$                      :CARRIAGE RETURN DETECTED
4772 017552 020450                  P9$                      :CONTROL Z <^Z> DETECTED
4773 017554 016352                  P0$                      :CONTROL C <^C> DETECTED
4774 017556 122713 000131          CMPB #'Y,(R3)         :CHECK IF YES
4775 017562 001013                  BNE  48$               :LOAD DEFAULT
4776 017564 112765 177777 000120  MOVB #-1,P.IERC(R5)   :SET INHIBIT ERROR CORRECTION
4777 017572 000411                  BR  50$               :GET COMMAND COUNT THRESHOLD
4778
4779 017574          47$:
4780 017574 010337 017602          MOV  R3,75$           :LOAD BUFFER ADDRESS
4781 017600 104401                  TYPE                      :TYPE RECEIVED INPUT
4782 017602 000000          75$: .WORD 0          :BUFFER ADDRESS
4783 017604 104401 001164          TYPE ,SQUES           :TYPE QUESTION MARK
4784 017610 000752                  BR  46$               :TRY AGAIN
4785
4786 017612 105065 000120          48$: CLRB P.IERC(R5)   :CLEAR INHIBIT ERROR CORRECTION
4787
4788 017616 104401 060503          50$: TYPE ,PAR011          :TYPE ''OPERATION COUNT THRESHOLD*65K ='
4789 017622 004037 020516          JSR  RO,TSTDEF          :CHECK FOR DEFAULT PARAMETERS
4790 017626 017672                  51$                      :COMMA DETECTED
4791 017630 017710                  52$                      :CARRIAGE RETURN DETECTED
4792 017632 020454                  P10$                     :CONTROL Z <^Z> DETECTED
4793 017634 016352                  P0$                      :CONTROL C <^C> DETECTED
4794 017636 010346                  MOV  R3,-(SP)           :STORE BUFFER ADDRESS
4795 017640 004737 054542          JSR  PC,DECBIN         :CONVERT TO BINARY
4796 017644 017672                  51$                      :ERROR RETURN
4797 017646 012665 000110          MOV  (SP)+,P.MXCD+2(R5);STORE NUMBER OF 65K BLOCKS
4798 017652 001407                  BEQ  51$               :IF ZERO, TRY AGAIN
4799 017654 100406                  BMI  51$               :IF MINUS, TRY AGAIN
```

4800	017656	005365	000110		DEC	P.MXCD+2(R5)	:DECREMENT BY 1
4801	017662	012765	177777	000106	MOV	#-1,P.MXCD(R5)	
4802	017670	000415			BR	54\$:GET WORD TRANSFERREC THRESHOLD
4803							
4804	017672			51\$:			
4805	017672	010337	017700		MOV	R3,76\$:LOAD BUFFER ADDRESS
4806	017676	104401			TYPE		:TYPE RECEIVED INPUT
4807	017700	000000		76\$:	.WORD	0	:BUFFER ADDRESS
4808	017702	104401	001164		TYPE	\$QUES	:TYPE QUESTION MARK
4809	017706	000743			BR	50\$:TRY AGAIN
4810							
4811	017710	012765	077777	000110	MOV	#D.CTRH,P.MXCD+2(R5)	:LOAD DEFAULT COMMAND THRESHOLD
4812	017716	012765	177770	000106	MOV	#D.CTRL,P.MXCD(R5)	
4813							
4814	017724	104401	060543	54\$:	TYPE	,PAR012	:TYPE 'WORD TRANSFERRED THRESHOLD*520K ='
4815	017730	004037	020516		JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
4816	017734	020032			55\$:COMMA DETECTED
4817	017736	020050			56\$:CARRIAGE RETURN DETECTED
4818	017740	020470			P11\$:CONTROL Z <^Z> DETECTED
4819	017742	016352			POS		:CONTROL C <^C> DETECTED
4820	017744	010346			MOV	R3,-(SP)	:STORE BUFFER ADDRESS
4821	017746	004737	054542		JSR	PC,DECBIN	:CONVERT TO BINARY
4822	017752	020032			55\$:ERROR RETURN
4823	017754	012665	000112		MOV	(SP)+,P.MXWT(R5)	:STORE NUMBER OF 520K BLOCKS
4824	017760	001424			BEQ	55\$:IF ZERO, TRY AGAIN
4825	017762	100423			BMI	55\$:IF NEGATIVE, TRY AGAIN
4826	017764	005065	000114		CLR	P.MXWT+2(R5)	:CLEAR HIGH ORDER BITS
4827	017770	006165	000112		ROL	P.MXWT(R5)	:SHIFT 4 BITS LEFT
4828	017774	006165	000114		ROL	P.MXWT+2(R5)	
4829	020000	006165	000112		ROL	P.MXWT(R5)	
4830	020004	006165	000114		ROL	P.MXWT+2(R5)	
4831	020010	006165	000112		ROL	P.MXWT(R5)	
4832	020014	006165	000114		ROL	P.MXWT+2(R5)	
4833	020020	006165	000112		ROL	P.MXWT(R5)	
4834	020024	006165	000114		ROL	P.MXWT+2(R5)	
4835	020030	000415			BR	60\$:CHECK FOR SAMPLED COMPARES
4836							
4837	020032			55\$:			
4838	020032	010337	020040		MOV	R3,77\$:LOAD BUFFER ADDRESS
4839	020036	104401			TYPE		:TYPE RECEIVED INPUT
4840	020040	000000		77\$:	.WORD	0	:BUFFER ADDRESS
4841	020042	104401	001164		TYPE	\$QUES	:TYPE QUESTION MARK
4842	020046	000726			BR	54\$:TRY AGAIN
4843							
4844	020050	012765	077777	000114	MOV	#D.WTHI,P.MXWT+2(R5)	:LOAD DEFAULT WORD TRANSFER
4845	020056	012765	177770	000112	MOV	#D.WTLO,P.MXWT(R5)	: THRESHOLD
4846							
4847	020064	005737	001362	60\$:	TST	SOFCMP	:CHECK IF ANY SOFTWARE COMPARES ARE TO BE MADE
4848	020070	001430			BEQ	PREXAR	:NO, DETERMINE TRACK EXCLUSION
4849	020072	104401	061041		TYPE	,PAR021	:TYPE 'SAMPLED COMPARES ='
4850	020076	004037	020516		JSR	R0,TSTDEF	:CHECK FOR DEFAULT PARAMETERS
4851	020102	020130			61\$:COMMA DETECTED
4852	020104	020146			62\$:CARRIAGE RETURN DETECTED
4853	020106	020504			P12\$:CONTROL Z <^Z> DETECTED
4854	020110	016352			POS		:CONTROL C <^C> DETECTED
4855	020112	122713	000116		CMPB	#'N,(R3)	:CHECK IF NO

4856	020116	001013			BNE	62\$:NO, MAKE SAMPLED COMPARES
4857	020120	112765	177777	000116	MOVB	#-1,P.SMPL(R5)		:DO NOT MAKE SAMPLED COMPARES
4858	020126	000411			BR	PREXAR		:DETERMINE TRACK EXCLUSION
4859								
4860	020130				61\$:			
4861	020130	010337	020136		MOV	R3,78\$:LOAD BUFFER ADDRESS
4862	020134	104401			TYPE			:TYPE RECEIVED INPUT
4863	020136	000000			78\$:	.WORD	0	:BUFFER ADDRESS
4864	020140	104401	001164		TYPE	.SQUES		:TYPE QUESTION MARK
4865	020144	000747			BR	60\$:TRY AGAIN
4866								
4867	020146	105065	000116		62\$:	CLRB	P.SMPL(R5)	:MAKE SAMPLED COMPARES
4868								
4869	020152	104401	060676		PREXAR:	TYPE	.PAR019	:TYPE "CHANGE EXCUDED PACK AREA ="
4870	020156	104401	060650		TYPE	.PAR018		
4871	020162	004037	020516		JSR	R0,TSTDEF		:CHECK FOR DEFAULT PARAMETERS
4872	020166	020210			1\$:COMMA DETECTED
4873	020170	020350			50\$:CARRIAGE RETURN DETECTED
4874	020172	020350			50\$:CONTROL Z <^Z> DETECTED
4875	020174	016352			POS			:CONTROL C <^C> DETECTED
4876	020176	122713	000131		CMPB	#'Y,(R3)		:CHECK IF YES
4877	020202	001411			BEQ	2\$:YES, LOAD NEW TRACK EXCULSIONS
4878	020204	000137	020350		JMP	50\$:NO, RETURN
4879								
4880	020210				1\$:			
4881	020210	010337	020216		MOV	R3,64\$:LOAD BUFFER ADDRESS
4882	020214	104401			TYPE			:TYPE RECEIVED INPUT
4883	020216	000000			64\$:	.WORD	0	:BUFFER ADDRESS
4884	020220	104401	001164		TYPE	.SQUES		:TYPE QUESTION MARK
4885	020224	000752			BR	PREXAR		:TRY AGAIN
4886								
4887	020226	010146			2\$:	MOV	R1,-(SP)	:STORE R1 ON STACK
4888	020230	010501			MOV	R5,R1		:STORE PARMETER BLOCK ADDRESS
4889	020232	062701	000232		ADD	#P.EXAR,R1		:CALCULATE EXCLUSION AREA
4890	020236	012704	000012		MOV	#10.,R4		:LOAD COUNT
4891	020242	005021			5\$:	CLR	(R1)+	:CLEAR EXCLUSION AREA
4892	020244	005304			DEC	R4		:DECREMENT COUNT
4893	020246	001375			BNE	5\$:CHECK IF DONE
4894	020250	162701	000024		SUB	#20.,R1		:REINITIALIZE POINTER
4895	020254	004037	020664		JSR	R0,LDEXAR		:GET FIRST EXCLUDED AREA
4896	020260	060610			PAR013			
4897	020262	020346			45\$:DEFAULT RETURN
4898	020264	020340			40\$:CONTROL C <^C> RETURN
4899	020266	004037	020664		JSR	R0,LDEXAR		:GET SECOND EXCLUDED AREA
4900	020272	060616			PAR014			
4901	020274	020346			45\$:DEFAULT RETURN
4902	020276	020340			40\$:CONTROL C <^C> RETURN
4903	020300	004037	020664		JSR	R0,LDEXAR		:GET THIRD EXCLUDED AREA
4904	020304	060625			PAR015			
4905	020306	020346			45\$:DEFAULT RETURN
4906	020310	020340			40\$:CONTROL C <^C> RETURN
4907	020312	004037	020664		JSR	R0,LDEXAR		:GET FOURTH EXCLUDED AREA
4908	020316	060633			PAR016			
4909	020320	020346			45\$:DEFAULT RETURN
4910	020322	020340			40\$:CONTROL C <^C> RETURN
4911	020324	004037	020664		JSR	R0,LDEXAR		:GET FIFTH EXCLUDED AREA

```
4912 020330 060642          PAR017
4913 020332 020346          45$          ;DEFAULT RETURN
4914 020334 020340          40$          ;CONTROL C <^C> RETURN
4915 020336 000403          BR           45$          ;RETURN
4916
4917 020340 012601          40$: MOV      (SP)+,R1      ;RESTORE R1
4918 020342 000137 016352  JMP      POS           ;START AGAIN
4919
4920 020346 012601          45$: MOV      (SP)+,R1      ;RESTORE R1
4921 020350 012603          50$: MOV      (SP)+,R3      ;RESTORE R3
4922 020352 012604          MOV      (SP)+,R4      ;RESTORE R4
4923 020354 000207          RTS      PC           ;RETURN
4924
4925 020356 013765 027332 000070 P1$: MOV      LSTCYL,P.MXCL(R5) ;LOAD DEFAULT MAXIMUM CYLINDER = 632/1456
4926 020364 005065 000066          CLR      P.MNCL(R5)    ;LOAD DEFAULT MINIMUM CYLINDER
4927 020370 112765 000002 000073 P2$: MOV      #D.MXTR,P.MXTR(R5) ;LOAD DEFAULT MAXIMUM TRACK = 2
4928 020376 105065 000072          CLRB     P.MNTR(R5)    ;LOAD DEFAULT MINIMUM TRACK
4929 020402 112765 000025 000075 P3$: MOV      #D.MXSC,P.MXSC(R5) ;LOAD DEFAULT MAXIMUM SECTOR = 21
4930 020410 105065 000074          CLRB     P.MNSC(R5)    ;LOAD DEFAULT MINIMUM SECTOR
4931 020414 112765 000003 000076 P4$: MOV      #D.RATE,P.RATE(R5) ;LOAD READ/WRITE RATIO
4932 020422 105065 000077          CLRB     P.AWCK(R5)    ;LOAD WRITE CHECK AFTER WRITE
4933 020426 012765 000012 000100 P6$: MOV      #D.CERT,P.CERT(R5) ;LOAD CORRECTABLE READ ERROR THRESHOLD
4934 020434 012765 000005 000102 P7$: MOV      #D.UERT,P.UERT(R5) ;LOAD UNCORRECTABLE READ ERROR THRESHOLD
4935 020442 012765 000005 000104 P8$: MOV      #D.SKET,P.SKET(R5) ;LOAD SEEK ERROR THRESHOLD
4936 020450 105065 000120          CLRB     P.IERC(R5)    ;CLEAR INHIBIT ERROR CORRECTION
4937 020454 012765 077777 000110 P10$: MOV     #D.CTRH,P.MXCD+2(R5) ;LOAD COMMAND THRESHOLD
4938 020462 012765 177770 000106          MOV     #D.CTRL,P.MXCD(R5)
4939 020470 012765 077777 000114 P11$: MOV     #D.WTHI,P.MXWT+2(R5) ;LOAD WORD TRANSFERED THRESHOLD
4940 020476 012765 177770 000112          MOV     #D.WTLO,P.MXWT(R5)
4941 020504 105065 000116          P12$: CLRB     P.SMPL(R5) ;MAKE SAMPLED COMPARES
4942
4943 020510 012603          MOV      (SP)+,R3      ;RESTORE R3
4944 020512 012604          MOV      (SP)+,R4      ;RESTORE R4
4945 020514 000207          RTS      PC           ;RETURN
```

4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001

020516 104403
020520 012603
020522 121327 000054
020526 001415
020530 005720
020532 105713
020534 001412
020536 005720
020540 122713 000032
020544 001406
020546 005720
020550 122713 000003
020554 001402
020556 005720
020560 000200
020562 011000
020564 000200
020566 104401 061064
020572 004037 020516
020576 020622
020600 020622
020602 020622
020604 020622
020606 121327 000131
020612 001406
020614 121327 000116
020620 001412
020622 104401 056440
020626 000757
020630 142765 000004 000007
020636 012737 000632 027332
020644 000207
020646 152765 000004 000007
020654 012737 001456 027332

.SBTTL TEST DEFAULT PARAMETERS ROUTINE

```

*****
*
* THIS ROUTINE WILL READ A LINE OF ASCII TEXT AND CHECK
* IF PARAMETER OR REST OF PARAMETERS WILL BE DEFAULTED.
* IT RETURNS WITH R3 POINTING TO THE ADDRESS OF THE BUFFER.
*
*CALL JSR RO,TSTDEF
*      <ADDRESS OF COMMA DETECTION RETURN>
*      <ADDRESS OF CARRIAGE RETURN DETECTION RETURN>
*      <ADDRESS OF CONTROL Z <^Z> DETECTION RETURN>
*      <ADDRESS OF CONTROL C <^C> DETECTION RETURN>
*      RETURN
*****

```

```

TSTDEF: RDLIN      :GET RESPONSE
MOV      (SP)+,R3  :STORE BUFFER ADDRESS
CMPB     (R3),#' , :CHECK IF COMMA
BEQ      10$       :YES, RETURN
TST      (R0)+     :ADJUST RO
TSTB     (R3)      :CHECK IF CARRIAGE RETURN
BEQ      10$       :YES, RETURN
TST      (R0)+     :ADJUST RO
CMPB     #32,(R3)  :CHECK IF CONTROL Z <^Z>
BEQ      10$       :YES, RETURN
TST      (R0)+     :ADJUST RO
CMPB     #3,(R3)   :CHECK IF CONTROL C <^C>
BEQ      10$       :YES, RETURN
TST      (R0)+     :ADJUST RO
RTS      R0        :RETURN

10$:     MOV      (R0),R0 :STORE RETURN ADDRESS
RTS      R0        :RETURN

DTYPE:   TYPE     ,PAR022 :TYPE "TESTING RK06? Y OR N"
JSR      R0,TSTDEF :GET INPUT
1$       :COMMA
1$       :CR
1$       :CONT Z
1$       :CONT C

CMPB     (R3),#'Y  :SEE IF RK06
BEQ      2$       :BR IF YES
CMPB     (R3),#'N  :SEE IF RK07
BEQ      3$       :BR IF YES

1$:     TYPE     ,ILLCMD :ELSE ERROR
BR       DTYPE    :TRY AGAIN

2$:     BICB     #B.CDT,P.CS1H(R5) :RK06
MOV      #632,LSTCYL :LAST CYLINDER ADDRESS
RTS      PC

3$:     BISB     #B.CDT,P.CS1H(R5) :RK07
MOV      #1456,LSTCYL :LAST CYL ADDRESS

```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 E 9
TEST DEFAULT PARAMETERS ROUTINE PAGE 111

SEQ 0108

5002 020662 000207
5003
5004

RTS PC

.SBTTL LOAD DISK EXCLUDED AREAS

```

*****
*
* THIS ROUTINE WILL ACCEPT TTY INPUT FOR EXCLUDED AREAS AND
* LOAD APPROPRIATE IDENTIFICATION NUMBERS IN THE EXCLUDED AREAS OF
* THE PARAMETER BLOCK. THE FOLLOWING THREE FORMATS ARE
* ACCEPTED BY THIS ROUTINE:

```

```

*
*       CYL
*       CYL,TRK
*       CYL1,TRK1,CYL2,TRK2

```

THE FOLLOWING FORMULA IS USED:

```

*
*       CYL1*66+TRK1*22+1 = BEGINNING OF EXCLUDED AREA
*       CYL2*66+TRK2*22+1 = END OF EXCLUDED AREA

```

```

*
* REGISTER      USE
* -----      ---
*
* R1            ADDRESS IN EXCLUDED AREA
* R3            ADDRESS OF INPUT STRING
* R4            ADDRESS IN INPUT STRING
* R5            PARAMETER BLOCK ADDRESS

```

```

*CALL JSR      R0,LDEXAR
*      <ADDRESS OF MESSAGE TO PRINT>
*      <ADDRESS FOR CARRIAGE RETURN OR CONTROL Z <^Z> DETECTION>
*      <ADDRESS FOR CONTROL C <^C> DETECTION>
*      RETURN

```

```

5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039 020664 012037 020672
5040
5041 020670 104401
5042 020672 000000
5043 020674 104401 060650
5044 020700 004037 020516
5045 020704 021310
5046 020706 021332
5047 020710 021332
5048 020712 021330
5049 020714 010346
5050 020716 004737 054406
5051 020722 021310
5052 020724 023716 027332
5053 020730 103566
5054 020732 011637 001616
5055 020736 006316
5056 020740 062637 001616
5057 020744 010304
5058
5059 020746 105714
5060 020750 001525

```

```

LDEXAR: MOV      (R0)+,2$      ;LOAD NUMBER OF EXCUSION AREA
1$:      TYPE
2$:      .WORD      0          ;TYPE 'NTH EXCLUDED AREA ='
          TYPE      ,PAR018
          JSR      R0,TSTDEF   ;CHECK FOR DEFAULT PARAMETERS
          30$
          45$          ;COMMA DETECTED
          45$          ;CARRIAGE RETURN DETECTED
          40$          ;CONTROL Z <^Z> DETECTED
          MOV      R3,-(SP)    ;CONTROL C <^C> DETECTED
          JSR      PC,OCTBIN   ;STORE BUFFER ADDRESS
          30$          ;CONVERT TO BINARY
          CMP      LSTCYL,(SP) ;ERROR RETURN
          BLO      29$        ;CHECK IF LEGAL CYLINDER
          MOV      (SP),EXAREA ;NO, TRY AGAIN
          ASL      (SP)        ;MULTIPLY CYLINDER ADDRESS BY 3
          ADD      (SP)+,EXAREA
          MOV      R3,R4      ;STORE BEGINNING OF COMMAND STRING
64$:    TSTB     (R4)
          BEQ      10$        ;CHECK FOR CARRIAGE RETURN
          ;YES, EXIT

```

5061	020752	122427	000054	CMPB	(R4)+,#'	:CHECK FOR COMMA
5062	020756	001373		BNE	64\$:NO, GO TEST NEXT CHARACTER
5063	020760	105714		TSTB	(R4)	:CHECK IF CARRIAGE RETURN
5064	020762	001552		BEQ	30\$:YES, TRY AGAIN
5065	020764	121427	000054	CMPB	(R4),#'	:CHECK IF COMMA
5066	020770	001547		BEQ	30\$:YES, TRY AGAIN
5067	020772	010446		MOV	R4,-(SP)	:STORE BUFFER ADDRESS
5068	020774	004737	054406	JSR	PC,OCTBIN	:CONVERT TO BINARY
5069	021000	021310		30\$:ERROR RETURN
5070	021002	022716	000002	CMP	#2,(SP)	:CHECK IF LEGAL TRACK
5071	021006	103537		BLO	29\$:TRY AGAIN
5072	021010	063716	001616	ADD	EXAREA,(SP)	:PUT MULTIPLIER ON STACK
5073	021014	012746	000026	MOV	#2,-(SP)	:PUT MULTIPLIER ON STACK
5074	021020	004737	055210	JSR	PC,\$MULT	:DO MULTIPLICATION
5075	021024	005216		INC	(SP)	:ADD 1
5076	021026	012637	001616	MOV	(SP)+,EXAREA	:STORE VALUE
5077	021032	005726		TST	(SP)+	:THROW AWAY MOST SIGNIFICANT BITS
5078						
5079	021034	105714		65\$: TSTB	(R4)	:CHECK FOR CARRIAGE RETURN
5080	021036	001511		BEQ	20\$:YES, EXIT
5081	021040	122427	000054	CMPB	(R4)+,#'	:CHECK FOR COMMA
5082	021044	001373		BNE	65\$:NO, GO TEST NEXT CHARACTER
5083	021046	105714		TSTB	(R4)	:CHECK IF CARRIAGE RETURN
5084	021050	001517		BEQ	30\$:YES, TRY AGAIN
5085	021052	121427	000054	CMPB	(R4),#'	:CHECK IF COMMA
5086	021056	001514		BEQ	30\$:YES, TRY AGAIN
5087	021060	010446		MOV	R4,-(SP)	:STORE BUFFER ADDRESS
5088	021062	004737	054406	JSR	PC,OCTBIN	:CONVERT TO BINARY
5089	021066	021310		30\$:ERROR RETURN
5090	021070	023716	027332	CMP	LSTCYL,(SP)	:CHECK IF LEGAL CYLINDER
5091	021074	103504		BLO	29\$:NO, TRY AGAIN
5092	021076	011637	001620	MOV	(SP),EXAREA+2	:MULTIPLY CYLINDER ADDRESS BY 3
5093	021102	006316		ASL	(SP)	
5094	021104	062637	001620	ADD	(SP)+,EXAREA+2	
5095						
5096	021110	105714		66\$: TSTB	(R4)	:CHECK FOR CARRIAGE RETURN
5097	021112	001476		BEQ	30\$:YES, EXIT
5098	021114	122427	000054	CMPB	(R4)+,#'	:CHECK FOR COMMA
5099	021120	001373		BNE	66\$:NO, GO TEST NEXT CHARACTER
5100	021122	105714		TSTB	(R4)	:CHECK IF CARRIAGE RETURN
5101	021124	001471		BEQ	30\$:YES, TRY AGAIN
5102	021126	121427	000054	CMPB	(R4),#'	:CHECK IF COMMA
5103	021132	001466		BEQ	30\$:YES, TRY AGAIN
5104	021134	010446		MOV	R4,-(SP)	:STORE BUFFER ADDRESS
5105	021136	004737	054406	JSR	PC,OCTBIN	:CONVERT TO BINARY
5106	021142	021310		30\$:ERROR RETURN
5107	021144	022716	000002	CMP	#2,(SP)	:CHECK IF LEGAL TRACK
5108	021150	103456		BLO	29\$:TRY AGAIN
5109	021152	063716	001620	ADD	EXAREA+2,(SP)	:PUT MULTIPLIER ON STACK
5110	021156	012746	000026	MOV	#2,-(SP)	:PUT MULTIPLIER ON STACK
5111	021162	004737	055210	JSR	PC,\$MULT	:DO MULTIPLICATION
5112	021166	005216		INC	(SP)	:ADD 1
5113	021170	012637	001620	MOV	(SP)+,EXAREA+2	:STORE VALUE
5114	021174	005726		TST	(SP)+	:THROW AWAY MOST SIGNIFICANT BITS
5115	021176	023737	001616 001620	CMP	EXAREA,EXAREA+2	:CHECK IF LEGAL RANGE
5116	021204	103041		BHIS	30\$:NO, TRY AGAIN

5117	021206	013721	001616		MOV	EXAREA,(R1)+	:LOAD EXCLUDED AREA
5118	021212	013721	001620		MOV	EXAREA+2,(R1)+	
5119	021216	062700	000004		ADD	#4,R0	:ADJUST R0
5120	021222	000200			RTS	R0	:RETURN
5121							
5122	021224	013746	001616	10\$:	MOV	EXAREA,-(SP)	:LOAD MULTIPLIER
5123	021230	012746	000026		MOV	#22,-(SP)	:LOAD MULTICAND
5124	021234	004737	055210		JSR	PC,\$MULT	:CALCULATE VALUE
5125	021240	005216			INC	(SP)	:INCREMENT RESULTS
5126	021242	011621			MOV	(SP),(R1)+	:STORE EXCLUDED AREA
5127	021244	062716	000102		ADD	#66,(SP)	
5128	021250	012621			MOV	(SP)+,(R1)+	
5129	021252	005726			TST	(SP)+	:THROW AWAY MUST SIGNIFICANT BITS
5130	021254	062700	000004		ADD	#4,R0	:ADJUST R0
5131	021260	000200			RTS	R0	:RETURN
5132							
133	021262	013721	001616	20\$:	MOV	EXAREA,(R1)+	:STORE EXCLUDEED AREA
5134	021266	062737	000026	001616	ADD	#22,EXAREA	
5135	021274	013721	001616		MOV	EXAREA,(R1)+	
5136	021300	062700	000004		ADD	#4,R0	:ADJUST R0
5137	021304	000200			RTS	R0	:RETURN
5138							
5139	021306	005726		29\$:	TST	(SP)+	:ADJUST STACK
5140	021310	010337	021316	30\$:	MOV	R3,31\$:LOAD BUFFER ADDRESS
5141	021314	104401			TYPE		:TYPE RECEIVED INPUT
5142	021316	000000		31\$:	.WORD	0	:BUFFER ADDRESS
5143	021320	104401	001164		TYPE	\$QUES	:TYPE QUESTION MARK
5144	021324	000137	020670		JMP	1\$:TRY AGAIN
5145							
5146	021330	005720		40\$:	TST	(R0)+	:ADJUST R0
5147	021332	011000		45\$:	MOV	(R0),R0	:STORE SPECIAL RETURN ADDRESS
5148	021334	000200			RTS	R0	:RETURN

5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204

.SBTTL GENERATE NEW RANDOM COMMAND

```

*****
*
* IF SWITCH 14 IS RESET, THIS ROUTINE WILL STORE THE PREVIOUS
* COMMAND AND GENERATE A NEW RANDOM COMMAND. A CHECK IS MADE
* ON THE DISK ADDRESS AND THE DISK ADDRESS + THE NUMBER OF
* SECTORS TO BE READ OR WRITTEN. IF THE DATA TRANSFER WILL
* CAUSE DATA TO BE READ OR WRITTEN BEYOND THE MAXIMUM PACK
* ADDRESS OR USES ANY PART OF AN EXCLUDED AREA, A NEW DISK
* ADDRESS WILL BE RANDOMLY GENERATED.
*
* REGISTER          USE
* -----          ---
*
* R1                EXCLUSION AREA COUNT
* R3                EXCLUSION AREA INDEX
* R4                SECTOR POSITION
* R5                PARAMETER BLOCK ADDRESS
* (SP)             SECTORS TO BE TRANSFERRED
*
* IF SWITCH 14 IS SET, THIS ROUTINE WILL SEEK TO THE PREVIOUS
* DISK POSITION OR REISSUE THE LAST COMMAND DEPENDENT ON THE
* P.SEEK FLAG IN THE PARAMETER BLOCK.
*
* ROUTINES USED
* -----
*
* Q.PUSH
* $RAND
* $MULT
*
* CALL JSR PC,GENERT
* RETURN
*****

```

```

*****
GENERT: CLR P.RECT(R5) ;CLEAR ERROR FLAGS AND
        CLR P.DSTT(R5) ; REPLY COUNTS
        CLR P.ERR(R5)
*
        BIT #SW14,@SWR ;CHECK IF NEW COMMAND
* ; SHOULD BE GENERATED
        BEQ 10$ ;YES, GENERATE NEW COMMAND
        TSTB P.SEEK(R5) ;CHECK IF PREVIOUS CYLINDER ISSUED
        BNE 1$ ;YES, SET UP DATA TRANSFER
        MOVB #-1,P.SEEK(R5) ;INDICATE SEEK HAS BEEN ISSUED
        MOVB #SEEK,P.CMND(R5) ;LOAD SEEK PARAMETERS
        MOV P.LCYL(R5),P.CYLN(R5) ;LOAD PRIOR CYLINDER
        MOV P.LSEC(R5),P.SECT(R5) ;LOAD PRIOR SECTOR AND TRACK
        JSR R0,Q.PUSH ;PUT PARAMETER BLOCK ON
* ; COMMAND INITIATION QUEUE
        CINITQ
        RTS PC ;RETURN
*****

```

```

5188 021336 005065 000146
5189 021342 005065 000150
5190 021346 005065 000212
5191
5192 021352 032777 040000 157560
5193
5194 021360 001454
5195 021362 105765 000211
5196 021366 001020
5197 021370 112765 177777 000211
5198 021376 112765 000117 000001
5199 021404 016565 000136 000002
5200 021412 016565 000140 000004
5201 021420 004037 051564
5202 021424 001324
5203 021426 000207
5204

```

```

5205 021430 105065 000211 1$: CLRB P.SEK(R5) ;INDICATE DATA TRANSFER ISSUED
5206 021434 116565 000122 000121 MOVB P.RDPT(R5),P.DPAT(R5) ;LOAD LAST DATA PATTERN
5207 021442 116565 000123 000001 MOVB P.RCMD(R5),P.CMND(R5) ;LOAD LAST COMMAND
5208 021450 016565 000124 000002 MOV P.RCYL(R5),P.CYLN(R5) ;LOAD LAST CYLINDER ADDRESS
5209 021456 122765 000131 000001 CMPB #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK
5210 021464 001003 BNE 2$ ;NO, DO NOT SET FLAG
5211 021466 052765 000200 000014 BIS #W.WCK,P.PRST(R5) ;SET ISSUE WRITE FOR WRITE CHECK
5212 021474 016565 000126 000004 2$: MOV P.RSEC(R5),P.SECT(R5) ;LOAD LAST SECTOR AND TRACK
5213 021502 016565 000130 000012 MOV P.RWC(R5),P.WC(R5) ;LOAD LAST WORD COUNT
5214 021510 000207 RTS PC ;RETURN
5215
5216 021512 005077 157426 10$: CLR @STKS ;LOCK OUT TTY INTERRUPTS
5217 021516 010446 MOV R4,-(SP) ;STORE R4 ON STACK
5218 021520 010346 MOV R3,-(SP) ;STORE R3 ON STACK
5219 021522 010146 MOV R1,-(SP) ;STORE R1 ON STACK
5220 021524 010046 MOV R0,-(SP) ;STORE R0 ON STACK
5221 021526 116565 000123 000135 MOVB P.RCMD(R5),P.LCMD(R5) ;STORE LAST COMMAND
5222 021534 016565 000124 000136 MOV P.RCYL(R5),P.LCYL(R5) ;STORE LAST CYLINDER
5223 021542 016565 000126 000140 MOV P.RSEC(R5),P.LSEC(R5) ;STORE LAST SECTOR AND TRACK
5224 021550 016565 000130 000142 MOV P.RWC(R5),P.LWC(R5) ;STORE LAST WORD COUNT
5225 021556 116565 000122 000144 MOVB P.RDPT(R5),P.LDPT(R5) ;STORE LAST PATTERN
5226 021564 004737 055324 JSR PC,$RAND ;GENERATE RANDOM NUMBER
5227 021570 013746 055424 MOV $LONUM,-(SP) ;LOAD MULTIPLIER
5228 021574 013746 001354 MOV MAXBUF,-(SP) ;LOAD MULTIPLICAND
5229 021600 005216 INC (SP) ;INCREMENT WORD COUNT MAX.
5230 021602 004737 055210 JSR PC,$MULT ;CALCULATE WORD COUNT
5231 021606 005726 TST (SP)+ ;THROW AWAY LEAST SIGNIFICANT BITS
5232 021610 042716 100003 BIC #100003,(SP) ;MAKE DIVISIBLE BY 4 AND < 32K
5233 021614 001002 BNE 11$ ;IF NOT ZERO USE WORD COUNT
5234 021616 012716 000004 MOV #4,(SP) ;MAKE WORD COUNT 4
5235 021622 116604 000001 11$: MOVB 1(SP),R4 ;CALCULATE NUMBER OF SECTORS TO
5236 021626 105716 TSTB (SP) ; TRANSFERRED
5237 021630 001401 BEQ 12$ ;IF EVEN SECTORS DO NOT MODIFY
5238 021632 005204 INC R4 ; SECTOR COUNT
5239 021634 011646 12$: MOV (SP),-(SP) ;SAVE WORD COUNT
5240 021636 005416 NEG (SP) ;GENERATE NEGATIVE NUMBER
5241 021640 012665 000012 MOV (SP)+,P.WC(R5) ;LOAD PRESENT WORD COUNT
5242 021644 062765 000001 000152 ADD #1,P.NODR(R5) ;INCREMENT NUMBER OF ORDERS
5243 021652 005565 000154 ADC P.NODR+2(R5)
5244 021656 113746 055422 MOVB $HINUM,-(SP) ;RANDOM NUMBER FOR COMMAND
5245 021662 042716 177770 BIC #177770,(SP) ;KEEP LOW 3 BITS
5246 021666 126526 000076 CMPB P.RATE(R5),(SP)+ ;DETERMINE IF READ OR WRITE
5247 021672 101462 BLOS 15$ ;ISSUE READ
5248 021674 113746 055423 MOVB $HINUM+1,-(SP) ;RANDOM NUMBER FOR PATTERN
5249 021700 042716 177760 BIC #177760,(SP) ;KEEP LOW 4 BITS
5250 021704 111665 000122 MOVB (SP),P.RDPT(R5) ;STORE DATA PATTERN
5251 021710 112665 000121 MOVB (SP)+,P.DPAT(R5)
5252 021714 105765 000077 TSTB P.AWCK(R5) ;CHECK IF AUTO WRITE CHECK
5253 021720 001017 BNE 13$ ;YES, ISSUE WRITE CHECK
5254 021722 113746 055422 MOVB $HINUM,-(SP) ;RANDOM NUMBER FOR WRITE CHECK
5255 021726 042726 177707 BIC #177707,(SP)+ ;CHECK IF WRITE COMMAND
5256 021732 001412 BEQ 13$ ;YES, ISSUE WRITE CHECK
5257 021734 112765 000123 000123 MOVB #WRDATA,P.RCMD(R5) ;ISSUE WRITE DATA
5258 021742 062665 000156 ADD (SP)+,P.NWRT(R5) ;ADD NUMBER OF WORDS TO BE WRITTEN
5259 021746 005565 000160 ADC P.NWRT+2(R5)
5260 021752 005565 000162 ADC P.NWRT+4(R5)

```

```

5261 021756 000441 BR 18$
5262
5263 021760 112765 000131 000123 13$: MOVB #WRTCHK,P.RCMD(R5) ;ISSUE WRITE CHECK
5264 021766 052765 000200 000014 BIS #W.WCK,P.PRST(R5) ;SET NO WRITE FOR WRITE CHECK
5265 021774 062765 000001 000152 ADD #1,P.NODR(R5) ;INCREMENT NUMBER OF COMMAND ISSUED
5266 022002 005565 000154 ADC P.NODR+2(R5)
5267 022006 061665 000156 ADD (SP),P.NWRT(R5) ;ADD NUMBER OF WORDS TO BE WRITTEN
5268 022012 005565 000160 ADC P.NWRT+2(R5)
5269 022016 005565 000162 ADC P.NWRT+4(R5)
5270 022022 062665 000164 ADD (SP)+,P.NRD(R5) ;ADD NUMBER OF WORDS READ
5271 022026 005565 000166 ADC P.NRD+2(R5)
5272 022032 005565 000170 ADC P.NRD+4(R5)
5273 022036 000411 BR 18$
5274
5275 022040 112765 000121 000123 15$: MOVB #RDDATA,P.RCMD(R5) ;ISSUE READ DATA
5276 022046 062665 000164 ADD (SP)+,P.NRD(R5) ;ADD NUMBER OF WORDS READ
5277 022052 005565 000166 ADC P.NRD+2(R5)
5278 022056 005565 000170 ADC P.NRD+4(R5)
5279 022062 116565 000123 000001 18$: MOVB P.RCMD(R5),P.CMND(R5) ;LOAD COMMAND
5280 022070 012700 000100 MOV #100,R0 ;LOAD REGENERATE COUNT
5281
5282 022074 010503 20$: MOV R5,R3 ;LOAD INDEX
5283 022076 062703 000232 ADD #P.EXAR,R3 ;CALCULATE EXCLUDED AREA INDEX
5284 022102 012701 000005 MOV #5,R1 ;LOAD BAD TRACK COUNT
5285 022106 004737 055324 JSR PC,$RAND ;GENERATE RANDOM NUMBER
5286 022112 016546 000070 MOV P.MXCL(R5),-(SP) ;STORE MAXIMUM CYLINDER
5287 022116 166516 000066 SUB P.MNCL(R5),(SP) ;CALCULATE DIFFERENCE
5288 022122 001406 BEQ 21$ ;IF 0, SKIP MULTIPLY
5289 022124 005216 INC (SP) ;INCREMENT DIFFERENCE
5290 022126 013746 055424 MOV $LONUM,-(SP) ;PUT MULTICAND ON STACK
5291 022132 004737 055210 JSR PC,$MULT ;CALCULATE CYLINDER ADDRESS
5292 022136 005726 TST (SP)+ ;THROW AWAY LEASE SIGNIFICANT BITS
5293
5294 022140 066516 000066 21$: ADD P.MNCL(R5),(SP) ;CALCULATE CYLINDER
5295 022144 011665 000124 MOV (SP),P.RCYL(R5) ;STORE CYLINDER
5296 022150 116537 000073 001572 MOV P.MXTR(R5),TMPTRK ;STORE MAXIMUM TRACK
5297 022156 116537 000072 001574 MOV P.MNTR(R5),TMPTRK+2 ;STORE MINIMUM TRACK
5298 022164 163737 001574 001572 SUB TMPTRK+2,TMPTRK ;CALCULATE PERFERENCE
5299 022172 001415 BEQ 22$ ;IF=0, SKIP MULTIPLY
5300 022174 005046 CLR -(SP) ;MAKE ROOM ON STACK
5301 022176 113716 055423 MOV $HINUM+1,(SP) ;PUT IN MULTIPLIER ON STACK
5302 022202 013746 001572 MOV TMPTRK,-(SP) ;PUT TRACK NUMBER ON STACK
5303 022206 005216 INC (SP) ;INCREMENT DIFFERENCE
5304 022210 004737 055210 JSR PC,$MULT ;CALCULATE TRACK
5305 022214 116637 000001 001572 MOV 1(SP),TMPTRK ;STORE TRACK
5306 022222 062706 000004 ADD #4,SP ;ADJUST STACK
5307
5308 022226 063737 001574 001572 22$: ADD TMPTRK+2,TMPTRK ;CALCULATE TRACK
5309 022234 113765 001572 000127 MOV TMPTRK,P.RTRK(R5) ;STORE TRACK NUMBER
5310 022242 116537 000075 001576 MOV P.MXSC(R5),TMPSEC ;STORE MAXIMUM SECTOR
5311 022250 116537 000074 001600 MOV P.MNSC(R5),TMPSEC+2 ;STORE MINIMUM SECTOR
5312 022256 163737 001600 001576 SUB TMPSEC+2,TMPSEC ;CALCULATE DIFFERENCE
5313 022264 001415 BEQ 23$ ;IF=0, SKIP MULTIPLY
5314 022266 005046 CLR -(SP) ;MAKE ROOM ON STACK
5315 022270 113716 055422 MOV $HINUM,(SP) ;PUT IN MULTIPLIER
5316 022274 013746 001576 MOV TMPSEC,-(SP) ;PUT SECTOR NUMBER ON STACK

```

```

5317 022300 005216          INC      (SP)          ;INCREMENT DIFFERENCE
5318 022302 004737 055210    JSR      PC,$MULT     ;CALCULATE SECTOR
5319 022306 116637 000001 001576    MOVVB   1(SP),TMPSEC  ;STORE SECTOR
5320 022314 062706 000004          ADD      #4,SP        ;ADJUST STACK
5321
5322 022320 063737 001600 001576 23$:    ADD      TMPSEC+2,TMPSEC ;CALCULATE SECTOR
5323 022326 113765 001576 000126    MOVVB   TMPSEC,P.RSEC(R5) ;STORE SECTOR NUMBER
5324 022334 006316          ASL      (SP)         ;MULTIPLY CYLINDER BY 3
5325 022336 066516 000124    ADD      P.RCYL(R5),(SP)
5326 022342 063716 001572    ADD      TMPTRK,(SP)   ;ADD TRACK
5327 022346 012746 000026    MOV      #22,-(SP)    ;MULTIPLICAND - 22 SECTORS/TRACK
5328 022352 004737 055210    JSR      PC,$MULT     ;THROW AWAY MOST SIGNIFICANT BITS
5329 022356 012616          MOV      (SP)+,(SP)   ;ADD SECTOR
5330 022360 063716 001576    ADD      TMPSEC,(SP)  ;CONSTANT FOR BAD TRACKS
5331 022364 005216          INC      (SP)         ;ADD NUMBER OF SECTORS
5332 022366 061604          ADD      (SP),R4
5333 022370 004737 027252    JSR      PC,GETNUM    ;CHK IF DATA XFER EXCEEDS DISK
5334 022374 023704 027340    CMP      HOLD1,R4    ;CALCULATE NEW DISK ADDRESS
5335 022400 103421          BLO     40$
5336
5337 022402 005713          25$:    TST      (R3)        ;CHECK IF BAD TRACK INTERVAL
5338                                ;EXIST
5339 022404 001407          BEQ     30$          ;NO, ISSUE COMMAND
5340 022406 021623          CMP     (SP),(R3)+   ;CHECK IF IN BAD TRACK INTERVAL
5341 022410 103402          BLO     26$          ;NO, CHECK NEXT BAD TRACK 1
5342 022412 020413          CMP     R4,(R3)
5343 022414 103413          BLO     40$          ;YES, CALCULATE NEW DISK ADDRESS
5344 022416 005723          26$:    TST      (R3)+
5345 022420 005301          DEC     R1           ;DECREMENT BAD BLOCK COUNT
5346 022422 001367          BNE     25$          ;IF NOT 0 GET NEXT BAD BLOCK
5347
5348 022424 005726          30$:    TST      (SP)+      ;ADJUST STACK
5349 022426 016565 000124 000002    MOV     P.RCYL(R5),P.CYLN(R5) ;LOAD CYLINDER
5350 022434 016565 000126 000004    MOV     P.RSEC(R5),P.SECT(R5) ;LOAD TRACK AND SECTOR
5351 022442 000416          BR      45$          ;RETURN
5352
5353 022444 162604          40$:    SUB     (SP)+,R4    ;GET NUMBER OF SECTOR TRANSFERRED
5354 022446 005300          DEC     R0           ;DECREMENT REGENERATE COUNT
5355 022450 001402          BEQ     41$          ;DROP DRIVE CANNOT GENERATE
5356                                ;RANDOM DISK ADDRESS
5357 022452 000137 022074          JMP     20$          ;GET ANOTHER DISK ADDRESS
5358
5359 022456 013737 001452 177776 41$:    MOV     RKPRI,PS     ;LOCK OUT RK06 AND TTY INERRUPTS
5360 022464 104401 060705          TYPE   ,PAR020      ;TYPE 'UNABLE TO GENERATE NEW COMMAND'
5361 022470 004737 013162          JSR     PC,DROP      ;DROP DRIVE
5362 022474 005037 177776          CLR     PS           ;ALLOW RK06 INTERRUPTS
5363
5364 022500 012600          45$:    MOV     (SP)+,R0    ;RESTORE R0
5365 022502 012601          MOV     (SP)+,R1    ;RESTORE R1
5366 022504 012603          MOV     (SP)+,R3    ;RESTORE R3
5367 022506 012604          MOV     (SP)+,R4    ;RESTORE R4
5368 022510 012777 000100 156426    MOV     #IE,@$TKS   ;ALLOW TTY INTERRUPTS
5369 022516 000207          RTS     PC           ;RETURN

```

```
5370      .SBTTL  GET AVAILABLE BUFFER ROUTINE
5371
5372 022520 013746 177776      GETBUF: MOV     PS,-(SP)      :STORE PSW ON STACK
5373 022524 013737 001452 177776      MOV     RKPRI,PS      :LOCK OUT RK06 INTERRUPTS
5374 022532 013705 001320      MOV     BWAITQ,R5     :CHECK IF COMMAND WAITING FOR BUFFER
5375 022536 001424      BEQ     10$           :RETURN IF QUEUE IS EMPTY
5376 022540 004037 022616      1$:   JSR     R0,BUFFAL :ALLOCATE BUFFER
5377 022544 022610      10$   JSR     R0,Q.POP    :NOT ENOUGH MEMORY RETURN
5378 022546 004037 051644      JSR     R0,Q.POP     :GET NEXT PARMETER BLOCK FROM
5379 022552 001320      BWAITQ      : BUFFER WAIT QUEUE
5380 022554 005726      TST     (SP)+         :ADJUST STACK
5381 022556 005037 177776      CLR     PS            :ALLOW RK06 INTERRUPTS
5382 022562 004737 023234      JSR     PC,LDDATA    :LOAD BUFFER
5383 022566 004037 051564      JSR     R0,Q.PUSH    :PUT PARAMETER BLOCK ON
5384 022572 001324      CINITQ      : COMMAND INITIATION QUEUE
5385 022574 013737 001452 177776      MOV     RKPRI,PS      :LOCK OUT RK06 INTERRUPTS
5386 022602 013705 001320      MOV     BWAITQ,R5     :CHECK IF COMMAND WAITING FOR BUFFER
5387 022606 001354      BNE     1$           :YES, TRY TO ALLOCATE BUFFER
5388
5389 022610 012637 177776      10$:  MOV     (SP)+,PS    :RESTORE PSW
5390 022614 000207      RTS     PC            :RETURN
```

.SBTTL BUFFER ALLOCATION ROUTINE

```

*****
*
* THIS ROUTINE WILL TRY TO ALLOCATE A MAIN MEMORY BUFFER
* IF MEMORY IS AVAILABLE. (CALLED BY PRIORITY 7)
*
* REGISTER      USE
* -----      ---
*
* R1             WORD COUNT * 2
* R3             FREE BLOCK TABLE INDEX
* R4             FREE BLOCK TABLE COUNT
* R5             PARAMETER BLOCK ADDRESS
*
*CALL JSR      R0,BUFFAL
*      <ADDRESS OF BUFFER MEMORY NOT AVAILABLE ROUTINE>
*      RETURN <BUFFER ALLOCATED>
*
*****

```

5391
5392
5393
5394
5395
5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446

```

022616 010446
022620 010346
022622 010146
022624 016501 000012
022630 010165 000130
022634 005401
022636 006301
022640 013704 001670
022644 001420
022646 012703 001672
022652 022704 000011
022656 101001
022660 000000
022662 012346
022664 060116
022666 103411
022670 021316
022672 001423
022674 101010
022676 005723
022700 005726
022702 005304
022704 001366
022706 011000
022710 000437
022712 005726
022714 000774
022716 014365 000132
022722 011365 000010
022726 012613
022730 005720
022732 112765 177777 000145

```

```

BUFFAL: MOV      R4,-(SP)      :STORE R4 ON STACK
        MOV      R3,-(SP)      :STORE R3 ON STACK
        MOV      R1,-(SP)      :STORE R1 ON STACK
        MOV      P.WC(R5),R1    :STORE WORD COUNT
        MOV      R1,P.RWC(R5)  :STORE WORD COUNT
        NEG      R1             :GET ACTUAL WORD COUNT
        ASL      R1             :MULTIPLY WORD COUNT BY 2
        MOV      FBLKC,R4      :STORE FREE BLOCK TABLE ADDRESS
        BEQ      2$            :RETURN, IF NO BLOCKS AVAILABLE
        MOV      #FBLKT,R3     :LOAD ADDRESS OF FREE BLOCK TABLE
        CMP      #9.,R4        :HANG IF FREE BLOCK COUNT IS GREATER
        BHI     1$             : OR EQUAL TO 9
        HALT

1$:     MOV      (R3)+,-(SP)    :STORE BEGINNING OF FREE BLOCK
        ADD      R1,(SP)       :ADD WORD COUNT
        BCS     5$            :CHECK IF ADDRESS OVERFLOW
        CMP      (R3),(SP)     :CHECK IF ENOUGH ROOM
        BEQ     15$           :YES, ELIMINATE BLOCK
        BHI     10$           :YES, SHORTEN BLOCK
        TST     (R3)+         :ADJUST R3
        TST     (SP)+         :ADJUST STACK
        DEC     R4             :DECREMENT FREE BLOCK COUNT
        BNE     1$            :EXAMINE NEXT FREE BLOCK
2$:     MOV      (R0),R0       :LOAD RETURN ADDRESS
        BR      25$           :RETURN

5$:     TST     (SP)+         :ADJUST STACK
        BR      2$            :RETURN

10$:    MOV      -(R3),P.RBAL(R5) :LOAD BUS ADDRESS
        MOV      (R3),P.BALO(R5) :LOAD BUS ADDRESS
        MOV      (SP)+,(R3)     :LOAD NEW BASE
        TST     (R0)+         :ADJUST RETURN
        MOVB    #-1,P.BUFF(R5) :INDICATE THAT BUFFER HAS BEEN ASSIGNED

```

```
5447 022740 000423          BR      25$          :RETURN
5448
5449 022742 010301          15$:  MOV      R3,R1          :STORE WORD COUNT
5450 022744 014365 000132    MOV      -(R3),P.RBAL(R5)
5451 022750 011365 000010    MOV      (R3),P.BALO(R5) :LOAD BUS ADDRESS
5452 022754 005726          TST      (SP)+          :ADJUST STACK
5453 022756 005721          TST      (R1)+          :ADJUST R1
5454 022760 005720          TST      (R0)+          :ADJUST RETURN
5455 022762 112765 177777 000145  MOVB     #-1,P.BUFF(R5)  :INDICATE THAT BUFFER HAS BEEN ASSIGNED
5456 022770 005337 001670    DEC      FBLKC          :DECREMENT FREE BLOCK COUNT
5457 022774 005304          DEC      R4             :DECREMENT REMAIN BLOCK COUNT
5458 022776 001404          BEQ      25$           :IF 0, RETURN
5459
5460 023000 012123          20$:  MOV      (R1)+,(R3)+  :TRANSFER TWO WORDS
5461 023002 012123          MOV      (R1)+,(R3)+
5462 023004 005304          DEC      R4             :DECREMENT REMAINING BLOCKS TO BE ADJUSTED
5463 023006 001374          BNE     20$           :IF NOT 0, TRANSFER NEXT TWO WORDS
5464 023010 012601          25$:  MOV      (SP)+,R1       :RESTORE R1
5465 023012 012603          MOV      (SP)+,R3       :RESTORE R3
5466 023014 012604          MOV      (SP)+,R4       :RESTORE R4
5467 023016 000200          RTS      R0            :RETURN
5468
```


5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524

.SBTTL BUFFER RELEASE ROUTINE

```

:*****
:
: THIS ROUTINE WILL RELEASE BUFFER USED.
: (MUST BE CALLED IN PRIORITY 7)
:
: REGISTER      USE
: -----      ---
:
: R0             NEW END OF FREE BLOCK TABLE
: R1             LAST ADDRESS OF RELEASED BUFFER
: R3             FREE BLOCK INDEX
: R4             FREE BLOCK COUNT
: R5             PARAMETER BLOCK ADDRESS
: (SP)          BEGINNING ADDRESS OF RELEASED BLOCK
:
:CALL JSR      PC,BUFREL
: RETURN
:
:*****

```

```

BUFREL: MOV      R4,-(SP)      ;STORE R4 ON STACK
        MOV      R3,-(SP)      ;STORE R3 ON STACK
        MOV      R1,-(SP)      ;STORE R1 ON STACK
        MOV      R0,-(SP)      ;STORE R0 ON STACK
        TSTB     P,BUFF(R5)    ;CHECK IF BUFFER ASSIGNED
        BEQ      25$           ;NO, RETURN
        CLRB     P,BUFF(R5)    ;CLEAR BUFFER ALLOCATED
        MOV      P,RWC(R5),R1  ;STORE ALLOCATED WORD COUNT
        NEG      R1            ;GET ACTUAL WORD COUNT
        ASL      R1            ;MULTIPLY BY 2
        MOV      P,RBAL(R5),-(SP) ;STORE BUFFER ADDRESS
        ADD      (SP),R1       ;CALCULATE FINAL ADDRESS
        MOV      #FBLKT,R3    ;LOAD FREE BLOCK TABLE ADDRESS
        MOV      FBLKC,R4     ;STORE FREE BLOCK COUNT
        BEQ      8$           ;ADD FREE BLOCK

1$:     CMP      (R3)+,R1      ;COMPARE END OF BUFFER AND FREE BLOCK
        BEQ      10$          ;EQUAL MERGE BLOCKS
        BHI      5$           ;LESS THEN ADD FREE BLOCK
        CMP      (R3)+,(SP)   ;COMPARE BEGINNING OF BUFFER
        ; AND END OF FREE BLOCK
        BEQ      15$          ;EQUAL MERGE
        DEC      R4           ;DECREMENT FREE BLOCK COUNT
        BNE      1$           ;GO TO NEXT FREE BLOCK
        BR       8$           ;GO ADD BLOCK

5$:     MOV      R4,-(SP)      ;PUT REMAINING COUNT ON STACK
        ASL      (SP)         ;MULTIPLY BY 4
        ASL      (SP)
        TST      -(R3)        ;DECREMENT R3 BY 2
        ADD      (SP)+,R3     ;CALCULATE OLD END OF FREE BLOCK TABLE
        MOV      R5,R0
        ADD      #4,R0        ;CALCULATE NEW END OF FREE BLOCK TABLE
6$:     MOV      -(R3),-(R0)   ;MOVE TABLE

```

023020 010446
023022 010346
023024 010146
023026 010046
023030 105765 000145
023034 001472
023036 105065 000145
023042 016501 000130
023046 005401
023050 006301
023052 016546 000132
023056 061601
023060 012703 001672
023064 013704 001670
023070 001424

023072 022301
023074 001427
023076 101005
023100 022316

023102 001426
023104 005304
023106 001371
023110 000414

023112 010446
023114 006316
023116 006316
023120 005743
023122 062603
023124 010300
023126 062700 000004
023132 014340

5525	023134	014340		MOV	-(R3),-(R0)	
5526	023136	005304		DEC	R4	:DECREMENT FREE BLOCK COUNT
5527	023140	001374		BNE	6\$:IF NOT ZERO, COUNT
5528						
5529	023142	005237	001670	8\$: INC	FBLKC	:INCREMENT FREE BLOCK COUNT
5530	023146	012623		MOV	(SP)+,(R3)+	:STORE BEGINNING OF FREE BLOCK
5531	023150	010123		MOV	R1,(R3)+	:STORE END OF FREE BLOCK
5532	023152	000423		BR	25\$:RETURN
5533						
5534	023154	012643		10\$: MOV	(SP)+,-(R3)	:LOAD BEGINNING OF FREE BLOCK
5535	023156	000421		BR	25\$:RETURN
5536						
5537	023160	005726		15\$: TST	(SP)+	:ADJUST STACK
5538	023162	005304		DEC	R4	:DECREMENT FREE BLOCKS LEFT
5539	023164	001402		BEQ	16\$:NO MORE FREE BLOCK ADJUST END
5540	023166	021301		CMP	(R3),R1	:CHECK IF RELEASED BLOCK END=

5541					: FREE BLOCK BEGINNING
5542	023170	001402			: YES, MERGE BLOCKS
5543	023172	010143	16\$:	BEQ 20\$: LOAD END OF FREE BLOCK
5544	023174	000412		MOV R1, -(R3)	: RETURN
5545				BR 25\$	
5546	023176	010300	20\$:	MOV R3, R0	: STORE R3 FOR BUFFER MOVEMENT
5547	023200	005743		TST -(R3)	: ADJUST R3
5548	023202	005720		TST (R0)+	: ADJUST R0
5549	023204	006304		ASL R4	: MULTIPLY COUNT BY TWO
5550	023206	005304		DEC R4	: DECREMENT TRANSFER COUNT
5551	023210	012023	21\$:	MOV (R0)+, (R3)+	: MOVE TABLE ENTRY
5552	023212	005304		DEC R4	: DECREMENT COUNT
5553	023214	001375		BNE 21\$: IF NOT 0, CONTINUE MOVE
5554	023216	005337	001670	DEC FBLKC	: DECREMENT FREE BLOCK COUNT
5555					
5556	023222	012600	25\$:	MOV (SP)+, R0	: RESTORE R0
5557	023224	012601		MOV (SP)+, R1	: RESTORE R1
5558	023226	012603		MOV (SP)+, R3	: RESTORE R3
5559	023230	012604		MOV (SP)+, R4	: RESTORE R4
5560	023232	000207		RTS PC	: RETURN
5561					

5562
5563
5564
5565
5566
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617

.SBTTL LOAD DATA BUFFER

```

*****
*
* THE REGISTERS BE USED AS FOLLOWS:
*
* REGISTER          USE
* -----          ---
*
*      R5          ADDRESS OF PARAMETER BLOCK
*      R4          BUFFER ADDRESS
*      R3          PATTERN ADDRESS
*      R2          WORDS LEFT IN SECTOR
*      R1          WORD COUNT
*      R0          PATTERN COUNT
*
*CALL JSR      PC,LDDATA
*      RETURN
*
* THE DATA WILL LOOK AS FOLLOWS:
*
*-----*-----*-----*
* CPU ID          ! PAT          ! PAT          ! WORD 0
* NO. OF PAT. WORDS (N) ! NO. OF PAT. WORDS (N) ! WORD 1
*                                RANDOM DATA                                ! WORDS 2
*-----*-----*-----*                                ! TO N+1
*                                ZERO FILL                                ! WORDS N+2
*-----*-----*-----*                                ! TO 255
*****

```

```

LDDATA: MOV      R4,-(SP)          ;STORE R4 ON STACK
        MOV      R3,-(SP)          ;STORE R3 ON STACK
        MOV      R2,-(SP)          ;STORE R2 ON STACK
        MOV      R1,-(SP)          ;STORE R1 ON STACK
        MOV      R0,-(SP)          ;STORE R0 ON STACK
        MOV      P,RBAL(R5),R4     ;GET BUS ADDRESS
        MOV      P,RWC(R5),R1     ;STORE WORD COUNT
        NEG      R1                ;GET POSITIVE NUMBER FOR WORD COUNT
        CMPB    #RDATA,P.CMND(R5) ;CHECK IF READ DATA
        BNE     S$                ;NO, LOAD BUFFER
        CLR     (R4)+              ;CLEAR BUFFER BEFORE READ
        DEC     R1                ;CHECK IF DONE
        BNE     S$                ;NO, CONTINUE
        JMP     25$                ;RESTORE REGISTERS

        S$:    MOVB   P,DPAT(R5),R0 ;STORE PATTERN COUNT
        MOV     R0,-(SP)           ;STORE PATTERN FOR FIRST WORD
        ASL    (SP)                ;MULTIPLY PATTERN BY 4 FOR
        ASL    (SP)                ; REDUNACY
        ASL    (SP)
        ASL    (SP)
        BIS    R0,(SP)            ;STORE FIRST WORD OF SECTOR

```

```

023234 010446
023236 010346
023240 010246
023242 010146
023244 010046
023246 016504 000132
023252 016501 000130
023256 005401
023260 122765 000121 000001
023266 001005
023270 005024
023272 005301
023274 001375
023276 000137 023446

023302 116500 000121
023306 010046
023310 006316
023312 006316
023314 006316
023316 006316
023320 050016

```

5618	023322	113766	001364	000001		MOVB	CPUID,1(SP)	:	ON STACK
5619	023330	006300				ASL	R0	:	MULTIPLY DATA PATTERN BY 2
5620	023332	016003	004602			MOV	PATTAB(R0),R3	:	LOAD DATA PATTERN ADDRESS
5621	023336	012700	000040			MOV	#32.,R0	:	LOAD PATTERN COUNT
5622	023342	011624			11\$:	MOV	(SP),(R4)+	:	STORE PATTERN AND CPU ID IN 1ST WORD
5623	023344	162701	000002			SUB	#2,R1	:	DECREMENT WORD COUNT BY 2
5624	023350	022701	000376			CMP	#254.,R1	:	CHECK IF REST OF SECTOR WILL BE
5625								:	WRITTEN
5626	023354	101003				BHI	12\$:	NO, USE R1 AS WORD COUNT
5627	023356	012746	000376			MOV	#254.,-(SP)	:	STORE SECTOR COUNT = 254
5628	023362	000401				BR	13\$		
5629									
5630	023364	010146			12\$:	MOV	R1,-(SP)	:	STORE REMAINING WORD COUNT
5631	023366	111666	000001		13\$:	MOVB	(SP),1(SP)	:	ADD REDUNDACY
5632	023372	012624				MOV	(SP)+,(R4)+	:	STORE WORD COUNT IN 2ND WORD
5633	023374	012702	000376			MOV	#254.,R2	:	LOAD SECTOR WORD COUNT
5634									
5635	023400	012324			15\$:	MOV	(R3)+,(R4)+	:	STORE DATA PATTERN
5636	023402	005301				DEC	R1	:	DECREMENT WORD COUNT
5637	023404	001417				BEQ	20\$:	BRANCH IF FINISHED
5638	023406	005302				DEC	R2	:	DECREMENT SECTOR COUNT
5639	023410	005300				DEC	R0	:	DECREMENT PATTERN COUNT
5640	023412	001407				BEQ	17\$:	IF ZERO, INITIALIZE PATTERN
5641	023414	005702				TST	R2	:	ARE WE AT END OF SECTOR
5642	023416	001370				BNE	15\$:	NO, CONTINUE TO LOAD DATA PATTERN
5643	023420	162703	000074			SUB	#60.,R3	:	INITIALIZE PATTERN ADDRESS
5644	023424	012700	000040			MOV	#32.,R0	:	INITIALIZE PATTERN COUNT
5645	023430	000744				BR	11\$:	START NEXT SECTOR
5646									
5647	023432	162703	000100		17\$:	SUB	#64.,R3	:	INITIALIZE PATTERN ADDRESS
5648	023436	012700	000040			MOV	#32.,R0	:	INITIALIZE PATTERN
5649	023442	000756				BR	15\$:	REPEAT PATTERN
5650									
5651	023444	005726			20\$:	TST	(SP)+	:	ADJUST STACK
5652	023446	012600			25\$:	MOV	(SP)+,R0	:	RESTORE R0
5653	023450	012601				MOV	(SP)+,R1	:	RESTORE R1
5654	023452	012602				MOV	(SP)+,R2	:	RESTORE R2
5655	023454	012603				MOV	(SP)+,R3	:	RESTORE R3
5656	023456	012604				MOV	(SP)+,R4	:	RESTORE R4
5657	023460	000207				RTS	PC	:	RETURN

```
.SBTTL FIND FIRST DATA ERROR
5658
5659
5660 023462 032777 000002 155450  BUFER1: BIT      #SW1,@SWR      ;CHECK IF INHIBIT SOFTWARE DATA
5661                                     ; COMPARE
5662 023470 001017                                     ;YES, RETURN
5663 023472 013765 001362 000220  MOV      SOFCMP,P.CMLG(R5) ;LOAD COMPARISON LENGTH
5664 023500 001413                                     ;IF NO MORE COMPARISONS, RETURN
5665 023502 016565 000010 000214  MOV      P.BALO(R5),P.ERHD(R5) ;LOAD ADDRESS OF START OF SECTOR
5666 023510 016565 000012 000222  MOV      P.WC(R5),P.BFCP(R5) ;LOAD TRANSFER LENGTH
5667 023516 005465 000222                                     ;MAKE IT POSITIVE IT POSITIVE
5668 023522 004737 023540  JSR      PC,CMPBUF      ;DO COMPARISON
5669 023526 023534                                     ;ERROR RETURN
5670
5671 023530 005720          5$:   TST      (R0)+      ;ADJUST R0 FOR RETURN
5672 023532 000200          RTS      R0          ;RETURN
5673
5674 023534 011000          10$:  MOV      (R0),R0     ;LOAD ERROR RETURN
5675 023536 000200          RTS      R0          ;RETURN
```

5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731

.SBTTL COMPARE DATA BUFFER

```

:*****
:
: THE REGISTERS WILL BE USED AS FOLLOWS:
:
: REGISTER          USE
: -----          ---
:
: R5                ADDRESS OF PARAMETER BLOCK
: R4                BUFFER ADDRESS
: R3                PATTERN ADDRESS
: R2                SECTOR WORD COUNT
: R1                WORDS ON SECTOR WRITTEN
: R0                PATTERN COUNT
:
: *CALL JSR PC,CMPBUF
:        <ADDRESS OF COMPARISON ERROR RETURN>
:        RETURN
:
:        (SP) NUMBER OF SOFT DATA COMPARES
:        (SP+2) WORD COUNT
:*****

```

```

CMPBUF: MOV R4,-(SP)      ;STORE R4
        MOV R3,-(SP)      ;STORE R3
        MOV R2,-(SP)      ;STORE R2
        MOV R1,-(SP)      ;STORE R1
        MOV R0,-(SP)      ;STORE R0
        MOV P.ERHD(R5),R4 ;GET START ADDRESS
        MOV P.BFCP(R5),-(SP) ;GET COMPARE COUNT
        MOV P.CMLG(R5),-(SP) ;LOAD NUMBER OF COMPARISON
        BNE 10$           ;COMPARE IF COMPARE COUNT NOT ZERO
        JMP 40$           ;ELSE RETURN

10$: MOV R4,P.ERHD(R5)   ;LOAD SECTOR HEAD IN CASE OF ERROR
     CLR R2              ;CLEAR OFFSET FROM HEAD
     MOV (R4)+,R0        ;STORE 1ST WORD (PATTERN INDEX)
     MOV R0,R3
     BIC #177760,R0      ;KEEP LOW 4 BITS
     BIC #177417,R3      ;MASK CPU ID
     ASR R3              ;SHIFT 4 BITS RIGHT
     ASR R3
     ASR R3
     ASR R3
     CMP R0,R3           ;CHECK FOR PATTERN MATCH
     BNE 30$            ;NO, REPORT ERROR
     INC R2              ;INCREMENT OFFSET
     DEC 2(SP)          ;DECREMENT COMPARISON COUNT

13$: ASL R0              ;MULTIPLY INDEX BY 2
     MOV PATTAB(R0),R3  ;STORE PATTERN ADDRESS
     CMPB 1(R4),(R4)    ;CHECK WORD COUNT
     BNE 30$            ;NO, REPORT ERROR
     MOV (R4)+,R1       ;STORE WORDS IN SECTOR WRITTEN

```

023540 010446
023542 010346
023544 010246
023546 010146
023550 010046
023552 016504 000214
023556 016546 000222
023562 016546 000220
023566 001002
023570 000137 024024

023574 010465 000214
023600 005002
023602 012400
023604 010003
023606 042700 177760
023612 042703 177417
023616 006203
023620 006203
023622 006203
023624 006203
023626 020003
023630 001066
023632 005202
023634 005366 000002

023640 006300
023642 016003 004602
023646 126414 000001
023652 001055
023654 012401

5732	023656	042701	177400		BIC	#177400,R1	:KEEP LOW BYTE
5733	023662	005202			INC	R2	:INCREMENT OFFSET VALUE
5734	023664	005366	000002		DEC	2(SP)	:DECREMENT WORD COUNT
5735	023670	005765	000116		TST	P.SMPL(R5)	:CHECK IF SAMPLED COMPARE
5736	023674	001467			BEQ	50\$:YES, START SAMPLED COMPARE SEQUENCE
5737	023676	005701			TST	R1	:CHECK IF SECTOR ALL ZEROES
5738	023700	001426			BEQ	20\$:YES GO TO ZERO FILL LOOP
5739	023702	012700	000040		MOV	#32.,R0	:LOAD PATTERN COUNT
5740	023706	022324		15\$:	CMP	(R3)+,(R4)+	:CHECK PATTERN
5741	023710	001036			BNE	30\$:MISCOMPARE ERROR
5742	023712	005366	000002		DEC	2(SP)	:DECREMENT WORD COUNT
5743	023716	001442			BEQ	40\$:NO MORE COMPARISONS
5744	023720	005316			DEC	(SP)	:DECREMENT NUMBER OF SOFT
5745							: DATA COMPARE
5746	023722	001440			BEQ	40\$:NO MORE COMPARISONS
5747	023724	005202			INC	R2	:INCREMENT OFFSET
5748	023726	022702	000400		CMP	#256.,R2	:CHECK IF COMPLETE SECTOR COMPARES
5749	023732	001720			BEQ	10\$:YES, START NEXT SECTOR
5750	023734	005301			DEC	R1	:DECREMENT NUMBER OF WORDS WRITTEN
5751	023736	001407			BEQ	20\$:IF ZERO, GO TO ZERO FILL LOOP
5752	023740	005300			DEC	R0	:DECREMENT PATTERN COUNT
5753	023742	001361			BNE	15\$:NEXT WORD
5754	023744	012700	000040		MOV	#32.,R0	:REINITIALIZE PATTERN COUNT
5755	023750	162703	000100		SUB	#64.,R3	:REINITIALIZE PATTERN ADDRESS
5756	023754	000754			BR	15\$:CHECK NEXT WORD
5757							
5758	023756	005724		20\$:	TST	(R4)+	:CHECK ZERO FILL OF PATTERN
5759	023760	001012			BNE	30\$:MISCOMPARE ERROR
5760	023762	005366	000002		DEC	2(SP)	:DECREMENT WORD COUNT
5761	023766	001416			BEQ	40\$:NO MORE COMPARISONS
5762	023770	005316			DEC	(SP)	:DECREMENT NUMBER OF SOFT
5763							: DATA COMPARES
5764	023772	001414			BEQ	40\$:NO MORE COMPARISONS
5765	023774	005202			INC	R2	:INCREMENT OFFSET VALUE
5766	023776	022702	000400		CMP	#256.,R2	:CHECK IF COMPLETE SECTOR COMPARED
5767	024002	001674			BEQ	10\$:YES, START NEXT SECTOR
5768	024004	000764			BR	20\$:COMPARE NEXT WORD ON
5769							
5770	024006	110265	000217	30\$:	MOVB	R2,P.ERAD(R5)	:STORE OFFSET ADDRESS
5771	024012	022626			CMP	(SP)+,(SP)+	:ADJUST STACK
5772	024014	017666	000012	000012	MOV	@12(SP),12(SP)	:LOAD ERROR RETURN
5773	024022	000404			BR	45\$:RETURN
5774							
5775	024024	022626		40\$:	CMP	(SP)+,(SP)+	:ADJUST STACK
5776	024026	062766	000002	000012	ADD	#2,12(SP)	:ADJUST RETURN
5777							
5778	024034	105065	000117	45\$:	CLRB	P.ECMP(R5)	:CLEAR ECC COMPARE FLAG
5779	024040	012600			MOV	(SP)+,R0	:RESTORE R0
5780	024042	012601			MOV	(SP)+,R1	:RESTORE R1
5781	024044	012602			MOV	(SP)+,R2	:RESTORE R2
5782	024046	012603			MOV	(SP)+,R3	:RESTORE R3
5783	024050	012604			MOV	(SP)+,R4	:RESTORE R4
5784	024052	000207			RTS	PC	:RETURN
5785							
5786	024054	005701		50\$:	TST	R1	:CHECK IF ALL ZEROES
5787	024056	001432			BEQ	52\$:YES, TEST FOR ZEROES

5788	024060	021314			CMP	(R3),(R4)		:CHECK FIRST WORD OF PATERN
5789	024062	001351			BNE	30\$:MISCOMPARE, REPORT ERROR
5790	024064	026601	000002		CMP	2(SP),R1		:CHECK IF ENOUGH WORDS READ FOR
5791								: BOUNDARY COMPARISON
5792	024070	103443			BLO	55\$:NO, DO SAMPLE COMPARE
5793	024072	010146			MOV	R1,-(SP)		:SAVE SECTOR COUNT
5794	024074	005301			DEC	R1		:CALCULATE NUMBER OF WORDS FROM START
5795	024076	006301			ASL	R1		:CONVERT TO BYTES
5796	024100	060104			ADD	R1,R4		:CALCULATE MEMORY ADDRESS
5797	024102	042701	177700		BIC	#177700,R1		:GET PATTERN ADDRESS OFFSET
5798	024106	060103			ADD	R1,R3		:CALCULATE PATTERN ADDRESS
5799	024110	021324			CMP	(R3),(R4)+		:CHECK IF DATA WORD IS CORRECT
5800	024112	001403			BEQ	51\$:YES, CHECK ZERO FILL
5801	024114	005216			INC	(SP)		:CALCULATE OFFSET
5802	024116	012602			MOV	(SP)+,R2		:STORE OFFSET
5803	024120	000732			BR	30\$:REPORT ERROR
5804								
5805	024122	012602		51\$:	MOV	(SP)+,R2		:STORE OFFSET
5806	024124	062702	000002		ADD	#2,R2		:CALCULATE OFFSET FOR START OF ZERO FILL
5807	024130	022702	000377		CMP	#377,R2		:CHECK IF NO ZERO FILL
5808	024134	103421			BLO	55\$:YES, DO SAMPLE COMPARE
5809	024136	026602	000002		CMP	2(SP),R2		:CHECK WORD WAS READ
5810	024142	103416			BLO	55\$:NO, DO SAMPLE COMPARE
5811	024144	005714		52\$:	TST	(R4)		:CHECK START OF ZERO FILL
5812	024146	001317			BNE	30\$:NO, REPORT ERROR
5813	024150	022766	000377	000002	CMP	#377,2(SP)		:CHECK IF LAST WORD READ
5814	024156	101010			BHI	55\$:NO, DO SAMPLE COMPARE
5815	024160	012702	000377		MOV	#377,R2		:LOAD OFFSET COUNT
5816	024164	016504	000214		MOV	P.ERHD(R5),R4		:LOAD START OF SECTOR
5817	024170	062704	000776		ADD	#776,R4		:CALCULATE END OF SECTOR
5818	024174	005714			TST	(R4)		:CHECK LAST WORD OF SECTOR
5819	024176	001303			BNE	30\$:REPORT ERROR
5820								
5821	024200	016504	000214	55\$:	MOV	P.ERHD(R5),R4		:SET START OF SECTOR
5822	024204	012400			MOV	(R4)+,R0		:GET PATTERN INDEX
5823	024206	042700	177760		BIC	#177760,R0		:MASK OFF UNNECESSARY BIT
5824	024212	006300			ASL	R0		:MULTIPLY BY 2
5825	024214	016003	004602		MOV	PATTAB(R0),R3		:GET PATTERN ADDRESS
5826	024220	012401			MOV	(R4)+,R1		:GET SECTOR WORD COUNT
5827	024222	042701	177400		BIC	#177400,R1		:MASK OFF UNNECESSARY BITS
5828	024226	012702	000002		MOV	#2,R2		:INITIALIZE OFFSET
5829	024232	012700	000007		MOV	#7,R0		:LOAD NUMBER OF SAMPLES
5830	024236	162701	000043	57\$:	SUB	#35,,R1		:SUBTRACT 35 FROM SECTOR WORD COUNT
5831	024242	101430			BLOS	60\$:CHECK IF ZERO FILL
5832	024244	162766	000043	000002	SUB	#35,,2(SP)		:SUBTRACT 35 FROM WORD COUNT
5833	024252	101664			BLOS	40\$:CHECK IF FINISHED
5834	024254	062702	000043		ADD	#35,,R2		:CALCULATE OFFSET
5835	024260	062704	000106		ADD	#106,R4		:DETERMINE ADDRESS OF COMPARE
5836	024264	062703	000006		ADD	#6,R3		:DETERMINE ADDRESS OF PATTERN
5837	024270	021413			CMP	(R4),(R3)		:CHECK IF DATA CORRECT
5838	024272	001245			BNE	30\$:NO, REPORT ERROR
5839	024274	005316			DEC	(SP)		:DECREMENT NUMBER OF COMPARES
5840	024276	001652			BEQ	40\$:CHECK IF FINISHED
5841	024300	005300			DEC	R0		:CHECK IF IF TIME FOR NEXT SECTOR
5842	024302	001355			BNE	57\$:NO, CHECK NEXT SAMPLE
5843	024304	162766	000011	000002	SUB	#11,2(SP)		:ADJUST WORD COUNT

5844	024312	101644			BLOS	40\$:CHECK IF FINISHED
5845	024314	062704	000022		ADD	#22,R4	:CALCULATE START OF NEXT SECTOR
5846	024320	000137	023574		JMP	10\$:GO SERVICE NEXT SECTOR
5847							
5848	024324	162766	000043	000002	SUB	#35.,2(SP)	:SUBTRACT 35 FROM WORD COUNT
5849	024332	101634			BLOS	40\$:CHECK IF FINISHED
5850	024334	062702	000043		ADD	#35.,R2	:DETERMINE OFFSET
5851	024340	062704	000106		ADD	#106,R4	:DETERMINE ADDRESS OF COMPARE
5852	024344	005714			TST	(R4)	:CHECK IF DATA CORRECT
5853	024346	001217			BNE	30\$:NO, REPORT ERROR
5854	024350	005316			DEC	(SP)	:DECREMENT NUMBER OF COMPARES
5855	024352	001624			BEQ	40\$:BRANCH IF FINISHED
5856	024354	005300			DEC	R0	:CHECK IF SAMPLE COUNT EXHAUSTED
5857	024356	001362			BNE	60\$:NO, CHECK NEXT SAMPLE
5858	024360	162766	000011	000002	SUB	#11,2(SP)	:ADJUST WORD COUNT
5859	024366	101616			BLOS	40\$:CHECK IF FINISHED
5860	024370	062704	000022		ADD	#22,R4	:CALCULATE START OF NEXT SECTOR
5861	024374	000137	023574		JMP	10\$:GO SERVICE NEXT SECTOR

5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917

.SBTTL NORMAL COMMAND RETURN

```

*****
* THIS ROUTINE WILL HANDLE ALL NORMAL COMMAND TERMINATIONS
*
* CALLED ROUTINE          USE
* -----
*
* ASNORM                  ASSIGNMENT NORMAL TERNIMATION
* CNTRL1                  CONTROLLER ERROR
* ABNORM                  DRIVE ERROR
* EVCVY1                  NORMAL ERROR RECOVERY
* DATERR                  INCORRECT DATA BUFFER
*
* ASSUMED REGISTERS
*
* REGISTER          CONTENTS
* -----
*
* R5                ADDRESS OF PARMETER BLOCK
* R2                ADDRESS OF RK06 REGISTERS
*
*****

```

```

024400 004737 047542
024404 032765 024000 000016
024412 001010

024414 032765 177400 000020
024422 001004

024424 032765 131761 000034
024432 001405
024434 052737 000001 001564 2$:
024442 000137 027342

```

```

NORMAL: JSR PC,I.CSTS ;GATHER CONTROLLER STATUS
        BIT #SPAR!CTO,P.CS1(R5) ;CHECK IF DRIVE BUS PARITY OR CONTROLLER TIME OUT
        BNE 2$ ;YES, INDICATE CONTROLLER FAULT

        CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET IN CS2
        UNIT FIELD ERROR
        MULTIPLE DRIVE SELECT
        PROGRAMMING ERROR
        NON-EXISTENT MEMORY
        NON-EXISTENT DRIVE
        UNIBUS PARITY ERROR
        WRITE CHECK ERROR
        DATA LATE
        BIT #UFE!MDS!PGE!NEM!NED!UPE!WCE!DLT,P.CS2(R5)
        BNE 2$ ;YES, INIDICATE CONTROLLER ERROR

        CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET IN RKER
        ILLEGAL FUNTION CODE
        FORMAT ERROR
        DRIVE TYPE ERROR
        ECC HARD
        BAD SECTOR ERROR
        HEADER VRC ERROR
        CYLINDER ADDRESS OVERFLOW
        DRIVE TIMING ERROR
        OPERATION INCOMPLETE
        DATA CHECK
        BIT #ILC!FMTE!DTYE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,P.ER(R5)
        BEQ 3$ ;NO, CHECK FOR UNREPORT DRIVE ERRORS
        BIS #BITO,ERCONT ;SET CONTROLLER ERROR NOT FLAGGED
        JMP CNTRL1 ;REPORT CONTROLLER ERROR

```

```
5918
5919
5920
5921
5922
5923
5924
5925
5926 024446 032765 046016 000034 3$: BIT
5927 024454 001004 BNE
5928
5929
5930
5931
5932
5933 024456 032765 000070 000036 BIT
5934 024464 001405 BEQ
5935 024466 052765 000010 000212 4$: BIS
5936 024474 000137 025470 JMP
5937
5938 024500 032765 000001 000014 5$: BIT
5939 024506 001005 BNE
5940 024510 052765 000200 000212 BIS
5941 024516 000137 025470 JMP
5942
5943 024522 005737 001564 6$: TST
5944 024526 100017 BPL
5945 024530 122765 000121 000001 CMPB
5946 024536 101007 BHI
5947 024540 132765 000040 000001 BITB
5948 024546 001003 BNE
5949 024550 004037 025210 JSR
5950 024554 024564 10$ JSR
5951 024556 004037 051564 7$: JSR
5952 024562 001324 CINITQ
5953 024564 000207 10$: RTS
5954
5955 024566 005765 000150 11$: TST
5956 024572 001417 BEQ
5957 024574 004737 040440 JSR
5958 024600 005737 001566 20$: TST
5959 024604 001367 BNE
5960 024606 004037 051644 JSR
5961 024612 001330 ERRPRQ
5962 024614 012605 MOV
5963
5964 024616 001762 BEQ
5965 024620 010537 001566 MOV
5966 024624 004737 032160 JSR
5967 024630 000763 BR
5968
5969 024632 105765 000210 32$: TSTB
5970 024636 001402 BEQ
5971 024640 000137 013662 JMP
5972
5973 024644 122765 000117 000001 33$: CMPB
```

CHECK THE FOLLOWING ERROR BITS IN RKER(UNREPORTED DRIVE ERROR)
SEEK INCOMPLETE
NON-EXECUTABLE DRIVE FUNCTION
DRIVE DETECTED DRIVE BUS PARITY ERROR
INVALID DISK ADDRESS
WRITE LOCK ERROR
UNSAFE
#SKI!NXF!DRPAR!IDAE!WLE!UNS,P.ER(R5)
4\$;YES,SET UNREPORTED DRIVE ERROR

CHECK THE FOLLOWING ERROR BITS IN RKDS
AC LOW
SPEED LOSS
DRIVE OFF TRACK
#ACLO!SPDLSS!DROT,P.DS(R5)
5\$;NO, CHECK IF SPECIAL SEQUENCE
#BIT3,P.ERR(R5) ;SET DRIVE ERROR NOT REPORTED
ABNORM ;PROCESS ERROR
#DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
6\$;YES, CHECK IF SPECIAL SEQUENCE
#BIT7,P.ERR(R5) ;SET UNSOLITATED ATTENTION
ABNORM ;PROCESS ERROR
ERCONT ;CHECK IF CONTROLLER ERROR HAS OCCURRED
11\$;NO, CONTINUE NORMAL PROCESSING
#RDDATA,P.CMND(R5) ;CHECK IF DATA TRANSFER COMMAND
7\$;NO, REISSUE COMMAND
#BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
7\$;YES, REISSUE COMMAND
RO,CHKADD ;CHECK BUS AND DISK ADDRESS
10\$;ERROR RETURN
RO,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN
CINITQ ;COMMAND INITIATION QUEUE
RTS ;RETURN
P.DSTT(R5) ;CHECK IF RETRY SEQUENCE
32\$;NO, CHECK IF DRIVE IS BEING ASSIGNED
PC,ERCVY1 ;GO THROUGH ERROR RECOVERY
ERRPRO ;CHECK IF ERROR PROCESSING COMPLETE
10\$;NO, RETURN
RO,Q.POP ;GET NEXT ERROR TO BE PROCESSED
(SP)+,R5 ;LOAD ADDRESS OF PARAMETER BLOCK FOR
NEXT ERROR
10\$;IF NO MORE ERRORS, RETURN
R5,ERRPRO ;INDICATE THAT ERROR IS BEING PROCESSED
PC,ERCVY ;GO THROUGH ERROR RECOVERY
20\$;CHECK IF ERROR PROCESSING COMPLETE
P.ASSN(R5) ;CHECK IF DRIVE IS BEING ASSIGNED
33\$;NO, CHECK IF SEEK OR RELEASE
ASNORM ;GO CONTINUE ASSIGNMENT
#SEEK,P.CMND(R5) ;CHECK IF SEEK TO PREVIOUS POSITION

```

5974                                     : (SWITCH 14 OPTION)
5975 024652 001416 BEQ 34$ :YES, RELEASE DRIVE
5976 024654 122765 000140 000001 CMPB #RELEAS,P.CMND(R5) :CHECK IF RELEASE COMMAND
5977 024662 001021 BNE 35$ :NO, CHECK ADDRESS AND WORD COUNT
5978 024664 112737 177777 001506 MOVB #-1,I.ISRL :INDICATE THAT AN INTERRUPT HAS BEEN ISSUED
5979 024672 012762 000300 000000 MOV #INTR,RKCS1(R2) :FORCE AN INTERRUPT SCAN DRIVE ATTENTIONS
5980 024700 004037 051564 JSR RO,Q.PUSH :PUT PARAMTER BLOCK ADDRESS ON
5981 024704 001314 AVAILQ : DRIVE AVAILIABLE QUEUE
5982 024706 000207 RTS PC :RETURN
5983
5984 024710 112765 000140 000001 34$: MOVB #RELEAS,P.CMND(R5) :PUT RELEASE COMMAND IN PARAMETER BLOCK
5985 024716 004037 051564 JSR RO,Q.PUSH :ENQUEUE PARAMETER BLOCK IN
5986 024722 001324 CINITQ : COMMAND INITIATION QUEUE
5987 024724 000207 RTS PC :RETURN
5988
5989 024726 004037 025210 35$: JSR RO,CHKADD :CHECK WORD COUNT, BUS ADDRESS,
5990 : SECTOR, TRACK, AND CYLINDER
5991 :ERROR RETURN ADDRESS
5992 024732 024564 10$ MOVB #-1,I.ISRL :INDICATE THAT AN INTERRUPT HAS BEEN ISSUED
5993 024734 112737 177777 001506 MOV #INTR,RKCS1(R2) :GO SCAN FOR DRIVE INTERRUPTS
5994 024742 012762 000300 000000 CLR PS :ALLOW RK06 INTERRUPTS
5995 024750 005037 177776 CMPB #RDDATA,P.CMND(R5) :CHECK IF READ DATA
5996 024754 122765 000121 000001 BNE NORM1 :NO, DO NOT COMPARE BUFFER
5997 024762 001004 JSR RO,BUFER1 :CHECK DATA READ
5998 024764 004037 023462 DATERR :ERROR RETURN
5999 024770 025602 BR NORM2 :RELEASE BUFFER
6000 024772 000417
6001 024774 122765 000131 000001 NORM1: CMPB #WRTCHK,P.CMND(R5) :CHECK IF WRITE COMMAND
6002 025002 001013 BNE NORM2 :NO, RELEASE BUFFER
6003 025004 032765 000200 000014 BIT #W.WCK,P.PRST(R5) :CHECK IF WRITE CHECK ISSUED
6004 025012 001407 BEQ NORM2 :YES, RELEASE BUFFER
6005 025014 042765 000200 000014 BIC #W.WCK,P.PRST(R5) :CLEAR WRITE FLAG
6006 025022 004037 051564 JSR RO,Q.PUSH :PUT PARAMETER BLOCK IN
6007 025026 001324 CINITQ : COMMAND INITIATION QUEUE
6008 025030 000207 RTS PC :RETURN
6009
6010 025032 013737 001452 177776 NORM2: MOV RKPRI,PS :LOCK OUT RK06 INTERRUPTS
6011 025040 004737 023020 JSR PC,BUFREL :RELEASE BUFFER
6012 025044 005037 177776 CLR PS :ALLOW RK06 INTERRUPTS
6013 025050 032777 040040 154062 BIT #SW5!SW14,@SWR :CHECK IF INHIBIT AUTOMATIC DESELECT
6014 025056 001043 BNE 33$ :YES, PUT PARAMETER BLOCK IN
6015 : AVAILIABLE QUEUE
6016 025060 026565 000110 000154 CMP P.MXCD+2(R5),P.NODR+2(R5) :CHECK IF MAXIMUM NUMBER
6017 : OF COMMAND HAVE BEEN ISSUED
6018 025066 101010 BHI 29$ :NO, CHECK IF MAXIMUM NUMBER OF
6019 : OF WORDS HAVE BEEN TRANSFERRED
6020 025070 103404 BLO 28$ :YES, DROP DRIVE
6021 025072 026565 000106 000152 CMP P.MXCD(R5),P.NODR(R5) :CHECK IF MAXIMUM NUMBER
6022 : OF COMMANDS HAVE BEEN ISSUED
6023 025100 103003 BHIS 29$ :NO, CHECK IF MAXIMUM NUMBER OF
6024 : HAVE BEEN TRANSFERRED
6025 025102 004737 013162 28$: JSR PC,DROP :DROP DRIVE
6026 025106 000435 BR 35$ :ALLOCATE BUFFER
6027
6028
6029 025110 016546 000166 29$: MOV P.NRD+2(R5),-(SP) :STORE NUMBER OF WORDS READ ON STACK

```

6030	025114	016546	000170			MOV	P.NRD+4(R5),-(SP)	
6031	025120	066566	000160	000002		ADD	P.NWRT+2(R5),2(SP)	:ADD NUMBER OF WORDS WRITTEN
6032	025126	005516				ADC	(SP)	
6033	025130	066516	000162			ADD	P.NWRT+4(R5),(SP)	
6034	025134	103410				BCS	30\$:DROP DRIVE IF OVERFLOW
6035	025136	026526	000114			CMP	P.MXWT+2(R5),(SP)+	:CHECK IF MAXIMUM NUMBER OF WORDS :HAVE BEEN TRANSFERRED
6036								:NO, PUT PARAMETER BLOCK IN :AVAILABLE QUEUE
6037	025142	101010				BHI	32\$	
6038								:YES, DROP DRIVE
6039	025144	103405				BLO	31\$	
6040	025146	026526	000112			CMP	P.MXWT(R5),(SP)+	:CHECK IF MAXIMUM NUMBER OF WORDS :HAVE BEEN TRANSFERRED
6041								:YES, DROP DRIVE
6042	025152	103753				BLO	28\$	
6043	025154	000404				BR	33\$:NO, PUT PARAMETER BLOCK IN :AVAILABLE QUEUE
6044								
6045								
6046	025156	005726			30\$:	TST	(SP)+	:ADJUST STACK
6047	025160	005726			31\$:	TST	(SP)+	:ADJUST STACK
6048	025162	000747				BR	28\$:DROP DRIVE
6049								
6050	025164	005726			32\$:	TST	(SP)+	:ADJUST STACK
6051	025166	112765	000140	000001	33\$:	MOVB	#RELEAS,P.CMND(R5)	:PUT RELEASE COMMAND IN PARAMETER BLOCK
6052	025174	004037	051564			JSR	R0,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN
6053	025200	001324				CINITQ		:COMMAND INITIATION QUEUE
6054								
6055	025202	004737	022520		35\$:	JSR	PC.GETBUF	:GET NEW BUFFER
6056	025206	000207				RTS	PC	:RETURN

```

6057      .SBTTL CHECK BUS ADDRESS, WORD COUNT, SECTOR, TRACK, AND CYLINDER
6058
6059 025210 005765 000022      CHKADD: TST      P.WCR(R5)      :CHECK IF WORD COUNT ZERO
6060 025214 001404              BEQ      6$      :YES, CHECK BUS ADDRESS
6061 025216 052737 000002 0C1564      BIS      #BIT1,ERCONT :SET WORD COUNT NOT EQUAL ZERO
6062 025224 000512              BR      24$      :REPORT ERROR
6063
6064 025226 032765 001400 000016 6$:      BIT      #BA16!BA17,P.CS1(R5) :CHECK IF HIGH BUS ADDRESS BITS SET
6065 025234 001011              BNE      10$      :YES, REPORT ERROR
6066 025236 016546 000012      MOV      P.WC(R5),-(SP) :STORE WORD COUNT
6067 025242 005416              NEG      (SP)      :MAKE IT POSITIVE
6068 025244 006316              ASL      (SP)      :DETERMINE NUMBER OF BITS
6069 025246 066516 000010      ADD      P.BALO(R5),(SP) :DETERMINE EXPECTED BUS ADDRESS
6070 025252 026526 000024      CMP      P.BAR(R5),(SP)+ :CHECK BUS ADDRESS
6071 025256 001404              BEQ      17$      :BUS ADDRESS CORRECT--CHECK CYLINDER,
6072              :TRACK, AND SECTOR
6073 025260 052737 000004 001564 10$:      BIS      #BIT2,ERCONT :SET BUS ADDRESS INCORRECT
6074 025266 000471              BR      24$      :REPORT ERROR
6075
6076 025270 016546 000012      17$:      MOV      P.WC(R5),-(SP) :STORE WORD COUNT
6077 025274 005416              NEG      (SP)      :MAKE IT POSITIVE
6078 025276 105716              TSTB     (SP)      :CHECK IF EVEN NUMBER OF SECTORS
6079 025300 001402              BEQ      18$      :YES, DO COMPARISON
6080 025302 105266 000001      INCB     1(SP)      :INCREMENT SECTOR COUNT
6081 025306 005046              18$:      CLR      -(SP)      :MAKE ROOM ON STACK
6082 025310 116616 000003      MOVB     3(SP),(SP) :STORE NUMBER OF SECTORS TRANSFERRED
6083 025314 005066 000002      CLR      2(SP)      :CLEAR LOCATION ON STACK
6084 025320 116566 000004 000002      MOVB     P.SECT(R5),2(SP) :STORE STARTING SECTOR
6085 025326 066616 000002      ADD      2(SP),(SP) :DETERMINE FINAL SECTOR ADDRESS
6086 025332 005066 000002      CLR      2(SP)      :CLEAR NUMBER OF TRACKS TRANSFERRED
6087
6088 025336 022716 000026      19$:      CMP      #22,(SP) :CHECK FOR SECTOR OVERFLOW
6089 025342 101005              BHI      20$      :NO, CHECK IF SECTOR CORRECT
6090 025344 162716 000026      SUB      #22,(SP) :DECREMENT SECTOR COUNT BY 22 AND
6091 025350 005266 000002      INC      2(SP)      : INCREMENT TRACKS TRANSFERRED
6092 025354 000770              BR      19$      :CHECK FOR SECTOR OVERFLOW
6093
6094 025356 126526 000026      20$:      CMPB     P.DTS(R5),(SP)+ :CHECK IF FINAL SECTOR CORRECT
6095 025362 001027              BNE      23$      :NO, REPORT ERROR
6096 025364 005046              CLR      -(SP)      :MAKE ROOM FOR TRACKS TRANSFERRED
6097 025366 116516 000005      MOVB     P.TRCK(R5),(SP) :STORE STARTING TRACK
6098 025372 066616 000002      ADD      2(SP),(SP) :DETERMINE FINAL TRACK ADDRESS
6099 025376 005066 000002      CLR      2(SP)      :CLEAR FINAL CYLINDER
6100
6101 025402 122716 000003      21$:      CMPB     #3,(SP) :CHECK FOR TRACK OVERFLOW
6102 025406 101005              BHI      22$      :NO, CHECK FINAL TRACK
6103 025410 162716 000003      SUB      #3,(SP) :DECREMENT TRACK COUNT BY 3 AND
6104 025414 005266 000002      INC      2(SP)      : INCREMENT CYLINDER COUNT
6105 025420 000770              BR      21$      :CHECK FOR TRACK OVERFLOW
6106
6107 025422 126526 000027      22$:      CMPB     P.DTS+1(R5),(SP)+ :CHECK IF FINAL TRACK CORRECT
6108 025426 001005              BNE      23$      :TRACK INCORRECT ERROR
6109 025430 066516 000002      ADD      P.CYLN(R5),(SP) :CALCULATE FINAL CYLINDER
6110 025434 026516 000030      CMP      P.DCYL(R5),(SP) :CHECK IF CYLINDER CORRECT
6111 025440 001410              BEQ      25$      :CORRECT GO TO NORMAL RETURN
6112 025442 005726              23$:      TST      (SP)+      :ADJUST STACK

```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 PAGE 137
CHECK BUS ADDRESS, WORD COUNT, SECTOR, TRACK, AND CYLINDER

SEQ 0134

6113	025444	052737	000010	001564		BIS	#BIT3,ERCONT	:SET INCORRECT SECTOR, TRACK, OR CYLINDER
6114	025452	004737	027342		24\$:	JSR	PC,CNTRL1	:REPORT ERROR
6115	025456	011000				MOV	(R0),R0	:LOAD R0 FOR ERROR RETURN
6116	025460	000200				RTS	R0	:RETURN
6117								
6118	025462	005726			25\$:	TST	(SP)+	:ADJUST STACK
6119	025464	005720				TST	(R0)+	:ADJUST R0
6120	025466	000200				RTS	R0	:NORMAL RETURN


```
.SBTTL ABNORMAL TERMINATION
6121
6122
6123 025470 004737 052004 ABNORM: JSR PC,Q.RMOV ;REMOVE PARAMETER BLOCK FROM COMMAND
6124 ; INITIATION QUEUE
6125 025474 032765 000100 000014 BIT #CMDTO,P.PRST(R5) ;CHECK IF COMMAND TIME OUT
6126 025502 001402 BEQ 3$ ;NO, PROCESS ABNORMAL TERMINATION
6127 025504 000137 025674 JMP TIMOUT ;JUMP TO TIME OUT ROUTINE
6128
6129 025510 005737 001564 3$: TST ERCONT ;CHECK IF CONTROLLER ERROR HAS OCCURRED
6130 025514 100010 BPL 10$ ;NO, CONTINUE ERROR PROCESSING
6131 025516 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK FOR UNSOLICATED INTERRUPT
6132 025524 001403 BEQ 5$ ;YES, DO NOT ENQUEUE PARAMETER BLOCK
6133 025526 004037 051564 JSR R0,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN
6134 025532 001324 CINITQ ; COMMAND INITIATION QUEUE
6135 025534 000207 5$: RTS PC ;RETURN
6136
6137 025536 032765 000001 000014 10$: BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
6138 025544 001004 BNE ABN000 ;YES, CHECK IF DRIVE BEING ASSIGNED
6139 025546 052765 000200 000212 BIS #BIT7,P.ERR(R5) ;INDICATE UNSOLICATED INTERRUPT
6140 025554 000420 BR ABN001 ;CHECK IF ERROR IS BEING PROCESSED
6141
6142 025556 005765 000150 ABN000: TST P.DSTT(R5) ;CHECK IF CURRENTLY UNDERGOING
6143 ; ERROR PROCESSING
6144 025562 001415 BEQ ABN001 ;NO, CHECK IF ERROR PRESENTLY BEING PROCESSED
6145 025564 100402 BMI 13$ ;ERROR ENQUEUED
6146 025566 000137 043236 JMP ERVCY2 ;ERROR RECOVERY ROUTINE
6147
6148 025572 052765 000100 000212 13$: BIS #BIT6,P.ERR(R5) ;SET ERROR WHILE ENQUEUED IN ERROR QUEUE
6149 025600 000207 RTS PC ;RETURN
6150
6151 025602 013737 001452 177776 DATERR: MOV RKPRI,PS ;LOCK RK06 INTERRUPTS
6152 025610 052765 000004 000212 BIS #BIT2,P.ERR(R5) ;SET SOFTWARE DETECTED DATA ERROR
6153
6154 025616 012765 100200 000150 ABN001: MOV #BIT15!BIT7,P.DSTT(R5) ;SET ERROR ENQUEUED AND FIRST ERROR
6155 025624 005737 001566 TST ERRPRO ;CHECK IF ERROR IS BEING PROCESSED
6156 025630 001015 BNE 5$ ;YES, GO ENQUEUE ERROR
6157 025632 010537 001566 2$: MOV R5,ERRPRO ;LOAD ERROR BEING PROCESSED
6158 025636 004737 032160 JSR PC,ERCVY ;START ERROR RECOVERY
6159 025642 005737 001566 TST ERRPRO ;CHECK IF ERROR PROCESSING THROUGH
6160 025646 001005 BNE 3$ ;NO, RETURN
6161 025650 004037 051644 JSR R0,Q.POP ;GET NEXT ERROR ENQUEUED
6162 025654 001330 ERRPRQ
6163 025656 012605 MOV (SP)+,R5 ;LOAD R5 FOR ERROR PROCESSING
6164 025660 001364 BNE 2$ ;IF NOT ZERO PROCESS NEXT ERROR
6165 025662 000207 3$: RTS PC ;RETURN
6166
6167 025664 004037 051564 5$: JSR R0,Q.PUSH ;PUT PARAMETER BLOCK ON
6168 025670 001330 ERRPRQ ; ERROR PROCESSING QUEUE
6169 025672 000207 RTS PC ;RETURN
```

```
.SBTTL COMMAND TIME OUT ROUTINE
TIMOUT: MOV RKCS2(R2),-(SP) ;GET CS2 FOR DRIVE NUMBER
        BIC #^C<DRVMSK>,(SP) ;KEEP DRIVE NUMBER
        CMPB (SP)+,P.DRVN(R5) ;CHECK IF TIME OUT IF FOR
        ; CURRENTLY SELECTED DRIVE
        BNE 5$ ;NO, TEST IF LOOPING ON CONTROLLER ERROR
        MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REGISTER 1
        BIT #GO,T.CS1 ;CHECK IF GO IS STILL SET
        BEQ 5$ ;NO, TEST IF LOOPING ON CONTROLLER ERROR
        BIS #E.CMTO,E.CONT ;SET CONTROLLER TIMED OUT ON COMMAND
        JMP CONTRL ;GO REPORT CONTROLLER ERROR
5$: TST ERCONT ;CHECK IF CYCLING ON CONTROLLER ERROR
    BPL 10$ ;NO, CHECK IF DRIVE IS CURRENTLY
    ; SEIZED BY OTHER PORT
    JSR R0,Q.PUSH ;REQUEUE COMMAND IN COMMAND
    CINITQ ; INITIATION QUEUE
    RTS PC ;RETURN
10$: BIT #DRVSZD,P.PRST(R5) ;CHECK IF DRIVE WAS SEIZED
    BEQ 12$ ;NO, CONTINUE
    BIS #BIT4,P.ERR(R5) ;SET TIME OUT BECAUSE DRIVE SEIZED
    BR ABN000 ;GO INITIATE ERROR PROCESSING
12$: BIT #E.UNLD,P.PRST(R5) ;CHECK IF WAITING FOR HEADS TO RELOAD
    BEQ 15$ ;NO, SET TIME OUT FOR POSITIONING
    BIS #BIT8,P.ERR(R5) ;SET TIME OUT IN RELOADING HEADS
    BR ABN000 ;GO INITIATE ERROR PROCESSING
15$: BIS #BIT5,P.ERR(R5) ;SET TIME OUT DURING POSITIONING
    BR ABN000 ;GO INITIATE ERROR PROCESSING
```

6202
6203
6204 026030 010346
6205 026032 010146
6206 026034 012703 001410
6207 026040 012701 001430
6208 026044 104401 065006
6209 026050 004737 026310
6210 026054 012701 001444
6211 026060 104401 065105
6212 026064 004737 026310
6213 026070 012601
6214 026072 012603
6215 026074 000207
6216
6217
6218
6219 026076 010346
6220 026100 104401 065165
6221 026104 016546 000036
6222 026110 004737 053224
6223 026114 104401 065174
6224 026120 010503
6225 026122 062703 000040
6226 026126 012346
6227 026130 004737 053224
6228 026134 104401 061126
6229 026140 012346
6230 026142 004737 053224
6231 026146 104401 065235
6232 026152 012346
6233 026154 004737 053224
6234 026160 104401 061126
6235 026164 012346
6236 026166 004737 053224
6237 026172 104401 065250
6238 026176 012346
6239 026200 004737 053224
6240 026204 104401 061126
6241 026210 012346
6242 026212 004737 053224
6243 026216 104401 065263
6244 026222 012346
6245 026224 004737 053224
6246 026230 104401 061126
6247 026234 012346
6248 026236 004737 053224
6249 026242 104401 001165
6250 026246 012603
6251 026250 000207

.SBTTL PRINT RK06 UNIBUS REGISTERS

PRIREG: MOV R3,-(SP) ;STORE R3 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV #T.CS1,R3 ;LOAD BEGINNING OF REGISTER STORAGE
MOV #T.DS,R1 ;LOAD END OF REGISTER STORAGE
TYPE ,ERR300 ;PRINT REGISTER NAMES
JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
MOV #T.DB,R1 ;LOAD END OF REGISTER STORAGE
TYPE ,ERR301 ;PRINT REGISTER NAMES
JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
MOV (SP)+,R1 ;RESTORE R1
MOV (SP)+,R3 ;RESTORE R3
RTS PC ;RETURN

.SBTTL PRINT DRIVE STATUS

PRDSTT: MOV R3,-(SP) ;STORE R3 ON STACK
TYPE ,ERR302 ;PRINT REGISTER NAMES
MOV P.DS(R5),-(SP) ;CONVERT DRIVE STATUS REG TO OCTAL
JSR PC,BINOCT
TYPE ,ERR303 ;PRINT HEADER
MOV R5,R3 ;DETERMINE START OF PRINT OUT
ADD #P.A00,R3
MOV (R3)+,-(SP) ;PRINT MESSAGE A
JSR PC,BINOCT
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,ERR304 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,ERR305 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,ERR306 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCT
TYPE ,\$CRLF
MOV (SP)+,R3 ;RESTORE R3
RTS PC ;RETURN

6252
6253
6254 026252 010346
6255 026254 010146
6256 026256 010503
6257 026260 010501
6258 026262 062703 000016
6259 026266 062701 000036
6260 026272 104401 065006
6261 026276 004737 026310
6262 026302 012601
6263 026304 012603
6264 026306 000207
6265
6266
6267
6268 026310 012346
6269 026312 004737 053224
6270 026316 020103
6271 026320 001403
6272 026322 104401 061126
6273 026326 000770
6274
6275 026330 104401 001165
6276 026334 000207

.SBTTL PRINT CONTROLLER STATUS

PRISTT: MOV R3,-(SP) ;STORE R3 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV R5,R3 ;STORE PARAMETER BLOCK ADDRESS
MOV R5,R1 ; FOR INDEX CALCULATIONS
ADD #P.CS1,R3 ;CALCULATE BEGINNING OF OUTPUT OCTAL
ADD #P.DS,R1 ;CALCULATE END OF OUTPUT OCTAL
TYPE ,ERR300 ;PRINT REGISTER NAMES
JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
MOV (SP)+,R1 ;RESTORE R1
MOV (SP)+,R3 ;RESTORE R3
RTS PC ;RETURN

.SBTTL PRINT REGISTER CONTENTS

PRSTAT: MOV (R3)+,-(SP) ;SAVE REGISTER VALUE FOR TYPE OUT
JSR PC,BINOCT ;CONVERT TO OCTAL
CMP R1,R3 ;CHECK IF FINISHED
BEQ 10\$;YES, RETURN
TYPE ,BLANKS ;TYPE 2 BLANKS
BR PRSTAT ;TYPE NEXT REGISTER

10\$: TYPE ,\$CRLF ;TYPE <CR><LF>
RTS PC ;RETURN

```

6277      .SBTTL  ECC CORRECTION ROUTINE
6278
6279 026336 010446      ECCCOR: MOV    R4,-(SP)      ;STORE R4 ON STACK
6280 026340 010346      MOV    R3,-(SP)      ;STORE R3 ON STACK
6281 026342 010146      MOV    R1,-(SP)      ;STORE R1 ON STACK
6282 026344 010046      MOV    R0,-(SP)      ;STORE R0 ON STACK
6283 026346 013703 001634  MOV    RTYBA,R3      ;LOAD BEGINNING OF SECTOR ADDRESS
6284 026352 013704 001636  MOV    RTYWC,R4      ;LOAD WORDS READ FROM SECTOR
6285 026356 006304      ASL    R4             ;MAKE IT BYTES
6286 026360 010346      MOV    R3,-(SP)      ;STORE ADDRESS OF START OF SECTOR
6287 026362 060416      ADD    R4,(SP)       ;CALCULATE END OF SECTOR
6288 026364 016501 000060  MOV    P.EPOS(R5),R1 ;SAVE POSITION INFORMATION
6289 026370 005301      DEC    R1             ;DECREMENT BIT COUNT BY ONE TO GET
6290                                     ; BIT ADDRESS
6291 026372 010100      MOV    R1,R0
6292 026374 006201      ASR    R1             ;DETERMINE WORDS THAT ARE BAD
6293 026376 006201      ASR    R1             ; FOR DATA READ (SHIFT 3 BITS
6294 026400 006201      ASR    R1             ; RIGHT)
6295 026402 042701 000001  BIC    #1,R1         ;CLEAR BYTE INDICATOR
6296 026406 020104      CMP    R1,R4         ;CHECK IF ERROR WITHIN TRANSFER
6297 026410 103037      BHIS   10$          ;NO, RETURN
6298 026412 060103      ADD    R1,R3         ;DETERMINE ADDRESS OF ERROR
6299 026414 016537 000062 001622  MOV    P.EPAT(R5),ECCPAT ;LOAD PATTERN
6300 026422 005037 001624  CLR    ECCPAT+2
6301 026426 042700 177760  BIC    #177760,R0    ;DETERMINE BIT POSITION FOR START OF ECC
6302                                     ; CORRECTION
6303 026432 001406      BEQ    5$           ;CORRECTION STARTS ON
6304                                     ; WORD BOUNDARY
6305
6306 026434 006337 001622  3$:   ASL    ECCPAT      ;SHIFT PATTERN 1 BIT LEFT
6307 026440 006137 001624  ROL    ECCPAT+2
6308 026444 005300      DEC    R0           ;DECREMENT COUNT
6309 026446 001372      BNE    3$          ;CHECK IF IN POSITION
6310
6311 026450 011300      5$:   MOV    (R3),R0     ;CORRECT FIRST WORD WITH EXCLUSIVE
6312 026452 013701 001622  MOV    ECCPAT,R1    ; OF BITS IN ERROR
6313 026456 043713 001622  BIC    ECCPAT,(R3)
6314 026462 040001      BIC    R0,R1
6315 026464 050123      BIS    R1,(R3)+
6316 026466 021603      CMP    (SP),R3     ;CHECK IF SECOND WORD IN TRANSFER
6317 026470 001407      BEQ    10$         ;NO, RETURN
6318 026472 011300      MOV    (R3),R0     ;CORRECT SECOND WORD WITH EXCLUSIVE
6319 026474 013701 001624  MOV    ECCPAT+2,R1  ; OF BITS IN ERROR
6320 026500 043713 001624  BIC    ECCPAT+2,(R3)
6321 026504 040001      BIC    R0,R1
6322 026506 050113      BIS    R1,(R3)
6323
6324 026510 005726      10$:  TST    (SP)+        ;ADJUST STACK
6325 026512 012600      MOV    (SP)+,R0    ;RESTORE R0
6326 026514 012601      MOV    (SP)+,R1    ;RESTORE R1
6327 026516 012603      MOV    (SP)+,R3    ;RESTORE R3
6328 026520 012604      MOV    (SP)+,R4    ;RESTORE R4
6329 026522 000207      RTS    PC          ;RETURN

```

```

6330          .SBTTL CONTROLLER ERROR DETECTED BY RK06 DRIVER
6331
6332 026524 010446          CONTRL: MOV    R4,-(SP)          ;STORE R4 ON STACK
6333 026526 004737 047414  JSR    PC,I.STOR        ;GET CONTROLLER STATUS
6334 026532 032777 002000 152400  BIT    #SW10,@SWR        ;CHECK IF BELL ON ERROR
6335 026540 001402          BEQ    1$                ;NO, CONTINUE
6336 026542 104401 001160  TYPE   ,SBELL            ;RING BELL
6337 026546 013704 001462  1$:   MOV    E.CONT,R4      ;STORE CONTROLLER INDICATORS
6338 026552 032704 100000  BIT    #E.MDS,R4        ;CHECK IF DUPLICATE DRIVE SELECT
6339 026556 001403          BEQ    2$                ;NO, CHECK FOR OTHER ERRORS
6340 026560 104401 061400  TYPE   ,ERR006          ;TYPE 'MULTIPLE DRIVE SELECT'
6341 026564 000501          BR     10$              ;DISPLAY REGISTERS
6342
6343 026566 032704 000001  2$:   BIT    #E.CCLR,R4    ;CHECK IF "CLEAR CONTROLLER DID NOT
6344                                ; CLEAR ERROR"
6345 026572 001403          BEQ    3$                ;NO, CHECK OTHER CONTROLLER ERRORS
6346 026574 104401 061156  TYPE   ,ERR001          ;
6347 026600 000473          BR     10$              ;DISPLAY REGISTERS
6348
6349 026602 032704 000002  3$:   BIT    #E.NOAT,R4     ;CHECK IF 'NO ATTENTION IN RKASOF'
6350 026606 001403          BEQ    4$                ;NO, CHECK THE OTHER CONTROLLER ERRORS
6351 026610 104401 061223  TYPE   ,ERR002          ;
6352 026614 000465          BR     10$              ;DISPLAY REGISTERS
6353
6354 026616 032704 000010  4$:   BIT    #E.UDAT,R4     ;CHECK IF 'UNEXPECTED DATA TYPE ERROR'
6355 026622 001403          BEQ    5$                ;NO, CHECK OTHER CONTROLLER ERRORS
6356 026624 104401 061252  TYPE   ,ERR003          ;
6357 026630 000457          BR     10$              ;DISPLAY REGISTERS
6358
6359 026632 032704 000020  5$:   BIT    #E.CLAT,R4     ;CHECK IF "ATTENTION DID NOT RESET
6360                                ; WITH DRIVE CLEAR"
6361 026636 001403          BEQ    6$                ;NO, CHECK OTHER CONTROLLER ERRORS
6362 026640 104401 061305  TYPE   ,ERR004          ;
6363 026644 000451          BR     10$              ;DISPLAY REGISTERS
6364
6365 026646 032704 000100  6$:   BIT    #E.ILLD,R4     ;CHECK IF "ILLEGAL DRIVER COMMAND"
6366 026652 001403          BEQ    7$                ;NO, CHECK IF DATA LATE WHEN UNLOADING HEADER
6367 026654 104401 061351  TYPE   ,ERR005          ;
6368 026660 000443          BR     10$              ;DISPLAY REGISTERS
6369
6370 026662 032704 000400  7$:   BIT    #E.DLT,R4      ;CHECK IF "DATA LATE WHEN UNLOADING HEADER"
6371 026666 001403          BEQ    8$                ;NO, CHECK IF "CONTROLLER ERROR WHILE GATERING
6372                                ; STATUS"
6373 026670 104401 064207  TYPE   ,ERR200          ;
6374 026674 000435          BR     10$              ;DISPLAY REGISTERS
6375
6376 026676 032704 001000  8$:   BIT    #E.CERR,R4     ;CHECK IF "CONTROLLER ERROR DURING
6377                                ; DRIVE SERVICING"
6378 026702 001425          BEQ    9$                ;NO, CHECK IF CONTROLLER TIME OUT
6379 026704 032737 000040 001426  BIT    #DTYE,T.ER        ;SEE IF DRIVE TYPE ERROR
6380 026712 001003          BNE   13$              ;BR IF YES
6381 026714 104401 064247  TYPE   ,ERR201          ;
6382 026720 000423          BR     10$              ;DISPLAY REGISTERS
6383 026722 132765 000001 000210 13$:  BITB   #BIT0,P.ASSN(R5) ;SEE IF IN PROCESS OF ASSIGNING
6384 026730 001003          BNE   14$              ;BR IF YES
6385 026732 104401 064247  TYPE   ,ERR201          ;ELSE VALID ERROR

```

```

6386 026736 000414 BR 10$ :DISPLAY REGS
6387
6388 026740 004737 027206 14$: JSR PC,FLPARM :FILL PARAM BLOCK WITH CORRECT RK06-07 DATA
6389 026744 012737 000001 027334 MOV #1,NEWFLG :SET FLAG TO SHOW DTYE ERROR ON ASSIGNING
6390 026752 012604 MOV (SP)+,R4 :RESTORE R4
6391 026754 000427 BR 12$
6392
6393 026756 032704 040000 9$: BIT #E.CMTO,R4 :CHECK IF CONTROLLER TIME OUT
6394 026762 001414 BEQ 20$ :NO, CHECK IF DRIVE DETECTED DRIVE BUS PARITY ERROR
6395 026764 104401 064534 TYPE ,ERR207
6396
6397 026770 104401 001165 10$: TYPE ,SCRLF :TYPE <CR><LF>
6398 026774 004737 044046 JSR PC,PRITIM :PRINT TIME
6399 027000 004737 026030 JSR PC,PRIREG :PRINT CONTROLLER REGISTERS
6400 027004 104401 061131 TYPE ,ERR000 :TYPE FATAL ERROR
6401 027010 000000 HALT :HANG ON FATAL ERROR
6402 027012 000776 BR .-2
6403
6404 027014 032704 002000 20$: BIT #E.DPAR,R4 :CHECK IF 'DRIVE DETECTED DRIVE BUS PARITY ERROR'
6405 027020 001001 BNE 30$ :YES, SET DRPAR IN ERROR REGISTER
6406 027022 000000 HALT
6407
6408 027024 012604 000010 000034 30$: MOV (SP)+,R4 :RESTORE R4
6409 027026 052765 000010 000034 BIS #DRPAR,P.ER(R5) :SET DRIVE DETECTED DRIVE BUSSOTT> PARITY
6410 027034 010346 12$: MOV R3,-(SP) :STORE R3 ON THE STACK
6411 027036 005037 001464 CLR O.WAIT :CLEAR WAITING FOR COMMAND COMPLETION
6412 027042 116503 000000 MOVB P.DRVN(R5),R3 :GET DRIVE NUMBER
6413 027046 005062 000026 CLR RKMR1(R2) :CLEAR MAINTENANCE REGISTER 1
6414 027052 010362 000010 MOV R3,RKCS2(R2) :LOAD DRIVE NUMBER
6415 027056 052762 000040 000010 BIS #SCLR,RKCS2(R2) :DO SUBSYS CLR
6416 027064 105762 000000 35$: TSTB RKCS1(R2) :WAIT FOR READY
6417 027070 001775 BEQ 35$
6418 027072 004737 047414 JSR PC,I.STOR :STORE REGISTERS
6419 027076 032737 100000 001410 BIT #CERR,T.CS1 :CHECK IF CONTROLLER ERROR
6420 027104 001413 BEQ 37$ :NO, CHECK IF ATTENTION CLEARED
6421
6422 : CHECK FOR CONTROLLER TYPE ERROR
6423 027106 032737 137400 001412 BIT #UFE!MDS!PGE!NEM!NED!UPE!DLT,T.CS2
6424 027114 001004 BNE 36$
6425 027116 032737 130761 001426 BIT #ILC!DTYE!FMTE!ECH!BSE!HVRC!DTE!OPI!DCK,T.ER
6426 027124 001412 BEQ 40$
6427
6428 027126 104401 064247 36$: TYPE ,ERR201 :TYPE CONTROLLER ERROR DURING DRIVE SERVICING
6429 027132 000716 BR 10$ :GO REPORT ERROR
6430
6431 027134 136337 001514 001425 37$: BITB INTMSK(R3),T.ASOF+1 :CHECK IF ATTENTION CLEARED
6432 027142 001403 BEQ 40$ :YES, REPORT ERROR
6433 027144 104401 061305 TYPE ,ERR004 :TYPE 'DRIVE CLEAR DID NOT RESET ATTENTION'
6434 027150 000707 BR 10$ :GO REPORT ERROR
6435
6436 027152 146337 001514 001512 40$: BICB INTMSK(R3),W.TIME :CLEAR TIMING CURRENTLY ON THIS DRIVE
6437 027160 006303 ASL R3 :MULTIPLY BY 2
6438 027162 005063 001544 CLR W.DRV(R3) :CLEAR WATCH DOG TIME
6439 027166 012603 MOV (SP)+,R3 :RESTORE R3
6440 027170 005737 027334 TST NEWFLG :SEE IF DTYE ERROR ON ASSIGNING
6441 027174 001402 BEQ 41$ :BR IF NO

```

```
6442 027176 000137 024400          JMP      NORMAL      ;ELSE GO NORMAL RET
6443 027202 000137 025470      41$:    JMP      ABNORM      ;GO TO ABNORMAL RETURN
6444
6445 027206 132765 000004 000007  FLPARM: BITB      #B.CDT,P.CS1H(R5)      ;SEE IF RK06 PARMETER USED LAST
6446 027214 001407          BEQ      1$              ;BR IF YES
6447 027216 142765 000004 000007  BICB      #B.CDT,P.CS1H(R5)      ;ELSE TRY RK06 PARAM BECAUSE OF DTYE
6448 027224 012765 000632 000070  MOV      #632,P.MXCL(R5)
6449 027232 000207          RTS      PC
6450
6451 027234 152765 000004 000007  1$:     BISB      #B.CDT,P.CS1H(R5)      ;TRY RK07 PARAM BECAUSE OF DTYE
6452 027242 012765 001456 000070  MOV      #1456,P.MXCL(R5)
6453 027250 000207          RTS      PC
6454
6455 027252 132765 000004 000007  GETNUM: BITB      #B.CDT,P.CS1H(R5)      ;SEE IF RK06
6456 027260 001412          BEQ      1$              ;BR IF YES
6457
6458 027262 012737 001456 027332  MOV      #814.,LSTCYL      ;ELSE LOAD RK07 VALUES
6459 027270 012737 151010 027336  MOV      #<814.*22.*3.>+<22.*2.>,HOLD
6460 027276 012737 151011 027340  MOV      #<814.*22.*3.>+<22.*2.>+1,HOLD1
6461 027304 000207          RTS      PC
6462
6463 027306 012737 000632 027332  1$:     MOV      #410.,LSTCYL      ;LOAD RK06 VALUES
6464 027314 012737 064740 027336  MOV      #<410.*22.*3.>+<22.*2.>,HOLD
6465 027322 012737 064741 027340  MOV      #<410.*22.*3.>+<22.*2.>+1,HOLD1
6466 027330 000207          RTS      PC
6467
6468 027332 000000          LSTCYL: 0
6469 027334 000000          NEWFLG: 0
6470 027336 000000          HOLD: 0
6471 027340 000000          HOLD1: 0
6472
```



```

6473      .SBTTL  CONTROLLER ERROR DUE TO REIGSTER INCONSISTANCIES
6474
6475 027342 004737 052004      CNTRL1: JSR      PC,Q.RMOV      ;REMOVE PARAMETER BLOCK FROM THE
6476                                     ; COMMAND INITIATION QUEUE
6477 027346 032777 002000 151564      BIT      #SW10,@SWR      ;CHECK IF BELL ON ERROR
6478 027354 001402                                     BEQ      1$              ;NO, CONTINUE
6479 027356 104401 001160                                     TYPE     ,SBELL         ;RING BELL
6480 027362 032777 020000 151550 1$:  BIT      #SW13,@SWR      ;CHECK IF INHIBIT PRINT OUT
6481 027370 001404                                     BEQ      2$              ;NO PRINT ERROR MESSAGE
6482 027372 032777 040000 151540      BIT      #SW14,@SWR      ;CHECK IF CYCLE ON OPERATION
6483 027400 001050                                     BNE     10$             ;YES, CHECK IF LOOP ON OPERATION
6484 027402 032737 000001 001564 2$:  BIT      #BIT0,ERCONT    ;CHECK IF "CONTROLLER ERROR NOT FLAGGED"
6485 027410 001402                                     BEQ      3$
6486 027412 104401 061707                                     TYPE     ,ERR013
6487 027416 032737 000002 001564 3$:  BIT      #BIT1,ERCONT    ;CHECK IF "WORD COUNT NOT 0"
6488 027424 001402                                     BEQ      4$
6489 027426 104401 061746                                     TYPE     ,ERR014
6490 027432 032737 000004 001564 4$:  BIT      #BIT2,ERCONT    ;CHECK IF "BUS ADDRESS INCORRECT"
6491 027440 001402                                     BEQ      5$
6492 027442 104401 061777                                     TYPE     ,ERR015
6493 027446 032737 000010 001564 5$:  BIT      #BIT3,ERCONT    ;CHECK IF "CYLINDER,TRACK,SECTOR
6494                                     ; INCORRECT"
6495 027454 001402                                     BEQ      7$
6496 027456 104401 062023                                     TYPE     ,ERR016
6497 027462 032777 020000 151450 7$:  BIT      #SW13,@SWR      ;CHECK IF INHIBIT PRINT OUT
6498 027470 001014                                     BNE     10$             ;YES, CHECK IF HALT ON ERROR
6499 027472 004737 044046      JSR      PC,PRITIM      ;PRINT TIME
6500 027476 004737 026252      JSR      PC,PRISTT      ;PRINT RK06 REGISTERS
6501 027502 004737 030422      JSR      PC,PRI002      ;PRINT CURRENT SUPPLIED COMMAND
6502 027506 032737 000001 001564      BIT      #BIT0,ERCONT    ;CHECK IF CONTROLLER ERROR
6503                                     ; NOT FLAGGED
6504 027514 001402                                     BEQ      10$            ;NO, DO NOT PRINT PREVIOUS COMMAND
6505 027516 004737 030244      JSR      PC,PRI001      ;PRINT PREVIOUS COMMAND
6506 027522 032777 100000 151410 10$: BIT      #SW15,@SWR      ;CHECK IF HALT ON ERROR
6507 027530 001010                                     BNE     11$             ;YES, HALT
6508 027532 032777 040000 151400      BIT      #SW14,@SWR      ;CHECK IF LOOP OPERATION
6509 027540 001005                                     BNE     15$             ;YES, CLEAR ERROR
6510 027542 104401 061131      TYPE     ,ERR000        ;PRINT "FATAL ERROR"
6511 027546 104401 061426      TYPE     ,ERR007        ;PRINT "SET SWITCH 14 TO CYCLE ON ERROR"
6512 027552 000000                                     HALT
6513 027554 012737 100000 001564 11$: MOV      #BIT15,ERCONT    ;SET CONTROLLER ERROR PRESENT
6514 027562 012762 100000 000000 15$: MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
6515 027570 005062 000026      CLR      RKMR1(R2)      ;CLEAR MAINTANENCE REGISTER 1
6516 027574 112737 000005 001410      MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND INTO TEMPORARY CS1
6517 027602 004037 047152      JSR      RO,I.ISSU      ;GO ISSUE DRIVE CLEAR
6518 027606 027616      20$
6519 027610 004037 051564      JSR      RO,Q.PUSH      ;ERROR RETURN
6520 027614 001324      JSR      CINITQ        ;PUT PARAMETER BLOCK ON COMMAND
6521 027616 000207      RTS      PC             ; INITIATION QUEUE
                                     ;RETURN
    
```

```
6522 .SBTTL PRINT WHOLE DATA BUFFER
6523
6524 027620 122765 000121 000123 PRIBUF: CMPB #RDATA,P.RCMD(R5) ;CHECK IF READ DATA
6525 027626 001017 BNE 5$ ;NO, RETURN
6526 027630 032765 000010 000150 BIT #BIT3,P.DSTT(R5) ;CHECK IF TIME TO PRINT BUFFER
```

6527	027636	001013			BNE	5\$:NO, RETURN
6528	027640	052765	000010	000150	BIS	#BIT3,P.DSTT(R5)	:SET BUFFER ATTEMPTED TO BE PRINTED
6529	027646	032777	020000	151264	BIT	#SW13,@SWR	:CHECK IF INHIBIT PRINT OUT
6530	027654	001004			BNE	5\$:YES, RETURN
6531	027656	032777	000010	151254	BIT	#SW3,@SWR	:CHECK IF PRINT SECTOR BEFORE BEGINNING
6532							: ERROR RECOVERY
6533	027664	001001			BNE	10\$:YES, PRINT SECTOR IN ERROR
6534	027666	000207			RTS	PC	:RETURN
6535							
6536	027670	010146			MOV	R1,-(SP)	:STORE R1 ON THE STACK
6537	027672	010346			MOV	R3,-(SP)	:STORE R3 ON THE STACK
6538	027674	010446			MOV	R4,-(SP)	:STORE R4 ON THE STACK
6539	027676	104401	064717		TYPE	,ERR213	:TYPE ADDRESS OF BEGINNING OF
6540							: SECTOR IN ERROR
6541	027702	013703	001634		MOV	RTYBA,R3	:GET ADDRESS OF START OF SCTOR
6542	027706	013704	001636		MOV	RTYWC,R4	:GET WORD COUNT
6543	027712	010346			MOV	R3,-(SP)	:STORE STARTING ADDRESS FOR PRINT OUT
6544	027714	004737	053224		JSR	PC,BINOCT	:CONVERT OF OCTAL
6545	027720	104401	064770		TYPE	,ERR214	:TYPE DATA READ
6546	027724	012701	000010		MOV	#8,R1	:SET 8 COLUMNS ON PAGE
6547	027730	104401	001165		TYPE	,\$CRLF	:TYPE <CR><LF>
6548	027734	012346			MOV	(R3)+,-(SP)	:GET NEXT WORD OF SECTOR
6549	027736	004737	053224		JSR	PC,BINOCT	:CONVERT TO OCTAL
6550	027742	005304			DEC	R4	:CHECK IF FINISHED
6551	027744	001405			BEQ	30\$:YES, RESTORE REGISTERS
6552	027746	005301			DEC	R1	:CHECK IF END OF LINE
6553	027750	001765			BEQ	15\$:YES, INITIALIZE COLUMN COUNT
6554	027752	104401	061126		TYPE	,BLANKS	:TYPE 2 BLANKS
6555	027756	000766			BR	20\$:GET NEXT WORD
6556							
6557	027760	104401	001165		TYPE	,\$CRLF	:TYPE <CR><LF>
6558	027764	012604			MOV	(SP)+,R4	:RESTORE R4
6559	027766	012603			MOV	(SP)+,R3	:RESTORE R3
6560	027770	012601			MOV	(SP)+,R1	:RESTORE R1
6561	027772	000207			RTS	PC	:RETURN

```

6562
6563
6564 027774 010146
6565 027776 010346
6566 030000 010446
6567 030002 012701 057176
6568 030006 016503 000224
6569 030012 006103
6570 030014 006103
6571 030016 006103
6572 030020 006103
6573 030022 012704 000003
6574 030026 105011
6575 030030 006103
6576 030032 106111
6577 030034 006103
6578 030036 106111
6579 030040 006103
6580 030042 106111
6581 030044 006103
6582 030046 106111
6583 030050 152721 000060
6584 030054 005304
6585 030056 001363
6586 030060 104401 057153
6587 030064 016503 000226
6588
6589 030070 016504 000230
6590
6591 030074 012701 057541
6592 030100 105011
6593 030102 000404
6594
6595 030104 105011
6596 030106 006103
6597 030110 006104
6598 030112 106111
6599 030114 006103
6600 030116 006104
6601 030120 106111
6602 030122 006103
6603 030124 006104
6604 030126 106111
6605 030130 152721 000060
6606 030134 022701 057554
6607 030140 101361
6608 030142 104401 057514
6609 030146 012604
6610 030150 012603
6611 030152 012601
6612 030154 000207

.SBTTL PRINT DRIVE AND PACK SERIAL NUMBERS
PRISER: MOV R1,-(SP) ;STORE R1 ON STACK
MOV R3,-(SP) ;STORE R3 ON STACK
MOV R4,-(SP) ;STORE R4 ON STACK
MOV #STT100,R1 ;GET ADDRESS OF ASCII STRING
MOV P.SERL(R5),R3 ;GET DRIVE SERIAL NUMBER
ROL R3 ;SHIFT DRIVE SERIAL NUMBER 4 BITS LEFT
ROL R3
ROL R3
ROL R3
2$: MOV #3,R4 ;LOAD NUMBER OF DIGITS OF SERIAL NUMBER
CLRB (R1) ;CLEAR NEXT BYTE FOR PRINT OUT
ROL R3 ;SHIFT IN NEXT 4 BITS
ROLB (R1)
ROL R3
ROLB (R1)
ROL R3
ROLB (R1)
BISB #60,(R1)+ ;CHANGE BCD TO DECIMAL
DEC R4 ;CHECK IF FINISHED
BNE 2$ ;CONVERT NEXT DIGIT
TYPE ,STT000 ;TYPE 'DRIVE SERIAL NUMBER'
MOV P.PKSR(R5),R3 ;GET LEAST SIGNIFICANT BITS OF
; PACK SERIAL NUMBER
MOV P.PKSR+2(R5),R4 ;GET MOST SIGNIFICANT BITS OF
; PACK SERIAL NUMBER
MOV #STT012,R1 ;GET ADDRESS OF ASCII STRING
CLRB (R1) ;CLEAR FIRST BYTE OF PRINT OUT
BR 7$ ;START CONVERSION TO OCTAL
5$: CLRB (R1) ;CLEAR NEXT BYTE OF PRINT OUT
ROL R3 ;SHIFT IN NEXT THREE BITS
ROL R4
ROLB (R1)
7$: ROL R3
ROL R4
ROLB (R1)
BISB #60,(R1)+ ;MAKE IT ASCII
CMP #STT013,R1 ;CHECK IF FINISHED
BHI 5$ ;NO, CONTINUE CONVERSION
TYPE ,STT011 ;TYPE PACK SERIAL NUMBER
MOV (SP)+,R4 ;RESTORE R4
MOV (SP)+,R3 ;RESTORE R3
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

```

```
6613 .SBTTL PRINT BEGINNING OF ERROR HEADER
6614
6615 030156 032777 002000 150754 PRI000: BIT #SW10,@SWR ;CHECK IF BELL ON ERROR
6616 030164 001402 BEQ 1$ ;NO, CHECK IF INHIBIT PRINT OUT
6617 030166 104401 001160 TYPE ,SBELL ;RING BELL
6618 030172 032777 020000 150740 1$: BIT #SW13,@SWR ;CHECK IF INHIBIT PRINT OUT
6619 030200 001020 BNE 20$ ;YES, INCREMENT CONTROLLER ERROR
6620 030202 116537 000000 056755 MOVB P.DRVN(R5),OPR011 ;STORE DRIVE NUMBER
6621 030210 152737 000060 056755 BISB #60,OPR011 ;MAKE IT ASCII
6622 030216 104401 056736 TYPE ,OPR010 ;TYPE 'DRIVE NUMBER N'
6623 030222 132765 000001 000210 BITB #BIT0,P.ASSN(R5) ;SEE IF SIZING FOR DRIVE
6624 030230 001004 BNE 20$ ;BR IF YES, TO RETURN
6625 030232 004737 027774 JSR PC,PRISER ;PRINT DRIVE AND PACK SERIAL NUMBERS
6626 030236 004737 044046 JSR PC,PRITIM ;PRINT TIME
6627 030242 000207 20$: RTS PC ;RETURN
```

```
6628 .SBTTL PRINT PREVIOUS COMMAND
6629
6630 030244 104401 064060 000135 PRI001: TYPE ,ERR101 :TYPE 'PREVIOUS CMND CYL TRK SEC WRD CT'
6631 030250 122765 000121 000135 CMPB #RDATA,P.LCMD(R5) :CHECK IF READ DATA
6632 030256 001402 BEQ 5$ :YES, DO NOT PRINT PATTERN NUM
6633 030260 104401 064142 TYPE ,ERR102 :PRINT 'PAT'
6634 030264 104401 001165 5$: TYPE ,$CRLF :PRINT <CR><LF>
6635 030270 005046 CLR -(SP) :MAKE ROOM ON STACK
6636 030272 116516 000135 MOVB P.LCMD(R5),(SP) :GET LAST COMMAND
6637 030276 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6638 030302 104401 061126 TYPE ,BLANKS :TYPE 2 BLANKS
6639 030306 016546 000136 MOV P.LCYL(R5),-(SP) :GET LAST CYLINDER USED
6640 030312 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6641 030316 104401 061126 TYPE ,BLANKS :TYPE 2 BLANKS
6642 030322 005046 CLR -(SP) :MAKE ROOM ON STACK
6643 030324 116516 000141 MOVB P.LTRK(R5),(SP) :GET LAST TRACK USED
6644 030330 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6645 030334 104401 061126 TYPE ,BLANKS :TYPE 2 BLANKS
6646 030340 005046 CLR -(SP) :MAKE ROOM ON STACK
6647 030342 116516 000140 MOVB P.LSEC(R5),(SP) :GET LAST SECTOR USED
6648 030346 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6649 030352 104401 061126 TYPE ,BLANKS :TYPE 2 BLANKS
6650 030356 016546 000142 MOV P.LWC(R5),-(SP) :STORE LAST WORD COUNT
6651 030362 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6652 030366 122765 000121 000135 CMPB #RDATA,P.LCMD(R5) :CHECK IF PAT # VALID
6653 030374 001407 BEQ 10$ :NO, DO NOT TYPE
6654 030376 104401 061126 TYPE ,BLANKS :TYPE 2 BLANKS
6655 030402 005046 CLR -(SP) :MAKE ROOM ON STACK
6656 030404 116516 000144 MOVB P.LDPT(R5),(SP) :GET LAST DATA PATTERN
6657 030410 004737 053224 JSR PC,BINOCT :CONVERT TO OCTAL
6658 030414 104401 001165 10$: TYPE ,$CRLF :TYPE <CR><LF>
6659 030420 000207 RTS PC :RETURN
```

```
.SBTTL PRINT CURRENT GENERATED COMMAND
6660
6661
6662 030422 104401 063747 PRI002: TYPE ,ERR100 ;PRINT 'CURRENT SUPPLIED COMMAND'
6663 ; CYL TRK SEC OFFSET
6664 ; BUS AD WRD CT''
6665 030426 122765 000121 000123 CMPB #RDDATA,P.RCMD(R5) ;CHECK IF PAT # IS TO BE PRINTED
6666 030434 001402 BEQ 5$ ;NO, DO NOT PRINT HEADER
6667 030436 104401 064142 TYPE ,ERR102 ;PRINT 'PAT'
6668 030442 104401 001165 5$: TYPE ,$CRLF ;PRINT <CR><LF>
6669 030446 005046 CLR -(SP) ;MAKE ROOM ON STACK
6670 030450 116516 000001 MOV P.CMND(R5),(SP) ;STORE COMMAND FOR PRINT OUT
6671 030454 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6672 030460 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6673 030464 016546 000002 MOV P.CYLN(R5),-(SP) ;STORE CYLINDER
6674 030470 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6675 030474 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6676 030500 005046 CLR -(SP) ;MAKE ROOM ON STACK
6677 030502 116516 000005 MOV P.TRCK(R5),(SP) ;STORE TRACK
6678 030506 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6679 030512 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6680 030516 005046 CLR -(SP) ;MAKE ROOM ON STACK
6681 030520 116516 000004 MOV P.SECT(R5),(SP) ;STORE SECTOR
6682 030524 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6683 030530 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6684 030534 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
6685 030536 122765 000021 000147 CMPB #21,P.RERD(R5) ;CHECK IF OFFSET
6686 030544 101002 BHI 10$ ;NO, INDICATE ZERO OFFSET
6687 030546 116516 000006 MOV P.OFST(R5),(SP) ;STORE OFFSET FOR PRINT OUT
6688 030552 004737 053224 10$: JSR PC,BINOCT ;CONVERT TO OCTAL
6689 030556 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6690 030562 016546 000010 MOV P.BALO(R5),-(SP) ;STORE BUS ADDRESS
6691 030566 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6692 030572 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6693 030576 016546 000012 MOV P.WC(R5),-(SP) ;STORE WORD COUNT
6694 030602 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6695 030606 122765 000121 000123 CMPB #RDDATA,P.RCMD(R5) ;CHECK IF PAT # IS TO BE PRINTED
6696 030614 001407 BEQ 15$ ;NO, RETURN
6697 030616 104401 061126 TYPE ,BLANKS ;TYPE 2 BLANKS
6698 030622 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
6699 030624 116516 000121 MOV P.DPAT(R5),(SP) ;STORE DATA PATTERN FOR PRINT OUT
6700 030630 004737 053224 JSR PC,BINOCT ;CONVERT TO OCTAL
6701 030634 104401 001165 15$: TYPE ,$CRLF ;TYPE <CR><LF>
6702 030640 000207 RTS PC ;RETURN
```

6703
6704
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717 030642 010446
6718 030644 012004
6719 030646 005204
6720 030650 010437 030662
6721 030654 004737 030156
6722 030660 104401
6723 030662 000000
6724 030664 132765 000001 000210
6725 030672 001045
6726 030674 104401 001165
6727 030700 032777 020000 150232
6728 030706 001033
6729 030710 105765 000150
6730 030714 100402
6731 030716 104401 064317
6732 030722 004737 026252
6733 030726 132744 000002
6734 030732 001407
6735 030734 132714 000040
6736 030740 001402
6737 030742 104401 064500
6738 030746 004737 026076
6739 030752 132714 000100
6740 030756 001402
6741 030760 004737 030422
6742 030764 132714 000004
6743 030770 001402
6744 030772 004737 030244
6745 030776 004737 031426
6746 031002 005265 000206
6747 031006 004737 023020
6748 031012 004737 013162
6749 031016 004737 022520
6750 031022 005037 001566
6751 031026 012604
6752 031030 000200

.SBTTL PRINT ERROR MESSAGE AND DROP DRIVE

```
*****  
: *  
: * EXPECTED PRINT CODES  
: * -----  
: *  
: * BIT 1 PRINT DRIVE STATUS  
: * BIT 2 PRINT PREVIOUS POSITION  
: * BIT 5 QUESTIONABLE DRIVE STATUS  
: * BIT 6 CURRENT GENERATED COMMAND  
: *  
: *****
```

```
PRI100: MOV R4, -(SP) :STORE R4 ON STACK  
MOV (R0)+, R4 :STORE ADDRESS OF PRINT CODE  
INC R4 :CALCULATE START OF MESSAGE  
MOV R4, 2$ :STORE ERROR MESSAGE ADDRESS  
JSR PC, PRI000 :PRINT HEADER  
TYPE :TYPE ERROR MESSAGE  
2$: .WORD 0  
BITB #BIT0, P.ASSN(R5) :SEE IF SIZING FOR DRIVE  
BNE 12$ :BR IF YES  
TYPE , SCRLF :TYPE <CR><LF>  
BIT #SW13, @SWR :CHECK IF INHIBIT PRINTOUT  
BNE 10$ :YES, RETURN  
TSTB P.DSTT(R5) :CHECK IF FIRST ERROR  
BMI 3$ :YES, GO PRINT CONTROLLER REGISTERS  
TYPE , ERR202 :TYPE ERROR DURING ERROR RECOVERY  
JSR PC, PRISTT :PRINT CONTROLLER REGISTERS  
BITB #BIT1, -(R4) :CHECK IF PRINT DRIVE STATUS  
BEQ 5$ :NO, CONTINUE  
BITB #BIT5, (R4) :CHECK IF QUESTIONABLE STATUS  
BEQ 4$ :NO, DO NOT PRINT MESSAGE  
TYPE , ERR206 :PRINT "QUESTIONABLE DRIVE STATUS"  
4$: JSR PC, PRDSTT :PRINT DRIVE STATUS  
5$: BITB #BIT6, (R4) :CHECK IF PRINT CURRENT GENERATED COMMAND  
BEQ 8$ :NO, DO NOT PRINT CURRENT COMMAND  
JSR PC, PRI002 :PRINT CURRENT GENERATED COMMAND  
8$: BITB #BIT2, (R4) :CHECK IF PRINT PREVIOUS COMMAND  
BEQ 10$ :NO, CHECK IF HALT ON ERROR  
JSR PC, PRI001 :PRINT PREVIOUS COMMAND  
10$: JSR PC, HLT000 :TEST IF HALT ON ERROR  
INC P.NER(R5) :INCREMENT OTHER ERROR COUNT  
12$: JSR PC, BUFREL :RELEASE BUFFER  
JSR PC, DROP :DROP DRIVE FROM TEST SEQUENCE  
JSR PC, GETBUF :GET NEW DRIVE, COMMAND, AND BUFFER  
CLR ERRPRO :CLEAR ERROR CURRENTLY BEING PROCESSED  
MOV (SP)+, R4 :RESTORE R4  
RTS R0 :RETURN
```


6753
6754
6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774
6775
6776
6777
6778
6779
6780
6781
6782
6783
6784
6785
6786
6787
6788
6789
6790
6791
6792
6793
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808

.SBTTL PRINT ERROR MESSAGE

```

*****
*
* EXPECTED PRINT CODES
* -----
*
* BIT 0 DATA TYPE ERROR = 1
*       NON-DATA TYPE ERROR = 0
*
* BIT 1 PRINT DRIVE STATUS
* BIT 2 PRINT PREVIOUS POSITION
* BIT 3 PRINT SECTOR IN ERROR
* BIT 4 PRINT HEADER
* BIT 5 QUESTIONABLE DRIVE STATUS
* BIT 6 CURRENT GENERATED COMMAND
*
*****

```

```

031032 010446 PRI200: MOV R4, -(SP) ;STORE R4 ON STACK
031034 012004 MOV (R0)+, R4 ;GET PRINT CODE ERROR FLAGS
031036 132724 000001 BITB #BIT0, (R4)+ ;CHECK IF DATA TYPE ERROR
031042 001416 BEQ 5$ ;NO, NON-DATA TYPE ERROR
031044 032765 000002 000150 BIT #BIT1, P.DSTT(R5) ;CHECK IF DATA TYPE ERROR
; BEING PROCESSED
BEQ 1$ ;NO, PRINT MESSAGE
031052 001402 MOV (SP)+, R4 ;RESTORE R4
031054 012604 RTS R0 ;RETURN
031056 000200

031060 052765 000002 000150 1$: BIS #BIT1, P.DSTT(R5) ;SET DATA TRANSFER ERROR OCCURRED
031066 100015 BPL 10$ ;CHECK IF START OF ERROR RECOVERY
031070 052765 000001 000150 BIS #BIT0, P.DSTT(R5) ;YES, SET SERVICING DATA ERROR
031076 000411 BR 10$ ;GO REPORT ERROR

031100 032765 000004 000150 5$: BIT #BIT2, P.DSTT(R5) ;CHECK IF NON-DATA TYPE ERROR OCCURRED
031106 001402 BEQ 6$ ;NO, REPORT ERROR
031110 012604 MOV (SP)+, R4 ;RESTORE R4
031112 000200 RTS R0 ;RETURN

031114 052765 000004 000150 6$: BIS #BIT2, P.DSTT(R5) ;SET NON-DATA ERROR OCCURRED
031122 004737 030156 10$: JSR PC, PRI000 ;TYPE ERROR HEADER
031126 032777 020000 150004 BIT #SW13, aSWR ;CHECK IF INHIBIT PRINT OUT
031134 001115 BNE 30$ ;YES, INHIBIT PRINT OUT
031136 010437 031156 MOV R4, 12$ ;STORE ERROR MESSAGE
031142 105765 000150 TSTB P.DSTT(R5) ;CHECK IF ERROR DURING RECOVERY
031146 100402 BMI 11$ ;NO, PRINT ERROR
031150 104401 064317 TYPE .ERR202 ;PRINT "ERROR DURING ERROR RECOVERY"
031154 104401 11$: TYPE ;PRINT ERROR MESSAGE
031156 000000 12$: .WORD 0
031160 104401 001165 TYPE .$CRLF ;TYPE <CR><LF>
031164 004737 026252 JSR PC, PR1TT ;PRINT CONTROLLER REGISTERS
031170 132744 000002 BITB #BIT1, -(R4) ;CHECK IF PRINT DRIVE STATUS
031174 001407 BEQ 22$ ;NO, CONTINUE
031176 132714 000040 BITB #BIT5, (R4) ;CHECK IF QUESTIONABLE DRIVE STATUS
031202 001402 BEQ 20$ ;NO, DO NOT PRINT MESSAGE
031204 104401 064500 TYPE .ERR206 ;TYPE "QUESTIONABLE DRIVE STATUS"
031210 004737 026076 20$: JSR PC, PRDSTT ;PRINT DRIVE STATUS

```

6809	031214	132714	000100		22\$:	BITB	#BIT6,(R4)	:CHECK IF PRINT CURRENT COMMAND
6810	031220	001402				BEQ	23\$:NO, CONTINUE
6811	031222	004737	030422			JSR	PC,PRIO02	:PRINT CURRENTLY ISSUED COMMAND
6812	031226	132714	000004		23\$:	BITB	#BIT2,(R4)	:CHECK IF PRINT PREVIOUS COMMAND
6813	031232	001402				BEQ	25\$:NO, CHECK IF PRINT HEADER
6814	031234	004737	030244			JSR	PC,PRIO01	:PRINT PREVIOUS COMMAND
6815	031240	132714	000001		25\$:	BITB	#BIT0,(R4)	:CHECK IF DATA TYPE ERROR
6816	031244	001422				BEQ	27\$:NO, DO NOT PRINT ECC
6817	031246	032765	100000	000034		BIT	#DCK,P.ER(R5)	:CHECK IF DATA CHECK
6818	031254	001416				BEQ	27\$:NO, DO NOT PRINT ECC
6819	031256	104401	063546			TYPE	,ERR067	:TYPE ECC PAT, ECC POS
6820	031262	016546	000062			MOV	P.EPAT(R5),-(SP)	:GET PATTERN
6821	031266	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
6822	031272	104401	061126			TYPE	,BLANKS	
6823	031276	016546	000060			MOV	P.EPOS(R5),-(SP)	:GET POSITION
6824	031302	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
6825	031306	104401	001165			TYPE	,\$CRLF	
6826	031312	132724	000020		27\$:	BITB	#BIT4,(R4)+	:CHECK IF PRINT HEADER
6827	031316	001424				BEQ	30\$:NO, CHECK IF HALT ON ERROR
6828	031320	104401	064161			TYPE	,ERR111	:TYPE HEADER DESCRIPTOR
6829	031324	013746	001646			MOV	HEAD1,-(SP)	:STORE FIRST WORD OF THE HEADER
6830	031330	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
6831	031334	104401	061126			TYPE	,BLANKS	:TYPE 2 BLANKS
6832	031340	013746	001650			MOV	HEAD2,-(SP)	:STORE SECOND WORD OF THE HEADER
6833	031344	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
6834	031350	104401	061126			TYPE	,BLANKS	:TYPE 2 BLANKS
6835	031354	013746	001652			MOV	HEAD3,-(SP)	:STORE THIRD WORD OF THE HEADER
6836	031360	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
6837	031364	104401	001165			TYPE	,\$CRLF	:TYPE <CR><LF>
6838	031370	132744	000010		30\$:	BITB	#BIT3,-(R4)	:CHECK IF PRINT WHOLE SECTOR
6839	031374	001402				BEQ	40\$:NO, CHECK IF HALT ON ERROR
6840	031376	004737	027620			JSR	PC,PRIBUF	:GO PRINT BUFFER
6841	031402	132714	000001		40\$:	BITB	#BIT0,(R4)	:CHECK IF DATA TYPE ERROR
6842	031406	001403				BEQ	45\$:NO, CHECK IF HALT ON ERROR
6843	031410	042765	100200	000150		BIC	#BIT15!BIT7,P.DSTT(R5)	:CLEAR FIRST ERROR AND ERROR
6844								: ENQUEUED
6845	031416	004737	031426		45\$:	JSR	PC,HLT000	:CHECK IF HALT ON ERROR
6846	031422	012604				MOV	(SP)+,R4	:RESTORE R4
6847	031424	000200				RTS	RO	:RETURN

```
6848 .SBTTL HALT ON ERROR CHECK
6849
6850 031426 032777 100000 147504 HLT000: BIT #SW15,@SWR ;CHECK IF HALT ON ERROR
6851 031434 001407 BEQ 10$ ;NO, RETURN
6852 031436 005737 001464 TST 0.WAIT ;CHECK IF WAITING FOR COMMAND
6853 ; COMPLETION
6854 031442 001403 BEQ 5$ ;NO, HALT
6855 031444 105762 000000 1$: TSTB RKCS1(R2) ;WAIT FOR READY
6856 031450 001775 BEQ 1$
6857 031452 000000 5$: HALT
6858 031454 000207 10$: RTS PC ;RETURN
```

```
6859 .SBTTL OTHER ERROR STATISTICS GATHERING
6860
6861 031456 105765 000150 NERSTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
6862 031462 100057 BPL STT5$ ;NO, CONTINUE ERROR RECOVERY
6863 031464 042765 100200 000150 BIC #BIT15!BIT7,P.DSTT(R5) ;RESET FIRST ERROR AND ERROR ENQUEUED
6864 031472 005265 000206 INC P.NER(R5) ;INCREMENT OTHER ERRORS COUNT
6865 031476 022765 000020 000206 CMP #T.NER,P.NER(R5) ;CHECK IF ERROR THRESHOLD EXCEEDED
6866 031504 103426 BLO STT2$ ;YES, CHECK IF DROP DRIVE
6867 031506 000445 BR STT5$ ;NO, START ERROR RECOVERY SEQUENCE
6868
6869 .SBTTL SEEK INCOMPLETE STATISTICS GATHERING
6870
6871 031510 105765 000150 SKISTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
6872 031514 100042 BPL STT5$ ;NO, CONTINUE ERROR RECOVERY
6873 031516 005265 000202 INC P.NSKI(R5) ;INCREMENT NUMBER OF SEEK INCOMPLETES
6874 031522 000405 BR STT1$ ;GO CHECK IF ERROR THRESHOLD EXCEEDED
6875
6876 .SBTTL OPERATION INCOMPLETE STATISTICS GATHERING
6877
6878 031524 105765 000150 OPISTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
6879 031530 100034 BPL STT5$ ;NO, CONTINUE ERROR PROCESSING
6880 031532 005265 000204 INC P.NOPI(R5) ;INCREMENT NUMBER OF OPERATION INCOMPLETES
6881 031536 042765 100200 000150 STT1$: BIC #BIT15!BIT7,P.DSTT(R5) ;RESET FIRST ERROR AND ERROR ENQUEUED
6882 031544 016546 000204 MOV P.NOPI(R5),-(SP) ;STORE NUMBER OF OPERATION INCOMPLETES
6883 031550 066516 000202 ADD P.NSKI(R5),(SP) ;ADD SEEK INCOMPLETES
6884 031554 026526 000104 CMP P.SKET(R5),(SP)+ ;CHECK IF ERROR THRESHOLD EXCEEDED
6885 031560 103020 BHIS STT5$ ;NO, START ERROR RECOVERY
6886 031562 032777 040020 147350 STT2$: BIT #SW4!SW14,@SWR ;CHECK IF INHIBIT DRIVE DESELECTION
6887 031570 001014 BNE STT5$ ;YES, START RECOVERY SEQUENCE
6888 031572 104401 064445 TYPE ,ERR205 ;PRINT 'ERROR THRESHOLD EXCEEDED'
6889 031576 004737 023020 JSR PC,BUFREL ;RELEASE BUFFER
6890 031602 004737 013162 JSR PC,DROP ;DROP DRIVE
6891 031606 004737 022520 JSR PC,GETBUF ;GET NEW COMMAND, DRIVE, AND BUFFER
6892 031612 005037 001566 CLR ERRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
6893 031616 011000 MOV (R0),R0 ;INDICATE NO RECOVERY SEQUENCE IN PROGRESS
6894 031620 000200 RTS R0 ;RETURN
6895
6896 031622 105765 000120 STT5$: TSTB P.IERC(R5) ;CHECK IF INHIBIT ERROR RECOVERY
6897 031626 001010 BNE 5$ ;YES, INDICATE UNRECOVERABLE RETURN
6898 031630 105265 000146 INCB P.RECT(R5) ;INCREMENT RETRY COUNT
6899 031634 122765 000010 000146 CMPB #8.,P.RECT(R5) ;CHECK IF RETRY FINISHED
6900 031642 103402 BLO 5$ ;YES, CLEAN UP AFTER UNSUCCESSFUL RETRY
6901 031644 005720 TST (R0)+ ;INDICATE RECOVERY SEQUENCE IN PROGRESS
6902 031646 000200 RTS R0 ;RETURN
6903
6904 031650 004737 031660 5$: JSR PC,UNRECY ;CLEAN UP FOR UNSUCCESSFUL RECOVERY
6905 031654 011000 MOV (R0),R0 ;INDICATE NO RECOVERY SEQUENCE IN PROGRESS
6906 031656 000200 RTS R0 ;RETURN
```

```

        .SBTTL UNSUCCESSFUL RECOVERY CLEANUP
6907
6908
6909 031660 105037 001644 UNRECY: CLRB RTYSCN ;CLEAR RETRY SECTOR COUNT
6910 031664 004737 023020 JSR PC,BUFREL ;RELEASE BUFFER
6911 031670 032765 000001 000150 BIT #BIT0,P.DSTT(R5) ;CHECK IF DATA HARD ERROR
6912 031676 001415 BEQ 1$ ;NO, CHECK IF DRIVE IS BEING ASSIGNED
6913 031700 005265 000200 INC P.HARD(R5) ;INCREMENT HARD ERROR COUNT
6914 031704 026565 000102 000200 CMP P.UERT(R5),P.HARD(R5) ;CHECK IF HARD ERROR THRESHOLD EXCEEDED
6915 031712 103007 BHIS 1$ ;NO, CHECK IF DRIVE IS BEING ASSIGNED
6916 031714 032777 040020 147216 BIT #SW4!SW14,@SWR ;INHIBIT DRIVE DESELECTION
6917 031722 001003 BNE 1$ ;YES, CHECK IF DRIVE IS BEING ASSIGNED
6918 031724 104401 064445 TYPE ,ERR205 ;PRINT 'ERROR THRESHOLD EXCEEDED'
6919 031730 000413 BR 2$ ;GO DROP DRIVE
6920
6921 031732 105765 000210 1$: TSTB P.ASSN(R5) ;CHECK IF DRIVE IS BEING ASSIGNED
6922 031736 001421 BEQ 5$ ;NO, CLEAN UP ERROR
6923 031740 132765 000010 000210 BITB #BIT3,P.ASSN(R5) ;CHECK IF READ PACK SERIAL NUMBER
6924 031746 001404 BEQ 2$ ;NO, DROP DRIVE
6925 031750 132765 000200 000210 BITB #BIT7,P.ASSN(R5) ;CHECK IF RECAL FOR READ OF
6926 ; PACK SERIAL NUMBER
6927 031756 001441 BEQ 10$ ;NO, ISSUE RECALIBRATE AND TRY AGAIN
6928 031760 104401 064410 2$: TYPE ,ERR204 ;TYPE 'ERROR RECOVERY UNSUCCESSFUL'
6929 031764 004737 013162 JSR PC,DROP ;DROP DRIVE FROM TEST SEQUENCE
6930 031770 004737 022520 JSR PC,GETBUF ;GET NEW DRIVE, COMMAND, AND BUFFER
6931 031774 005037 001566 CLR ERRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
6932 032000 000207 RTS PC ;RETURN
6933
6934 032002 105765 000120 5$: TSTB P.IERC(R5) ;CHECK IF INHIBIT RETRY
6935 032006 001006 BNE 6$ ;YES, DO NOT PRINT MESSAGE
6936 032010 032777 020000 147122 BIT #SW13,@SWR ;CHECK IF INHIBIT PRINT OUT
6937 032016 001002 BNE 6$ ;YES, CONTINUE
6938 032020 104401 064410 TYPE ,ERR204 ;TYPE 'ERROR RECOVERY UNSUCCESSFUL'
6939 032024 010446 MOV R4,-(SP) ;STORE R4 ON STACK
6940 032026 116504 000000 6$: MOV P.DRVN(R5),R4 ;GET DRIVE NUMBER
6941 032032 156437 001514 001513 BISB INTMSK(R4),O.OVER ;INDICATE IMPLIED SEEKS ARE ALLOWED
6942 032040 012604 MOV (SP)+,R4 ;RESTORE R4
6943 032042 004037 051564 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN DRIVE
6944 032046 001314 AVAILQ ; AVAILIABLE QUEUE
6945 032050 004737 022520 JSR PC,GETBUF ;GET NEW DRIVE, COMMAND, AND BUFFER
6946 032054 005037 001566 CLR ERRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
6947 032060 000207 RTS PC ;RETURN
6948
6949 032062 122765 000010 000126 10$: CMPB #8.,P.RSEC(R5) ;CHECK IF ALL 16-BIT MODE SECTORS HAVE BEEN TRIED
6950 032070 101733 BLOS 2$ ;YES, DROP DRIVE
6951 032072 010446 MOV R4,-(SP) ;STORE R4 ON STACK
6952 032074 116504 000000 MOV P.DRVN(R5),R4 ;GET DRIVE NUMBER FOR INDEX
6953 032100 156437 001514 001513 CISB INTMSK(R4),O.OVER ;INDICATE THAT IMPLIED SEEKS ARE ALLOWED
6954 032106 012604 MOV (SP)+,R4 ;RESTORE R4
6955 032110 152765 000200 000210 BISB #BIT7,P.ASSN(R5) ;INDICATE THAT RECALIBRATE HAS BEEN ISSUED
6956 032116 005065 000146 CLR P.RECT(R5) ;CLEAR ERROR FLAGS
6957 032122 005065 000150 CLR P.DSTT(R5)
6958 032126 005065 000212 CLR P.ERR(R5)
6959 032132 112765 000113 000001 MOVB #RECAL,P.CMND(R5) ;LOAD RECALIBRATE COMMAND
6960 032140 004037 051564 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
6961 032144 001324 CINITQ ; INITIATION QUEUE
6962 032146 004737 022520 JSR PC,GETBUF ;GO ALLOCATE MEMORY BUFFER
    
```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 N 12
UNSUCCESSFUL RECOVERY CLEANUP PAGE 159

SEQ 0156

6963 032152 005037 001566
6964 032156 000207

CLR ERRPRO
RTS PC

:CLEAR ERROR PROCESSING IN PROGRESS
:RETURN

```

6965      .SBTTL  ERROR RECOVERY SEQUENCE
6966
6967 032160 032765 000200 000212 ERVY: BIT   #BIT7,P.ERR(R5) ;CHECK IF UNSOLICITED INTERRUPT
6968 032166 001441          BEQ   3$          ;NO, CHECK IF ERROR INFORMATION STILL VALID
6969 032170 032777 002000 146742 BIT   #SW10,@SWR   ;CHECK IF BELL ON ERROR
6970 032176 001402          BEQ   1$          ;NO, CONTINUE PROCESSING
6971 032200 104401 001160          TYPE  ,SBELL      ;RING BELL
6972 032204 032777 020000 146726 1$: BIT   #SW13,@SWR   ;CHECK IF INHIBIT PRINT OUT
6973 032212 001020          BNE   2$          ;YES, CHECK IF HALT ON ERROR
6974 032214 116537 000000 056755 MOVB  P.DRVN(R5),OPR011 ;LOAD DRIVE NUMBER FOR PRINT OUT
6975 032222 152737 000060 056755 BISB  #60,OPR011    ;MAKE IT ASCII
6976 032230 104401 056736          TYPE  ,OPR010    ;TYPE "DRIVE N"
6977 032234 004737 044046          JSR   PC,PRITIM   ;PRINT TIME
6978 032240 104401 062261          TYPE  ,ERR026    ;TYPE "INT FORM DRV NOT UNDER TEST"
6979 032244 004737 026252          JSR   PC,PRISTT  ;PRINT CONTROLLER REGISTERS
6980 032250 004737 026076          JSR   PC,PRDSTT  ;PRINT DRIVE STATUS
6981 032254 004737 031426          2$: JSR   PC,HLT000  ;CHECK IF HALT ON ERROR
6982 032260 005065 000212          CLR   P.ERR(R5)  ;CLEAR ERROR FLAGS
6983 032264 005037 001566          CLR   ERRPRO    ;CLEAR ERROR BEING PROCESSED
6984 032270 000207          RTS   PC        ;RETURN
6985
6986 032272 032765 000020 000212 3$: BIT   #BIT4,P.ERR(R5) ;CHECK IF SERVICING DRIVE SEIZED TIME OUT
6987 032300 001427          BEQ   5$          ;NO, CONTINUE
6988 032302 004737 030156          JSR   PC,PRI000  ;PRINT HEADER
6989 032306 104401 063351          TYPE  ,ERR059    ;PRINT "TIME OUT BECAUSE DRIVE SEIZED"
6990 032312 032777 020000 146620 BIT   #SW13,@SWR   ;CHECK IF INHIBIT PRINT OUT
6991 032320 001002          BNE   4$          ;YES, DO NOT PRINT CURRENT GENERATED COMMAND
6992 032322 004737 030422          JSR   PC,PRI002  ;PRINT CURRENT GENERATED COMMAND
6993 032326 005265 000206          4$: INC   P.NER(R5)  ;INCREMENT OTHER ERRORS
6994 032332 004737 031426          JSR   PC,HLT000  ;CHECK IF HALT ON ERROR
6995 032336 004737 023020          JSR   PC,BUFREL  ;RELEASE BUFFER
6996 032342 004737 013162          JSR   PC,DROP    ;DROP DRIVE
6997 032346 004737 022520          JSR   PC,GETBUF  ;GET NEW DRIVE, COMMAND, AND BUFFER
6998 032352 005037 001566          CLR   ERRPRO    ;CLEAR ERROR BEING PROCESSED
6999 032356 000207          RTS   PC        ;RETURN
7000
7001 032360 032765 000400 000212 5$: BIT   #BIT8,P.ERR(R5) ;CHECK IF TIME OUT BECAUSE HEADS DID NOT RELOAD
7002 032366 001404          BEQ   6$          ;NO, CONTINUE
7003 032370 004037 030642          JSR   R0,PRI100  ;PRINT TIME OUT BECAUSE HEADS DID NOT RELOAD
7004 032374 063704          ERR074
7005 032376 000207          RTS   PC        ;RETURN
7006
7007 032400 032765 000040 000212 6$: BIT   #BIT5,P.ERR(R5) ;CHECK IF TIME OUT DURING DRIVE POSITIONING
7008 032406 001425          BEQ   7$          ;NO, CONTINUE
7009 032410 042765 100000 000150 BIC   #BIT15,P.DSTT(R5) ;RESET ERROR ENQUEUED
7010 032416 052765 000100 000150 BIS   #BIT6,P.DSTT(R5) ;SET READ STATUS ISSUED
7011 032424 116537 000001 001666 MOVB  P.CMND(R5),POSCMD ;STORE COMMAND IN ERROR
7012 032432 042765 000002 000014 BIC   #DRVPOS,P.PRST(R5) ;RESET DRIVE POSITIONING
7013 032440 004737 037124          JSR   PC,GETCUR  ;STORE PRESENT STATUS FOR PRINT OUT
7014 032444 112765 000141 000001 MOVB  #RDSTAT,P.CMND(R5) ;GET DRIVE STATUS
7015 032452 004037 051564          JSR   R0,Q.PUSH  ;ENQUEUE COMMAND IN COMMAND
7016 032456 001324          CINITQ         ;INITIATION QUEUE
7017 032460 000207          RTS   PC        ;RETURN
7018
7019 032462 032765 000100 000212 7$: BIT   #BIT6,P.ERR(R5) ;CHECK IF ERROR WHILE ENQUEUED
7020 032470 001404          BEQ   8$          ;NO, CONTINUE

```

7021	032472	004037	030642			JSR	R0,PRI100	:PRINT HEADER
7022	032476	062141				ERR022		:TYPE 'ERROR WHILE WAITING TO
7023								: REPORT ERROR''
7024	032500	000207				RTS	PC	:RETURN
7025								
7026	032502	032765	004000	000016	8\$:	BIT	#CTO,P.CS1(R5)	:CHECK IF CONTROLLER TIME OUT
7027	032510	001404				BEQ	9\$:NO, CONTINUE
7028	032512	004037	037500			JSR	R0,CONERR	:GO REPORT ERROR
7029	032516	062053				ERR017		
7030	032520	000207				RTS	PC	:RETURN
7031								
7032	032522	032765	002000	000020	9\$:	BIT	#PGE,P.CS2(R5)	:CHECK IF PROGRAMMING ERROR
7033	032530	001404				BEQ	10\$:NO, CONTINUE
7034	032532	004037	037500			JSR	R0,CONERR	:PRINT HEADER
7035	032536	062107				ERR020		:TYPE 'PROG ERROR''
7036	032540	000207				RTS	PC	:RETURN
7037								
7038	032542	032765	000001	000034	10\$:	BIT	#ILC,P.ER(R5)	:CHECK IF ILLEGAL FUNCTION CODE
7039	032550	001404				BEQ	11\$:NO, CONTINUE
7040	032552	004037	037500			JSR	R0,CONERR	:PRINT HEADER
7041	032556	062122				ERR021		:TYPE 'ILLEGAL FUNCTION CODE''
7042	032560	000207				RTS	PC	:RETURN
7043								
7044	032562	032765	010000	000014	11\$:	BIT	#DRVSZD,P.PRST(R5)	:CHECK FOR DRIVE AVAIL LOST
7045	032570	001404				BEQ	12\$:BR IF NOT
7046	032572	004037	030642			JSR	R0,PRI100	:PRINT HEADER
7047	032576	062400				ERR029		:TYPE 'DRIVE BECAME NOT AVAIL''
7048	032600	000207				RTS	PC	:RETURN
7049								
7050	032602	032765	010000	000020	12\$:	BIT	#NED,P.CS2(R5)	:CHECK FOR NON-EXISTENT DRIVE
7051	032610	001404				BEQ	15\$:NO, CONTINUE
7052	032612	004037	030642			JSR	R0,PRI100	:PRINT HEADER
7053	032616	062206				ERR023		:TYPE 'NON-EXISTENT DRIVE''
7054	032620	000207				RTS	PC	:RETURN
7055								
7056	032622	032765	000400	000020	15\$:	BIT	#UFE,P.CS2(R5)	:CHECK IF UNIT FIELD ERROR
7057	032630	001404				BEQ	18\$:NO, CONTINUE
7058	032632	004037	030642			JSR	R0,PRI100	:PRINT HEADER
7059	032636	062241				ERR025		:TYPE 'UNIT FIELD ERROR''
7060	032640	000207				RTS	PC	:RETURN
7061								
7062	032642	032765	020000	000016	18\$:	BIT	#SPAR,P.CS1(R5)	:CHECK IF CONTROLLER DETECTED BAD PARITY
7063	032650	001004				BNE	19\$:YES, PRINT MESSAGE
7064	032652	032765	000010	000034		BIT	#DRPAR,P.ER(R5)	:CHECK IF DRIVE DETECTED BAD PARITY
7065	032660	001437				BEQ	25\$:NO, CONTINUE
7066	032662	004037	031032		19\$:	JSR	R0,PRI200	:PRINT HEADER
7067	032666	062225				ERR024		:PRINT 'SERCON PARITY''
7068	032670	004037	031456			JSR	R0,NERSTT	:TAKE STATISTICS FOR OTHER ERRORS
7069	032674	033142				60\$:ERROR RECOVERY SEQUENCE FINISHED RETURN
7070	032676	032765	020000	000016		BIT	#SPAR,P.CS1(R5)	:CHECK IF CONTROLLER DETECTD SERCON PARITY ERROR
7071	032704	001021				BNE	22\$:YES, GO RETRY COMMAND
7072	032706	032765	003400	000150		BIT	#BIT8!BIT9!BIT10,P.DSTT(R5)	:CHECK IF NO POSITIONING
7073	032714	001015				BNE	22\$:YES, RETRY COMMAND
7074	032716	132765	000007	000210		BITB	#BIT0!BIT1!BIT2,P.ASSN(R5)	:CHECK IF NO POSITION
7075								: DURING DRIVE ASSIGNMENT
7076	032724	001011				BNE	22\$:YES, RETRY COMMAND

7077	032726	042765	030000	000150		BIC	#BIT12!BIT13,P.DSTT(R5)	:RESET RE-SEEK AND RE-OFFSET
7078	032734	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	:SET RECALIBRATE DRIVE
7079	032742	112765	000113	000001		MOVB	#RECAL,P.CMND(R5)	:LOAD RECALIBRATE COMMAND
7080	032750	004037	051564		22\$:	JSR	RO,Q.PUSH	:PUT PARAMETER BLOCK IN COMMAND
7081	032754	001324				CINITQ		: INITIATION QUEUE
7082	032756	000207				RTS	PC	:RETURN
7083								
7084	032760	032765	004000	000020	25\$:	BIT	#NEM,P.CS2(R5)	:CHECK IF NON-EXISTENT MEMORY
7085	032766	001404				BEQ	30\$:NO, CONTINUE
7086	032770	004037	037500			JSR	RO,CONERR	:PRINT HEADER
7087	032774	062675				ERR044		:PRINT 'NON-EXISTENT MEMORY'
7088	032776	000207				RTS	PC	:RETURN
7089								
7090	033000	032765	020000	000020	30\$:	BIT	#UPE,P.CS2(R5)	:CHECK IF UNIBUS PARITY
7091	033006	001404				BEQ	35\$:NO, CONTINUE
7092	033010	004037	037500			JSR	RO,CONERR	:PRINT HEADER
7093	033014	062713				ERR045		:PRINT 'UNIBUS PARITY'
7094	033016	000207				RTS	PC	:RETURN
7095								
7096	033020	004037	037670		35\$:	JSR	RO,TSTDRP	:CHECK IF DROP DRIVE ERROR
7097	033024	033142				60\$:YES, ERROR RETURN
7098	033026	032765	000002	000034		BIT	#SKI,P.ER(R5)	:CHECK IF SEEK INCOMPLETE
7099	033034	001407				BEQ	45\$:NO, CONTINUE
7100	033036	004037	031032			JSR	RO,PR1200	:PRINT HEADER
7101	033042	062577				ERR038		:PRINT 'SEEK INCOMPLETE'
7102	033044	004037	031510			JSR	RO,SKISTT	:TAKE SEEK INCOMPLETE STATISTICS
7103	033050	033142				60\$:ERROR RECOVERY FINISHED RETURN
7104	033052	000412				BR	48\$:GO ISSUE RECALIBRATE
7105								
7106	033054	032765	000040	000036	45\$:	BIT	#DROT,P.DS(R5)	:CHECK IF DRIVE OFF TRACK
7107	033062	001420				BEQ	50\$:NO, CONTINUE
7108	033064	004037	031032			JSR	RO,PR1200	:PRINT HEADER
7109	033070	062614				ERR039		:PRINT 'DRIVE OFF TRACK'
7110	033072	004037	031456			JSR	RO,NERSTT	:TAKE STATISTICS ON OTHER ERRORS
7111	033076	033142				60\$:END OF ERROR RECOVERY RETURN
7112	033100	042765	034000	000150	48\$:	BIC	#BIT11!BIT12!BIT13,P.DSTT(R5)	:RESET STATUS
7113	033106	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	:SET RECALIBRATE ISSUED
7114	033114	112765	000113	000001		MOVB	#RECAL,P.CMND(R5)	:LOAD RECALIBRATE
7115	033122	000712				BR	22\$:ENQUEUE COMMAND
7116								
7117	033124	032765	001000	000034	50\$:	BIT	#COE,P.ER(R5)	:CHECK IF CYLINDER ADDRESS OVERFLOW
7118	033132	001404				BEQ	E0\$:NO, CONTINUE
7119	033134	004037	037500			JSR	RO,CONERR	:PRINTER HEADER
7120	033140	063013				ERR051		:PRINT 'CYLINDER ADDRESS OVERFLOW'
7121	033142	000207			60\$:	RTS	PC	:RETURN
7122								
7123	033144	005765	000022		E0\$:	TST	P.WCR(R5)	:CHECK IF FINAL WORD COUNT IS ZERO
7124	033150	001410				BEQ	5\$:YES, CONTINUE
7125	033152	026565	000022	000012		CMP	P.WCR(R5),P.WC(R5)	:CHECK THAT FINAL WORD COUNT IS LESS
7126								: THAN OR EQUAL TO INITIAL WORD COUNT
7127	033160	103004				BHIS	5\$:YES, CONTINUE
7128	033162	004037	037500			JSR	RO,CONERR	:TYPE 'WORD COUNT INVALID'
7129	033166	063567				ERR070		
7130	033170	000207				RTS	PC	:RETURN
7131								
7132	033172	032765	001400	000016	5\$:	BIT	#BA16!BA17,P.CS1(R5)	:CHECK IF BUS ADDRESS BITS 16 OR 17 SET

7133	033200	001015				BNE	8\$:YES, REPORT ERROR
7134	033202	026565	000024	000010		CMP	P.BAR(R5),P.BALO(R5)	:CHECK THAT FINAL BUS ADDRESS IS GREATER
7135								: THAN OR EQUAL TO INITIAL BUS ADDRESS
7136	033210	103411				BLO	8\$:NO, REPORT ERROR
7137	033212	016546	000012			MOV	P.WC(R5),-(SP)	:PUT INITIAL WORD COUNT ON STACK
7138	033216	005416				NEG	(SP)	:MAKE IT POSITIVE
7139	033220	006316				ASL	(SP)	:MULTIPLY BY 2
7140	033222	066516	000010			ADD	P.BALO(R5),(SP)	:CALCULATE MAXIMUM FINAL ADDRESS
7141	033226	026526	000024			CMP	P.BAR(R5),(SP)+	:MAKE SURE IT IS NOT TOO LARGE
7142	033232	101404				BLOS	10\$:GOOD ADDRESS CONTINUE
7143	033234	004037	037500		8\$:	JSR	RO,CONERR	:TYPE 'BUS ADDRESS INVALID'
7144	033240	063612				ERR071		
7145	033242	000207				RTS	PC	:RETURN
7146								
7147	033244	026565	000030	000002	10\$:	CMP	P.DCYL(R5),P.CYLN(R5)	:CHECK THAT FINAL CYLINDER IS GREATER
7148								: THAN OR EQUAL TO INITIAL CYLINDER
7149	033252	001405				BEQ	15\$:IF EQUAL CHECK TRACK/SECTOR
7150	033254	101014				BHI	E1\$:IF GREATER THAN, DETERMINE ERROR
7151	033256	004037	037500			JSR	RO,CONERR	:TYPE 'CYLIN ADDRESS INVALID'
7152	033262	063636				ERR072		
7153	033264	000207				RTS	PC	:RETURN
7154								
7155	033266	026565	000026	000004	15\$:	CMP	P.DTS(R5),P.SECT(R5)	:CHECK TRACK/SECTOR GREATER THAN OR
7156								: EQUAL TO INITIAL VALUES
7157	033274	103004				BHIS	E1\$:YES, GO INTO ERROR RECOVERY
7158	033276	004037	037500			JSR	RO,CONERR	:TYPE 'SECTOR/TRACK INVALID'
7159	033302	063657				ERR073		
7160	033304	000207				RTS	PC	
7161								
7162	033306	032765	010000	000034	E1\$:	BIT	#DTE,P.ER(R5)	:CHECK FOR DRIVE TIMING ERROR
7163	033314	001002				BNE	2\$:YES, PROCESS DRIVE TIMING ERROR
7164	033316	000137	034010			JMP	E2\$:NO, CONTINUE
7165								
7166	033322	105765	000147		2\$:	TSTB	P.RERD(R5)	:CHECK IF FIRST DATA ERROR IN
7167								: RETRY SEQUENCE
7168	033326	001117				BNE	30\$:NO, REPORT ERROR
7169	033330	016537	000010	001634		MOV	P.BALO(R5),RTYBA	:STORE SUPPLIED BUS ADDRESS
7170	033336	016537	000012	001636		MOV	P.WC(R5),RTYWC	:STORE SUPPLIED WORD COUNT
7171	033344	005437	001636			NEG	RTYWC	:MAKE IT POSITIVE
7172	033350	022737	000400	001636		CMP	#256.,RTYWC	:CHECK IF LESS THAN OR EQUAL TO A SECTOR
7173	033356	103013				BHIS	21\$:YES, WORK WITH ONE SECTOR
7174	033360	026565	000026	000004		CMP	P.DTS(R5),P.SECT(R5)	:CHECK IF STARTING TRACK AND SECTOR
7175	033366	001021				BNE	25\$:NOT RETRY 2 SECTORS
7176	033370	026565	000030	000002		CMP	P.DCYL(R5),P.CYLN(R5)	:CHECK IF STARTING CYLINDER
7177	033376	001015				BNE	25\$:NO, RETRY 2 SECTORS
7178	033400	012737	000400	001636		MOV	#256.,RTYWC	:SET WORD COUNT = 1 SECTOR
7179	033406	112737	000001	001644	21\$:	MOVB	#1,RTYSCN	:SET SECTOR COUNT = 1
7180	033414	016537	000004	001640		MOV	P.SECT(R5),RTYSEC	:LOAD SECTOR AND TRACK
7181	033422	016537	000002	001642		MOV	P.CYLN(R5),RTYCYL	:LOAD CYLINDER
7182	033430	000456				BR	30\$:GO REPORT ERROR
7183								
7184	033432	016537	000026	001640	25\$:	MOV	P.DTS(R5),RTYSEC	:SET SECTOR AND TRACK
7185	033440	016537	000030	001642		MOV	P.DCYL(R5),RTYCYL	:GET CYLINDER
7186	033446	105737	001640			TSTB	RTYSEC	:CHECK IF SECTOR = 0
7187	033452	001403				BEQ	26\$:YES, GO DECREMENT TRACK
7188	033454	105337	001640			DECB	RTYSEC	:DECREMENT SECTOR

7189	033460	000416			BR	28\$:DETERMINE WORD COUNT
7190							
7191	033462	112737	000025	001640	26\$: MOVB	#21.,RTYSEC	:LOAD SECTOR = 21
7192	033470	105737	001641		TSTB	RTYTRK	:CHECK IF TRACK 0
7193	033474	001403			BEQ	27\$:YES, GO DECREMENT CYLINDER
7194	033476	105337	001641		DECB	RTYTRK	:DECREMENT TRACK
7195	033502	000405			BR	28\$:DETERMINE WORD COUNT
7196							
7197	033504	112737	000002	001641	27\$: MOVB	#2,RTYTRK	:LOAD TRACK = 2
7198	033512	005337	001642		DEC	RTYCYL	:DECREMENT CYLINDER
7199	033516	004737	037366		28\$: JSR	PC,CALWC	:CALCULATE RETRY WORD COUNT
7200	033522	112737	000002	001644	MOVB	#2,RTYSCN	:LOAD SECTOR COUNT = 2
7201	033530	022737	001000	001636	CMP	#512.,RTYWC	:CHECK IF LESS THAN OR EQUAL
7202	033536	101004			BHI	29\$: TWO SECTORS
7203	033540	012737	001000	001636	MOV	#512.,RTYWC	:MAKE IT 512
7204	033546	000407			BR	30\$:GO REPORT ERROR
7205							
7206	033550	022737	000400	001636	29\$: CMP	#256.,RTYWC	:CHECK IF ONLY ONE SECTOR
7207	033556	103403			BLO	30\$:NO, GO REPORT ERROR
7208	033560	112737	000001	001644	MOVB	#1,RTYSCN	:LOAD SECTOR COUNT = 1
7209							
7210	033566	004037	031032		30\$: JSR	R0,PR1200	:PRINT HEADER
7211	033572	062726			ERR046		:PRINT DRIVE TIMING ERROR
7212	033574	105765	000120		TSTB	P.IERC(R5)	:CHECK IF INHIBIT ERROR RECOVERY
7213	033600	001016			BNE	35\$:YES, GO CLEAN UP FOR NEXT COMMAND
7214	033602	105265	000147		INCB	P.RERD(R5)	:INCREMENT REREAD COUNT
7215	033606	122765	000043	000147	CMPB	#43,P.RERD(R5)	:CHECK IF RETRY UNSUCCESSFUL
7216	033614	101410			BLOS	35\$:YES, GO INDICATE UNSUCCESSFUL
7217							:RECOVERY
7218	033616	122765	000121	000123	CMPB	#RDDATA,P.RCMD(R5)	:CHECK IF READ DATA
7219	033624	001407			BEQ	37\$:YES, GO TO RECOVERY SEQUENCE
7220	033626	122765	000021	000147	CMPB	#21,P.RERD(R5)	:CHECK IF RETRY UNSUCCESSFUL
7221	033634	101026			BHI	40\$:NO, GO TO RETRY SEQUENCE
7222	033636	004737	031660		35\$: JSR	PC,UNRECY	:CLEAN UP FOR UNSUCCESSFUL RECOVERY
7223	033642	000207			RTS	PC	:RETURN
7224							
7225	033644	122765	000021	000147	37\$: CMPB	#21,P.RERD(R5)	:CHECK IF OFFSET +400 MICRO-INCHES
7226	033652	001017			BNE	40\$:NO, CONTINUE RECOVERY
7227	033654	122737	000001	001644	CMPB	#1,RTYSCN	:CHECK IF NUMBER OF SECTORS = 1
7228	033662	001404			BEQ	38\$:YES, DO OFFSET
7229	033664	122737	000025	001640	CMPB	#21.,RTYSEC	:CHECK IF MID-TRANSFER SEEK
7230							: IS REQUIRED
7231	033672	001761			BEQ	35\$:YES, INDICATE UNSUCCESSFUL RETRY
7232	033674	010446			38\$: MOV	R4,-(SP)	:STORE R4 ON THE STACK
7233	033676	116504	000000		MOVB	P.DRVN(R5),R4	:GET DRIVE NUMBER FOR INDEX
7234	033702	146437	001514	001513	BICB	INTMSK(R4),O.OVER	:DO NOT SEEK BEFORE DATA
7235							: TRANSFER (RESETTING OFFSET)
7236	033710	012604			MOV	(SP)+,R4	:RESTORE R4
7237							
7238	033712	004037	037204		40\$: JSR	R0,DETOFF	:DETERMINE IF OFFSET INSTRUCTION
7239	033716	034006			50\$:OFFSET ISSUED RETURN
7240	033720	116565	000123	000001	MOVB	P.RCMD(R5),P.CMND(R5)	:LOAD COMMAND
7241	033726	013765	001640	000004	MOV	RTYSEC,P.SECT(R5)	:LOAD SECTOR AND TRACK
7242	033734	013765	001642	000002	MOV	RTYCYL,P.CYLN(R5)	:LOAD CYLINDER
7243	033742	013765	001634	000010	MOV	RTYBA,P.BALO(R5)	:LOAD BUS ADDRESS
7244	033750	013765	001636	000012	MOV	RTYWC,P.WC(R5)	:LOAD WORD COUNT

7245	033756	005465	000012			NEG	P.WC(R5)	:MAKE IT NEGATIVE
7246	033762	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK
7247	033770	001003				BNE	45\$:NO, ISSUE COMMAND
7248	033772	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	:YES, SET ISSUE WRITE COMMAND
7249	034000	004037	051564		45\$:	JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN
7250	034004	001324				CINITQ		: COMMAND INITIATION QUEUE
7251	034006	000207			50\$:	RTS	PC	:RETURN
7252								
7253	034010	032765	020400	000034	E2\$:	BIT	#OPI!HVRC,P.ER(R5)	:CHECK IF HEADER TYPE ERROR
7254	034016	001002				BNE	2\$:YES, PROCESS ERROR
7255	034020	000137	034346			JMP	E3\$:NO, CONTINUE
7256								
7257	034024	004737	037124		2\$:	JSR	PC,GETCUR	:GET CURRENT REGISTER STATUS
7258	034030	112737	000001	001644		MOVB	#1,RTYSCN	:LOAD RETRY SECTOR COUNT TO 1
7259	034036	016537	000026	001640		MOV	P.DTS(R5),RTYSEC	:STORE CURRENT SECTOR AND TRACK
7260	034044	016537	000030	001642		MOV	P.DCYL(R5),RTYCYL	:STORE CURRENT SECTOR
7261	034052	016537	000010	001634		MOV	P.BALO(R5),RTYBA	:GET INITIAL BUS ADDRESS
7262	034060	016537	000012	001636		MOV	P.WC(R5),RTYWC	:GET INITIAL WORD COUNT
7263	034066	005437	001636			NEG	RTYWC	:MAKE IT POSITIVE
7264	034072	022737	000400	001636		CMP	#256.,RTYWC	:CHECK IF LESS THEN OR EQUAL TO ONE SECTOR
7265	034100	103021				BHIS	20\$:RETRY COMMAND
7266	034102	026565	000026	000004		CMP	P.DTS(R5),P.SECT(R5)	:CHECK IF CURRENT SECTOR AND TRACK EQUALS
7267								: INITIAL TRACK AND SECTOR
7268	034110	001004				BNE	15\$:NO, CALCULATE DESIRED SECTOR
7269	034112	026565	000030	000002		CMP	P.DCYL(R5),P.CYLN(R5)	:CHECK IF CURRENT CYLINDER EQUALS
7270								: INITIAL CYLINDER
7271	034120	001406				BEQ	18\$:YES, USE INITIAL CYLINDER, TRACK, AND SECTOR
7272	034122	004737	037366		15\$:	JSR	PC,CALWC	:CALCULATE WORD COUNT
7273	034126	022737	000400	001636		CMP	#256.,RTYWC	:CHECK IF WORD COUNT LESS THAN OR EQUAL
7274								: TO ONE SECTOR
7275	034134	103003				BHIS	20\$:YES, USE CALCULATED WORD COUNT
7276	034136	012737	000400	001636	18\$:	MOV	#256.,RTYWC	:USE 256 WORDS
7277	034144	016546	000052		20\$:	MOV	P.B10(R5),-(SP)	:STORE DRIVE CYLINDER ADDRESS
7278	034150	042716	160017			BIC	#^C<M.CADD>,(SP)	:THROW AWAY PARITY AND MESSAGE TYPE
7279	034154	006216				ASR	(SP)	:SHIFT 4 BITS RIGHT
7280	034156	006216				ASR	(SP)	
7281	034160	006216				ASR	(SP)	
7282	034162	006216				ASR	(SP)	
7283	034164	022637	001642			CMP	(SP)+,RTYCYL	:CHECK IF CYLINDER IS CORRECT
7284	034170	001420				BEQ	30\$:YES, ISSUE READ HEADER
7285	034172	004037	031032			JSR	RO,PR1200	:REPORT ERROR
7286	034176	062631				ERR040		:TYPE 'MISPOS'
7287	034200	004037	031524			JSR	RO,OP1STT	:GET OPI/MISPOSITIONING STATISTICS
7288	034204	034344				60\$:ERROR THRESHOLD EXCEEDED RETURN
7289	034206	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	:SET RECALIBRATE ISSUED
7290	034214	112765	000113	000001		MOVB	#RECAL,P.CMND(R5)	:LOAD RECALIBRATE IN PARAMETER BLOCK
7291	034222	004037	051564			JSR	RO,Q.PUSH	:PUT PARAMETER BLOCK IN COMMAND
7292	034226	001324				CINITQ		: INITIATION QUEUE
7293	034230	000207				RTS	PC	:RETURN
7294								
7295	034232	010446			30\$:	MOV	R4, -(SP)	:STORE R4 ON THE STACK
7296	034234	113704	001641			MOVB	RTYTRK,R4	:GET TRACK FOR INDEX
7297	034240	136465	001507	000057		BITB	H.HEAD(R4),P.B11+1(R5)	:CHECK IF CORRECT HEAD IS SELECTED
7298	034246	001013				BNE	35\$:YES, CONTINUE
7299	034250	012604				MOV	(SP)+,R4	:RESTORE R4
7300	034252	004037	031032			JSR	RO,PR1200	:PRINT HEAD SELECT ERROR

7301	034256	063503				ERR065		
7302	034260	004037	031524			JSR	RO,OPISTT	:RECORD AS AN OPI
7303	034264	034344				60\$:ERROR THRESHOLD EXCEEDED RETURN
7304	034266	004037	051564			JSR	RO,Q.PUSH	:REISSUE COMMAND
7305	034272	001324				CINITQ		
7306	034274	000207				RTS	PC	:RETURN
7307								
7308	034276	012604			35\$:	MOV	(SP)+,R4	:RESTORE R4
7309	034300	052765	000020	000150		BIS	#BIT4,P.DSTT(R5)	:SET READ ALL HEADERS ISSUED
7310	034306	013765	001640	000004		MOV	RTYSEC,P.SECT(R5)	:LOAD SECTOR AND TRACK
7311	034314	013765	001642	000002		MOV	RTYCYL,P.CYLN(R5)	:LOAD CYLINDER
7312	034322	012765	066404	000010		MOV	#HDBUFF,P.BALO(R5)	:LOAD ADDRESS OF HEADER BUFFER
7313	034330	112765	000164	000001		MOVB	#RDALHD,P.CMND(R5)	:LOAD COMMAND
7314	034336	004037	051564			JSR	RO,Q.PUSH	:PUT PARAMETER BLOCK IN COMMAND
7315	034342	001324				CINITQ		: INITIATION QUEUE
7316	034344	000207			60\$:	RTS	PC	:RETURN
7317								
7318	034346	032765	000200	000034	E3\$:	BIT	#BSE,P.ER(R5)	:CHECK FOR BAD SECTOR ERROR
7319	034354	001403				BEQ	5\$:NO, CONTINUE
7320	034356	004737	042534			JSR	PC,BDSECT	:SERVICE BAD SECTOR
7321	034362	000207				RTS	PC	:RETURN
7322								
7323	034364	032765	100000	000020	5\$:	BIT	#DLT,P.CS2(R5)	:CHECK IF DATA LATE
7324	034372	001444				BEQ	E4\$:NO, CONTINUE
7325	034374	004737	030156			JSR	PC,PR1000	:TYPE HEADER
7326	034400	032777	020000	144532		BIT	#SW13,@SWR	:CHECK IF INHIBIT PRINT OUT
7327	034406	001004				BNE	10\$:YES, CHECK IF HALT ON ERROR
7328	034410	104401	062746			TYPE	,ERR047	:TYPE "DATA LATE"
7329	034414	004737	026252			JSR	PC,PR1STT	:PRINT CONTROLLER STATUS
7330	034420	004737	031426		10\$:	JSR	PC,HLT000	:CHECK IF HALT ON ERROR
7331	034424	005237	001626			INC	DLTCNT	:INCREMENT DATA LATE COUNT
7332	034430	022737	001000	001626		CMP	#T.DLT,DLTCNT	:CHECK IF THRESHOLD EXCEEDED
7333	034436	103006				BHIS	15\$:NO, REISSUE COMMAND
7334	034440	104401	061131			TYPE	,ERR000	:TYPE FATAL ERROR
7335	034444	104401	061534			TYPE	,ERR009	:TYPE DATA LATE THRESHOLD EXCEEDED
7336	034450	000000				HALT		:HANG
7337	034452	000776				BR	.-2	
7338								
7339	034454	105765	000150		15\$:	TSTB	P.DSTT(R5)	:CHECK IF FIRST ERROR
7340	034460	100005				BPL	20\$:NO, REISSUE COMMAND
7341	034462	042765	100200	000150		BIC	#BIT15!BIT7,P.DSTT(R5)	:CLEAR ERROR ENQUEUED AND FIRST ERROR
7342	034470	005037	001566			CLR	ERRPRO	:CLEAR ERROR RECOVERY IN PROGRESS
7343	034474	004037	051564		20\$:	JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
7344	034500	001324				CINITQ		: INITIATION QUEUE
7345	034502	000207				RTS	PC	:RETURN
7346								
7347	034504	032765	100000	000034	E4\$:	BIT	#DCK,P.ER(R5)	:CHECK IF DATA CHECK
7348	034512	001002				BNE	2\$:YES, START RECOVERY
7349	034514	000137	035314			JMP	E5\$:NO, CONTINUE
7350								
7351	034520	112737	000001	001644	2\$:	MOVB	#1,RTYSCN	:SET SECTOR COUNT = 1
7352	034526	016537	000004	001640		MOV	P.SECT(R5),RTYSEC	:STORE RETRY SECOTOR AND TRACK
7353	034534	016537	000002	001642		MOV	P.CYLN(R5),RTYCYL	:STORE RETRY CYLINDER
7354	034542	016537	000010	001634		MOV	P.BALO(R5),RTYBA	:STORE RETRY BUS ADDRESS
7355	034550	016537	000012	001636		MOV	P.WC(R5),RTYWC	:STORE RETRY WORD COUNT
7356	034556	005437	001636			NEG	RTYWC	:MAKE IT POSITIVE

Line	Address	Track	Block	Label	Op	Opnd	Comment
7357	034562	022737	000400	001636	CMP	#256.,RTYWC	:CHECK IF LESS THAN OR EQUAL TO A SECTOR
7358	034570	103043			BHIS	15\$:YES, CHECK IF ECC CORRECTION
7359	034572	016537	000026	001640	MOV	P.DTS(R5),RTYSEC	:STORE PRESENT TRACK AND SECTOR
7360	034600	016537	000030	001642	MOV	P.DCYL(R5),RTYCYL	:STORE PRESENT CYLINDER
7361	034606	105737	001640		TSTB	RTYSEC	:CHECK IF SECTOR = 0
7362	034612	001403			BEQ	11\$:YES, DECREMENT TRACK
7363	034614	105337	001640		DECB	RTYSEC	:DECREMENT SECTOR
7364	034620	000416			BR	13\$:GO CALCULATE WORD COUNT
7365							
7366	034622	112737	000025	001640	11\$:	MOVB #21.,RTYSEC	:SET SECTOR = 21
7367	034630	105737	001641		TSTB	RTYTRK	:CHECK IF TRACK = 0
7368	034634	001403			BEQ	12\$:YES, DECREMENT CYLINDER
7369	034636	105337	001641		DECB	RTYTRK	:DECREMENT TRACK
7370	034642	000405			BR	13\$:GO CALCULATE WORD COUNT
7371							
7372	034644	112737	000002	001641	12\$:	MOVB #2,RTYTRK	:SET TRACK = 2
7373	034652	005337	001642		DEC	RTYCYL	:DECREMENT CYLINDER
7374	034656	004737	037366		13\$:	JSR PC,CALWC	:CALCULATE WORD COUNT
7375	034662	022737	000400	001636		CMP #256.,RTYWC	:CHECK IF LESS THAN OR EQUAL TO ONE SECTOR
7376	034670	103003			BHIS	15\$:YES, PERFORM ECC CORRECTION
7377	034672	012737	000400	001636		MOV #256.,RTYWC	:MAKE IT ONE SECTOR
7378	034700	004037	031032		15\$:	JSR RO,PR1200	:PRINT HEADER
7379	034704	062762			ERR048		:PRINT DATA CHECK
7380	034706	105765	000120		TSTB	P.IERC(R5)	:CHECK IF INHIBIT ERROR RECOVERY
7381	034712	001123			BNE	40\$:YES, INDICATE UNSUCCESSFUL RECOVERY
7382	034714	122765	000121	000123	CMPB	#RDDATA,P.RCMD(R5)	:CHECK IF READ DATA
7383	034722	001101			BNE	30\$:NO, DO NOT DO ECC CORRECTION
7384	034724	032765	000100	000034	BIT	#ECH,P.ER(R5)	:CHECK IF CORRECTABLE
7385	034732	001075			BNE	30\$:NO, DO NOT DO ECC CORRECTION
7386	034734	022765	010040	000060	CMP	#10040,P.EPOS(R5)	:CHECK IF ECC POSITION TOO LARGE
7387	034742	101406			BLOS	16\$:YES, REPORT ERROR
7388	034744	005765	000060		TST	P.EPOS(R5)	:CHECK IF ECC POSITION = 0
7389	034750	001403			BEQ	16\$:YES, REPORT ERROR
7390	034752	005765	000062		TST	P.EPAT(R5)	:CHECK IF ECC PATTERN = 0
7391	034756	001004			BNE	17\$:NO, DO ECC CORRECTION
7392	034760	004037	037500		16\$:	JSR RO,CONERR	:TYPE 'ECC LOGIC ERROR'
7393	034764	063526			ERR066		
7394	034766	000207			RTS	PC	:RETURN
7395							
7396	034770	004737	026336		17\$:	JSR PC,ECCOR	:DO ECC CORRECTION
7397	034774	105765	000210		TSTB	P.ASSN(R5)	:CHECK IF IN ASSIGNMENT SEQUENCE
7398	035000	001017			BNE	18\$:YES, DO NOT CHECK ECC CORRECTION
7399	035002	013765	001634	000214	MOV	RTYBA,P.ERHD(R5)	:LOAD START OF SECTOR FOR
7400							:DATA COMPARISON
7401	035010	013765	001636	000222	MOV	RTYWC,P.BFCP(R5)	:LOAD NUMBER OF WORDS OF
7402	035016	013765	001636	000220	MOV	RTYWC,P.CMLG(R5)	:SECTOR PRESENT
7403	035024	112765	177777	000117	MOVB	#-1,P.ECMP(R5)	:SET ECC COMPARE FLAG
7404	035032	004737	023540		JSR	PC,CMPBUF	:CHECK ECC CORRECTION
7405	035036	035046			20\$:INDICATE ECC MISCORRECTION
7406	035040	004737	036454		18\$:	JSR PC,SURECY	:INDICATE SUCCESSFUL RECOVERY
7407	035044	000207			RTS	PC	:RETURN
7408							
7409	035046	032777	020000	144064	20\$:	BIT #SW13,@SWR	:CHECK IF INHIBIT ERROR PRINT OUT
7410	035054	001024			BNE	30\$:YES, GO TO RECOVERY SEQUENCE
7411	035056	104401	063422		TYPE	,ERR060	:TYPE 'ECC MISCORRECTION'
7412	035062	104401	064701		TYPE	,ERR211	:TYPE 'RKPOS'

7413	035066	016546	000060			MOV	P.EPOS(R5),-(SP)	:STORE RKPOS FOR PRINT OUT
7414	035072	004737	053224			JSR	PC,BINOCT	:CONVERT TO OCTAL
7415	035076	104401	001165			TYPE	,\$CRLF	:TYPE <CR><LF>
7416	035102	104401	064710			TYPE	,ERR212	:TYPE 'RKPAT'
7417	035106	016546	000062			MOV	P.EPAT(R5),-(SP)	:STORE RKPAT FOR PRINT OUT
7418	035112	004737	053224			JSR	PC,BINOCT	:COVERT TO OCTAL
7419	035116	104401	001165			TYPE	,\$CRLF	:TYPE <CR><LF>
7420	035122	004737	036202			JSR	PC,PRERWD	:PRINT WORD IN ERROR
7421	035126	105265	000147		30\$:	INCB	P.RERD(R5)	:INCREMENT RETRY COUNT
7422	035132	122765	000043	000147		CMPB	#43,P.RERD(R5)	:CHECK IF UNSUCCESSFUL RETRY
7423	035140	101410				BLOS	40\$:YES, INDICATE UNSUCCESSFUL RETRY
7424	035142	122765	000121	000123		CMPB	#RDDATA,P.RCMD(R5)	:CHECK IF READ DATA
7425	035150	001407				BEQ	45\$:YES, CHECK FOR OFFSET
7426	035152	122765	000021	000147		CMPB	#21,P.RERD(R5)	:CHECK IF ABOUT TO ENTER OFFSET OPERATIONS
7427	035160	101021				BHI	50\$:NO, DO REREAD
7428	035162	004737	031660		40\$:	JSR	PC,UNRECY	:INDICATE UNSUCCESSFUL RECOVERY
7429	035166	000207				RTS	PC	:RETURN
7430								
7431	035170	122765	000021	000147	45\$:	CMPB	#21,P.RERD(R5)	:CHECK IF OFFSET +400 MICRO-INCHES
7432	035176	001007				BNE	48\$:NO, GO DETERMINE IF OFFSET IS TO
7433								: BE ISSUED
7434	035200	010446				MOV	R4, -(SP)	:STORE R4 ON STACK
7435	035202	116504	000000			MOV	P.DRVN(R5),R4	:GET DRIVE NUMBER AS INDEX
7436	035206	146437	001514	001513		BICB	INTMSK(R4),O.OVER	:RESET ISSUE IMPLIED SEEK
7437	035214	012604				MOV	(SP)+,R4	:RESTORE R4
7438	035216	004037	037204		48\$:	JSR	RO,DETOFF	:DETERMINE IF OFFSET IS TO BE ISSUED
7439	035222	035312					60\$:OFFSET ISSUED RETURN
7440	035224	116565	000123	000001	50\$:	MOV	P.RCMD(R5),P.CMND(R5)	:REISSUE LAST COMMAND
7441	035232	013765	001640	000004		MOV	RTYSEC,P.SECT(R5)	:LOAD SECTOR AND TRACK
7442	035240	013765	001642	000002		MOV	RTYCYL,P.CYLN(R5)	:LOAD CYLINDER
7443	035246	013765	001634	000010		MOV	RTYBA,P.BALO(R5)	:LOAD BUS ADDRESS
7444	035254	013765	001636	000012		MOV	RTYWC,P.WC(R5)	:LOAD WORD COUNT
7445	035262	005465	000012			NEG	P.WC(R5)	:MAKE IT NEGATIVE
7446	035266	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK
7447	035274	001003				BNE	55\$:NO, GO ISSUE COMMAND
7448	035276	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	:ISSUE WRITE BEFORE WRITE CHECK
7449	035304	004037	051564		55\$:	JSR	RO,Q.PUSH	:PUT PARAMETER BLOCK ON COMMAND
7450	035310	001324				CINITQ		:COMMAND INITIATION QUEUE
7451	035312	000207			60\$:	RTS	PC	:RETURN
7452								
7453	035314	032765	040000	000020	E5\$:	BIT	#WCE,P.CS2(R5)	:CHECK IF WRITE CHECK ERROR
7454	035322	001002				BNE	1\$:YES, PROCESS WRITE CHECK ERROR
7455	035324	000137	035466			JMP	E10\$:NO, CONTINUE
7456								
7457	035330	004037	031032		1\$:	JSR	RO,PR1200	:PRINT WRITE CHECK ERROR
7458	035334	062774				ERR050		
7459	035336	105765	000120			TSTB	P.IERC(R5)	:CHECK IF INHIBIT ERROR CORRECTION AND RETRY
7460	035342	001006				BNE	5\$:YES, INITIATE NEXT COMMAND
7461	035344	105265	000147			INCB	P.RERD(R5)	:INCREMENT RETRY COUNT
7462	035350	122765	000021	000147		CMPB	#21,P.RERD(R5)	:CHECK IF 16 REREADS OCCURRED
7463	035356	101003				BHI	10\$:NO, START RECOVERY SEQUENCE
7464	035360	004737	031660		5\$:	JSR	PC,UNRECY	:INDICATE UNSUCCESSFUL RECOVERY
7465	035364	000207				RTS	PC	:RETURN
7466								
7467	035366	016537	000024	001634	10\$:	MOV	P.BAR(R5),RTYBA	:GET ADDRESS OF WORD IN ERROR
7468	035374	162737	000002	001634		SUB	#2,RTYBA	:ADJUST ADDRESS

7516	035662	122765	000021	000147	20\$:	CMPB	#21,P.RERD(R5)	:CHECK IF ENTERING OFFSET
7517	035670	001007				BNE	25\$:NO, DETERMINE IF TIME TO CHANGE OFFSET
7518	035672	010446				MOV	R4,-(SP)	:STORE R4 ON STACK
7519	035674	116504	000000			MOVB	P.DRVN(R5),R4	:GET DRIVE NUMBER FOR INDEX
7520	035700	146437	001514	001513		BICB	INTMSK(R4),O.OVER	:RESET SEEK BEFORE DATA TRANSFER
7521	035706	012604				MOV	(SP)+,R4	:RESTORE R4
7522	035710	004037	037204		25\$:	JSR	R0,DETOFF	:DETERMINE IF OFFSET TO BE ISSUED
7523	035714	035766				50\$:OFFSET HAS BEEN ISSUED
7524	035716	116565	000123	000001		MOVB	P.RCMD(R5),P.CMND(R5)	:GET LAST ISSUED COMMAND
7525	035724	013765	001640	000004		MOV	RTYSEC,P.SECT(R5)	:LOAD SECTOR AND TRACK ADDRESS
7526	035732	013765	001642	000002		MOV	RTYCYL,P.CYLN(R5)	:LOAD CYLINDER ADDRESS
7527	035740	013765	001634	000010		MOV	RTYBA,P.BALO(R5)	:LOAD BUS ADDRESS
7528	035746	013765	001636	000012		MOV	RTYWC,P.WC(R5)	:GET WORD COUNT
7529	035754	005465	000012			NEG	P.WC(R5)	:MAKE IT NEGATIVE
7530	035760	004037	051564			JSR	R0,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
7531	035764	001324				CINITQ		: INITIATION QUEUE
7532	035766	000207			50\$:	RTS	PC	:RETURN

```

7533 .SBTTL CALCULATE CYLINDER, TRACK, AND SECTOR FOR RETRY
7534
7535 035770 010146 CALSEC: MOV R1,-(SP) ;STORE R1 ON STACK
7536 035772 010346 MOV R3,-(SP) ;STORE R3 ON STACK
7537 035774 013703 001634 MOV RTYBA,R3 ;GET ADDRESS OF ERROR WORD
7538 036000 166503 000010 SUB P.BALO(R5),R3 ;CALCULATE BYTES CORRECTLY TRANSFERRED
7539 036004 042703 000777 BIC #777,R3 ;MAKE IT ON SECTOR BOUNDARY
7540 036010 010337 001634 MOV R3,RTYBA ;STORE CALCULATED BUS ADDRESS
7541 036014 066537 000010 001634 ADD P.BALO(R5),RTYBA ;CALCULATE START OF SECTOR WITH ERROR
7542 036022 016501 000012 MOV P.WC(R5),R1 ;GET INITIAL WORD COUNT
7543 036026 005401 NEG R1 ;MAKE IT POSITIVE
7544 036030 006203 ASR R3 ;DETERMINE WORDS BEFORE ERROR
7545 036032 160301 SUB R3,R1 ;CALCULATE NUMBER OF WORDS LEFT
7546 036034 022701 000400 CMP #256.,R1 ;CHECK IF LESS THAN OR EQUAL TO ONE SECTOR
7547 036040 103002 BHIS 5$ ;LOAD RETRY WORD COUNT
7548 036042 012701 000400 MOV #256.,R1 ;NO, MAKE IT ONE SECTOR
7549 036046 010137 001636 5$: MOV R1,RTYWC ;LOAD RETY WORD COUNT
7550 036052 116537 000005 001641 MOV P.TRCK(R5),RTYTRK ;STORE RETRY TRACK
7551 036060 016537 000002 001642 MOV P.CYLN(R5),RTYCYL ;STORE RETRY CYLINDER
7552 036066 010346 MOV R3,-(SP) ;STORE WORDS TRANSFERRED
7553 036070 116616 000001 MOV 1(SP),(SP) ;STORE NUMBER OF SECTORS
7554 036074 105066 000001 CLRB 1(SP) ;CLEAR HIGH ORDER BYTE
7555 036100 005046 CLR -(SP) ;MAKE ROOM ON STACK
7556 036102 116516 000004 MOV P.SECT(R5),(SP) ;GET INITIAL SECTOR
7557 036106 062616 ADD (SP)+,(SP) ;ADD SECTORS TRANSFERRED
7558 036110 022716 000025 10$: CMP #21.,(SP) ;CHECK IF SECTOR COUNT GREATER THAN 21
7559 036114 103005 BHIS 15$ ;NO, CHECK TRACK
7560 036116 162716 000026 SUB #22.,(SP) ;SUBTRACT 22 SECTORS/TRACK
7561 036122 105237 001641 INCB RTYTRK ;INCREMENT TRACK
7562 036126 000770 BR 10$ ;CHECK SECTOR COUNT
7563
7564 036130 112637 001640 15$: MOV (SP)+,RTYSEC ;STORE RETRY SECTOR
7565 036134 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
7566 036136 113716 001641 MOVE RTYTRK,(SP) ;STORE RETRY TRACK
7567 036142 122716 000002 17$: CMPE #2,(SP) ;CHECK TRACK GREATER THAN 2
7568 036146 103005 BHIS 20$ ;NO, RETURN
7569 036150 162716 000003 SUB #3,(SP) ;SUBTRACT 3 TRACKS/CYLINDER
7570 036154 005237 001642 INC RTYCYL ;INCREMENT CYLINDER
7571 036160 000770 BR 17$ ;CHECK TRACK COUNT
7572
7573 036162 112637 001641 20$: MOV (SP)+,RTYTRK ;LOAD RETRY COUNT
7574 036166 112737 000001 001644 MOV #1,RTYSCN ;INDICATE ONE SECTOR IN RETRY COUNT
7575 036174 012603 MOV (SP)+,R3 ;RESTORE R3
7576 036176 012601 MOV (SP)+,R1 ;RESTORE R1
7577 036200 000207 RTS PC ;RETURN

```

```
7578  
7579  
7580 036202 010046  
7581 036204 010146  
7582 036206 010346  
7583 036210 010446  
7584 036212 104401 061571  
7585 036216 016503 000214  
7586 036222 005046  
7587 036224 116516 000217  
7588 036230 006316  
7589 036232 011601  
7590 036234 062603  
7591 036236 010346  
7592 036240 004737 053224  
7593 036244 122765 000001 000217  
7594 036252 103403  
7595 036254 104401 061113  
7596 036260 000436  
7597  
7598 036262 016504 000214  
7599 036266 016400 000002  
7600 036272 042700 177400  
7601 036276 005200  
7602  
7603 036300 126500 000217  
7604 036304 101402  
7605  
7606 036306 005046  
7607 036310 000414  
7608  
7609 036312 011404  
7610 036314 042704 177760  
7611 036320 006304  
7612 036322 016404 004602  
7613 036326 162701 000004  
7614 036332 042701 177700  
7615 036336 060104  
7616 036340 011446  
7617 036342 104401 061126  
7618 036346 004737 053224  
7619 036352 104401 061126  
7620 036356 011346  
7621 036360 004737 053224  
7622 036364 104401 061126  
7623 036370 005046  
7624 036372 116516 000217  
7625 036376 004737 053224  
7626 036402 105765 000217  
7627 036406 001003  
7628 036410 104401 061113  
7629 036414 000410  
7630  
7631 036416 104401 061126  
7632 036422 017546 000214  
7633 036426 042716 177760
```

.SBTTL PRINT FIRST ERROR WORD IN BUFFER

PRERWD: MOV R0,-(SP) ;STORE R0 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV R3,-(SP) ;STORE R3 ON STACK
MOV R4,-(SP) ;STORE R4 ON STACK
TYPE ,ERR012 ;TYPE ERROR HEADER
MOV P.ERHD(R5),R3 ;GET ADDRESS OF START OF SECTOR
CLR -(SP) ;MAKE ROOM ON STACK
MOVB P.ERAD(R5),(SP) ;GET WORD IN ERROR OF SECTOR
ASL (SP) ;MULTIPLY IT BY 2
MOV (SP),R1 ;STORE IT IN R1
ADD (SP)+,R3 ;GET ADDRESS OF BAD WORD
MOV R3,-(SP) ;STORE ADDRESS FOR PRINT OUT
JSR PC,BINOCT ;CONVERT TO OCTAL
CMPB #1,P.ERAD(R5) ;CHECK IF IN DESCRIPTOR AREA
BLO 5\$;NO, CALCULATE WORD
TYPE ,QUESMK ;TYPE QUESTION MARKS
BR 10\$;GO TYPE WORD READ

5\$: MOV P.ERHD(R5),R4 ;GET HEADER FOR PATTERN INDEX
MOV 2(R4),R0 ;GET NUMBER OF WORDS WRITTEN IN SECTOR
BIC #177400,R0 ;KEEP LOW BYTE
INC R0 ;INCREMENT R0 TO MAKE OFFSET FROM START
; OF SECTOR
CMPB P.ERAD(R5),R0 ;CHECK IF IN ZERO FILL AREA
BLOS 7\$;NO, CALCULATE ADDRESS OF
; WORD IN DATA PATTERN
CLR -(SP) ;LOAD ZERO FOR PRINT OUT
BR 9\$;GO PRINT EXPECTED CONTENTS

7\$: MOV (R4),R4 ;GET DATA PATTERN NUMBER FOR INDEX
BIC #177760,R4 ;MASK OFF INSIGNIFICANT BITS
ASL R4 ;MULTIPLY BY 2
MOV PATTAB(R4),R4 ;GET PATTERN BASE
SUB #4,R1 ;SUBTRACT 4 FROM SECTOR OFFSET
BIC #177700,R1 ;MAKE IT MOD 32 WORDS
ADD R1,R4 ;CALCULATE PATTERN ADDRESS
MOV (R4),-(SP) ;STORE EXPECTED CONTENTS FOR PRINT OUT

9\$: TYPE ,BLANKS ;TYPE BLANKS
JSR PC,BINOCT ;CONVERT TO OCTAL
TYPE ,BLANKS ;TYPE BLANKS

10\$: MOV (R3),-(SP) ;STORE WORD READ
JSR PC,BINOCT ;CONVERT TO OCTAL
TYPE ,BLANKS ;TYPE BLANKS
CLR -(SP) ;MAKE ROOM ON STACK
MOVB P.ERAD(R5),(SP) ;STORE SECTOR WORD NUMBER FOR PRINT OUT
JSR PC,BINOCT ;CONVERT TO OCTAL
TSTB P.ERAD(R5) ;CHECK IF ERROR IS IN PATTERN WORD
BNE 15\$;NO, PRINT PATTERN NUMBER
TYPE ,QUESMK ;TYPE QUESTION MARKS
BR 20\$;GO RESTORE REGISTERS

15\$: TYPE ,BLANKS ;TYPE BLANKS
MOV @P.ERHD(R5),-(SP) ;GET FIRST WORD OF SECTOR READ
BIC #177760,(SP) ;CLEAR INSIGNIFICANT BITS

7634	036432	004737	053224
7635	036436	104401	001165
7636	036442	012604	
7637	036444	012603	
7638	036446	012601	
7639	036450	012600	
7640	036452	000207	

20\$:	JSR	PC,BIN OCT	:CONVERT TO OCTAL
	TYPE	,SCLF	:TYPE <CR><LF>
	MOV	(SP)+,R4	:RESTORE R4
	MOV	(SP)+,R3	:RESTORE R3
	MOV	(SP)+,R1	:RESTORE R1
	MOV	(SP)+,R0	:RESTORE R0
	RTS	PC	:RETURN

```

7641          .SBTTL  SUCCESSFUL RECOVERY CLEAN UP
7642
7643 036454 032765 000001 000150 SURECY: BIT      #BIT0,P.DSTT(R5) ;CHECK IF DATA TYPE ERROR
7644 036462 001466          BEQ      10$          ;NO, CLEAN UP FOR NEXT COMMAND
7645 036464 105765 000147          TSTB   P.RERD(R5)   ;CHECK IF FIRST ERROR
7646 036470 001003          BNE     5$          ;NO, CHECK IF OFFSET
7647 036472 005265 000176          INC    P.SECC(R5) ;INCREMENT NO. OF ECC RECOVERABLE ERRORS
7648 036476 000426          BR     9$          ;GO CLEAN UP FOR NEXT COMMAND
7649
7650 036500 122765 000020 000147 5$:  CMPB  #20,P.RERD(R5) ;CHECK IF OFFSET
7651 036506 103403          BLO   8$          ;YES, INCREMENT OFFSET RECOVERABLE
7652 036510 005265 000172          INC    P.SRRD(R5) ;INCREMENT REREAD RECOVERABLE
7653 036514 000417          BR     9$          ;GO CLEAN UP FOR NEXT COMMAND
7654
7655 036516 005265 000174          INC    P.SOFF(R5) ;INCREMENT OFFSET RECOVERABLE
7656 036522 032777 020000 142410 8$:  BIT   #SW13,@SWR   ;CHECK IF INHIBIT PRINT OUT
7657 036530 001011          BNE   9$          ;YES, CLEAN UP FOR NEXT COMMAND
7658 036532 104401 064150          TYPE ,ERR108      ;TYPE "OFFSET"
7659 036536 005046          CLR  -(SP)       ;MAKE ROOM ON STACK
7660 036540 116516 000006          MOVB  P.OFST(R5),(SP) ;STORE OFFSET FOR PRINT OUT
7661 036544 004737 053224          JSR  PC,BIN OCT  ;CONVERT TO OCTAL
7662 036550 104401 001165          TYPE ,$CRLF      ;TYPE <CR><LF>
7663 036554 105037 001644          9$:  CLRB  RTYSCN     ;CLEAR SECTOR RETRY FLAG
7664 036560 016546 000176          MOV   P.SECC(R5),-(SP) ;STORE NUMBER OF ECC RECOVERABLE ERRORS
7665 036564 066516 000172          ADD   P.SRRD(R5),(SP) ;ADD NUMBER OF REREAD RECOVERABLE ERRORS
7666 036570 066516 000174          ADD   P.SOFF(R5),(SP) ;ADD NUMBER OF OFFSET RECOVERABLE ERRORS
7667 036574 022665 000100          CMP   (SP)+,P.CERT(R5) ;CHECK IF SOFT ERROR RATE EXCEEDED
7668 036600 101417          BLOS 10$         ;NO, GO ISSUE NEXT COMMAND
7669 036602 032777 040020 142330 BIT   #SW4!SW14,@SWR ;CHECK IF INHIBIT AUTOMATIC DRIVE DROPPING
7670 036610 001013          BNE  10$         ;YES, GO ISSUE NEXT COMMAND
7671 036612 004737 023020          JSR  PC,BUFREL   ;GO RELEASE BUFFER
7672 036616 104401 064445          TYPE ,ERR205     ;TYPE "ERROR THRESHOLD EXCEEDED"
7673 036622 004737 013162          JSR  PC,DROP     ;GO DROP DRIVE
7674 036626 005037 001566          CLR  ERRPRO     ;CLEAR ERROR CURRENTLY BEING PROCESSED
7675 036632 004737 022520          JSR  PC,GETBUF  ;GET BUFFER FOR NEXT COMMAND
7676 036636 000207          RTS            ;RETURN
7677
7678 036640 032777 020000 142272 10$: BIT   #SW13,@SWR   ;CHECK IF INHIBIT PRINT OUT
7679 036646 001002          BNE  15$         ;YES, CHECK IF READ PACK SERIAL NUMBER
7680 036650 104401 064355          TYPE ,ERR203     ;TYPE RECOVERABLE ERROR
7681 036654 105037 001644          15$: CLRB  RTYSCN     ;CLEAR SECTOR RETRY COUNT
7682 036660 005065 000146          CLR  P.RECT(R5) ;CLEAR ERROR FLAGS AND RETRY COUNTS
7683 036664 005065 000150          CLR  P.DSTT(R5)
7684 036670 005065 000212          CLR  P.ERR(R5)
7685 036674 005037 001566          CLR  ERRPRO     ;CLEAR ERROR CURRENTLY BEING PROCESSED
7686 036700 010446          MOV   R4,-(SP)  ;STORE R4 ON STACK
7687 036702 116504 000000          MOVB  P.DRVN(R5),R4 ;STORE DRIVE NUMBER FOR INDEX
7688 036706 156437 001514 001513 BISB  INTMSK(R4),O.OVER ;ALLOW OVERLAPPED OPERATIONS ON THIS DRIVE
7689 036714 012604          MOV   (SP)+,R4 ;RESTORE R4
7690 036716 105765 000210          TSTB P.ASSN(R5) ;CHECK IF IN ASSIGNMENT SEQUENCE
7691 036722 001025          BNE  20$         ;YES, TERMINATE SEQUENCE
7692 036724 122765 000117 000001 CMPB  #SEEK,P.CMND(R5) ;CHECK IF SEEK TO PREVIOUS CYLINDER
7693 036732 001410          BEQ  16$         ;YES, GO ISSUE RELEASE
7694 036734 122765 000140 000001 CMPB  #RELEAS,P.CMND(R5) ;CHECK IF RELEASE
7695 036742 001013          BNE  17$         ;NO, CARRY OUT NORMAL RECOVERY
7696 036744 004037 051564          JSR  R0,Q.PUSH  ;ENQUEUE PARAMETER BLOCK IN DRIVE

```

```
7697 036750 001314          AVAILQ          : AVAILIABLE QUEUE
7698 036752 000207          RTS            PC          :RETURN
7699
7700 036754 112765 000140 000001 16$: MOVB #RELEAS,P.CMND(R5) :ISSUE RELEASE COMMAND
7701 036762 004037 051564          JSR            R0,Q.PUSH  :ENQUEUE PARAMETER BLOCK IN COMMAND
7702 036766 001324          CINITQ        : INITIATION QUEUE
7703 036770 000207          RTS            PC          :RETURN
7704
7705 036772 000137 024774          JMP            NORM1      :CARRY OUT NORMAL DATA TRANSFER TERMINATION
7706
7707          : CHECK IF ANY OF THE FOLLOWING ASSIGNMENT STAGES
7708          : READ DRIVE STATUS
7709          : START SPINDLE
7710          : SET VOLUME VALID
7711          : RECAL
7712          : READ PACK SERIAL NUMBER
7713 036776 132765 000117 000210 20$: BITB #BIT0!BIT1!BIT2!BIT3!BIT6,P.ASSN(R5)
7714 037004 001402          BEQ            25$       :NO, PROCESS WRITE PACK
7715 037006 000137 013662          JMP            ASNORM     :PROCESS NORMAL TERMINATION
7716
7717 037012 032765 002000 000014 25$: BIT #DRPDRV,P.PRST(R5) :CHECK IF DROP DRIVE
7718 037020 001030          BNE            30$       :YES, CO DROP DRIVE
7719 037022 105065 000211          CLRB          P.SEEK(R5) :CLEAR SEEK TO PREVIOUS LOCATION
7720 037026 116565 000123 000001  MOVB          P.RCMD(R5),P.CMND(R5) :REISSUE LAST WRITE/WRITE CHECK COMMAND
7721 037034 052765 000200 000014  BIS          #W.WCK,P.PRST(R5) :SET ISSUE WRITE BEFORE WRITE CHECK
7722 037042 016565 000126 000004  MOV          P.RSEC(R5),P.SECT(R5) :LOAD PREVIOUS TRACK AND SECTOR
7723 037050 016565 000124 000002  MOV          P.RCYL(R5),P.CYLN(R5) :LOAD PREVIOUS CYLINDER
7724 037056 016565 000132 000010  MOV          P.RBAL(R5),P.BALO(R5) :LOAD PREVIOUS BUS ADDRESS
7725 037064 016565 000130 000012  MOV          P.RWC(R5),P.WC(R5) :LOAD PREVIOUS WORD COUNT
7726 037072 004037 051564          JSR            R0,Q.PUSH  :ENQUEUE PARAMETER BLOCK IN COMMAND
7727 037076 001324          CINITQ        : INITIATION QUEUE
7728 037100 000207          RTS            PC          :RETURN
7729
7730 037102 004737 023020          JSR            PC,BUFREL  :GO RELEASE BUFFER
7731 037106 104401 056656          TYPE          ,OPR008    :TYPE 'OPERATOR INITIATED DROP DRIVE
7732          : DURING PACK WRITING'
7733 037112 004737 013162          JSR            PC,DROP    :GO DROP DRIVE
7734 037116 004737 022520          JSR            PC,GETBUF  :GET BUFFER FOR NEXT COMMAND
7735 037122 000207          RTS            PC          :RETURN
```

```
7736  
7737  
7738 037124 010446  
7739 037126 010346  
7740 037130 010504  
7741 037132 012703 004516  
7742  
7743 037136 012423  
7744 037140 022703 004602  
7745 037144 001374  
7746 037146 012603  
7747 037150 012604  
7748 037152 000207  
7749  
7750  
7751  
7752 037154 010446  
7753 037156 010346  
7754 037160 010504  
7755 037162 012703 004516  
7756  
7757 037166 012324  
7758 037170 022703 004602  
7759 037174 001374  
7760 037176 012603  
7761 037200 012604  
7762 037202 000207
```

.SBTTL GET CURRENT STATUS INFORMATION

```
GETCUR: MOV R4,-(SP) :STORE R4 ON STACK  
MOV R3,-(SP) :STORE R3 ON STACK  
MOV R5,R4 :LOAD START OF CURRENT PARRAMETER BLOCK  
MOV #PARM10,R3 :LOAD START OF RETRY PARAMETER BLOCK  
  
5$: MOV (R4)+,(R3)+ :COPY PARAMTER BLOCK  
CMP #PARM10+P.EPAT+2,R3 :CHECK IF FINISHED  
BNE 5$ :NO, GET NEXT WORD  
MOV (SP)+,R3 :RESTORE R3  
MOV (SP)+,R4 :RESTORE R4  
RTS PC :RETURN
```

.SBTTL RESTORE STATUS

```
RSTAT: MOV R4,-(SP) :STORE R4 ON STACK  
MOV R3,-(SP) :STORE R3 ON STACK  
MOV R5,R4 :LOAD START OF CURRENT PARRAMETER BLOCK  
MOV #PARM10,R3 :LOAD START OF RETRY PARAMETER BLOCK  
  
5$: MOV (R3)+,(R4)+ :COPY PARAMTER BLOCK  
CMP #PARM10+P.EPAT+2,R3 :CHECK IF FINISHED  
BNE 5$ :NO, GET NEXT WORD  
MOV (SP)+,R3 :RESTORE R3  
MOV (SP)+,R4 :RESTORE R4  
RTS PC :RETURN
```

```
.SBTTL DETERMINE OFFSET FOR RETRY
7763
7764
7765 037204 122765 000021 000147 DETOFF: CMPB #21,P.RERD(R5) :CHECK IF OFFSET = +400 MICRO-INCHES
7766 037212 001004 BNE 10$ :NO, CONTINUE
7767 037214 112765 000020 000006 MOVB #OFFP4,P.OFST(R5) :LOAD OFFSET VALUE = +400 MICRO-INCHES
7768 037222 000447 BR 15$ :ISSUE OFFSET
7769
7770 037224 122765 000024 000147 10$: CMPB #24,P.RERD(R5) :CHECK IF OFFSET = -400 MICRO-INCHES
7771 037232 001004 BNE 11$ :NO, CONTINUE
7772 037234 112765 000220 000006 MOVB #OFFM4,P.OFST(R5) :LOAD OFFSET VALUE = -400 MICRO-INCHES
7773 037242 000437 BR 15$ :ISSUE OFFSET
7774
7775 037244 122765 000027 000147 11$: CMPB #27,P.RERD(R5) :CHECK IF OFFSET = +800 MICRO-INCHES
7776 037252 001004 BNE 12$ :NO, CONTINUE
7777 037254 112765 000040 000006 MOVB #OFFP8,P.OFST(R5) :LOAD OFFSET VALUE = +800 MICRO-INCHES
7778 037262 000427 BR 15$ :ISSUE OFFSET
7779
7780 037264 122765 000032 000147 12$: CMPB #32,P.RERD(R5) :CHECK IF OFFSET = -800 MICRO-INCHES
7781 037272 001004 BNE 13$ :NO, CONTINUE
7782 037274 112765 000240 000006 MOVB #OFFM8,P.OFST(R5) :LOAD OFFSET VALUE = -800 MICRO-INCHES
7783 037302 000417 BR 15$ :ISSUE OFFSET
7784
7785 037304 122765 000035 000147 13$: CMPB #35,P.RERD(R5) :CHECK IF OFFSET = +1200 MICRO-INCHES
7786 037312 001004 BNE 14$ :NO, CONTINUE
7787 037314 112765 000060 000006 MOVB #OFFP12,P.OFST(R5) :LOAD OFFSET VALUE = +1200 MICRO-INCHES
7788 037322 000407 BR 15$ :ISSUE OFFSET
7789
7790 037324 122765 000040 000147 14$: CMPB #40,P.RERD(R5) :CHECK IF OFFSET = -1200 MICRO-INCHES
7791 037332 001013 BNE 30$ :NO, DO REREAD
7792 037334 112765 000260 000006 MOVB #OFFM12,P.OFST(R5) :LOAD OFFSET VALUE = -1200 MICRO-INCHES
7793 037342 112765 000115 000001 15$: MOVB #OFFSET,P.CMND(R5) :LOAD OFFSET COMMAND
7794 037350 004037 051564 JSR R0,Q.PUSH :ENQUEUE PARAMETER BLOCK IN COMMAND
7795 037354 001324 CINITQ : INITIATION QUEUE
7796 037356 011000 MOV (R0),R0 :LOAD OFFSET ISSUED RETURN
7797 037360 000200 RTS R0 :RETURN
7798
7799 037362 005720 30$: TST (R0)+ :ADJUST RETURN FOR NO OFFSET
7800 037364 000200 RTS R0 :RETURN
```



```
7801 .SBTTL CALCULATE RETRY WORD COUNT
7802
7803 037366 013746 001642 CALWC: MOV RTYCYL,-(SP) :STORE RETRY CYLINDER
7804 037372 166516 000002 SUB P.CYLN(R5),(SP) :SUBTRACT INITIAL CYLINDER
7805 037376 011646 MOV (SP),-(SP) :STORE CYLINDER DIFFERENCE
7806 037400 006316 ASL (SP) :MULTIPLY DIFFERENCE BY 2
7807 037402 062616 ADD (SP)+,(SP) :MULTIPLY DIFFERENCE BY 3
7808 037404 005046 CLR -(SP) :MAKE ROOM ON STACK
7809 037406 113716 001641 MOVB RTYTRK,(SP) :GET RETRY TRACK
7810 037412 062616 ADD (SP)+,(SP) :ADD RETRY TRACK
7811 037414 005046 CLR -(SP) :MAKE ROOM ON THE STACK
7812 037416 116516 000005 MOVB P.TRCK(R5),(SP) :GET INITIAL TRACK
7813 037422 162616 SUB (SP)+,(SP) :SUBTRACT INITIAL TRACK
7814 037424 012746 000026 MOV #22,-(SP) :MULTIPLY TRACK BY 22
7815 037430 004737 055210 JSR PC,$MULT
7816 037434 012616 MOV (SP)+,(SP) :THROW AWAY MOST SIGNIFICANT BITS
7817 037436 005046 CLR -(SP) :MAKE ROOM ON THE STACK
7818 037440 113716 001640 MOVB RTYSEC,(SP) :GET RETRY SECTOR
7819 037444 062616 ADD (SP)+,(SP) :ADD PRESENT SECTOR
7820 037446 005046 CLR -(SP) :MAKE ROOM ON STACK
7821 037450 116516 000004 MOVB P.SECT(R5),(SP) :GET INITIAL SECTOR
7822 037454 162616 SUB (SP)+,(SP) :CALCULATE NUMBER OF SECTORS UNTIL ERROR
7823 037456 111666 000001 MOVB (SP),1(SP) :MULTIPLY BY 256 WORDS/SECTOR
7824 037462 105016 CLRB (SP)
7825 037464 161637 001636 SUB (SP),RTYWC :SUBTRACT FROM INITIAL WORD COUNT
7826 037470 006316 ASL (SP) :DETERMINE NUMBER OF BYTES
7827 037472 062637 001634 ADD (SP)+,RTYBA :CALCULATE STARTING BUFFER ADDRESS
7828 037476 000207 RTS PC :RETURN
```

.SBTTL RECOVERABLE CONTROLLER ERROR TERMINATION

7829									
7830									
7831	037500	012037	037522		CONERR:	MOV	(R0)+,2\$:STORE ERROR MESSAGE ADDRESS
7832	037504	004737	030156			JSR	PC,PRI000		:PRINTER HEADER
7833	037510	032777	020000	141422		BIT	#SW13,@SWR		:CHECK IF INHIBIT PRINT OUT
7834	037516	001032				BNE	10\$:YES, CHECK IF HALT ON ERROR
7835	037520	104401				TYPE			:TYPE ERROR MESSAGE
7836	037522	000000			2\$:	.WORD	0		
7837	037524	104401	001165			TYPE	,SCLF		:TYPE <CR><LF>
7838	037530	004737	026252			JSR	PC,PRISTT		:PRINT CONTROLLER REGISTERS
7839	037534	004737	030422			JSR	PC,PRI002		:PRINT CURRENT EXECUTED COMMAND
7840	037540	022737	063526	037522		CMP	#ERR066,2\$:CHECK IF ECC LOGIC ERROR
7841	037546	001016				BNE	10\$:NO, CHECK IF HALT ON ERROR
7842	037550	104401	063546			TYPE	,ERR067		:TYPE 'RKECPT RKECPS'
7843	037554	016546	000062			MOV	P.EPAT(R5),-(SP)		:STORE ECC PATTERN FOR PRINT OUT
7844	037560	004737	053224			JSR	PC,BINOCT		:CONVERT TO OCTAL
7845	037564	104401	061126			TYPE	,BLANKS		:TYPE 2 SPACES
7846	037570	016546	000060			MOV	P.EPOS(R5),-(SP)		:STORE ECC POSITION FOR PRINT OUT
7847	037574	004737	053224			JSR	PC,BINOCT		:CONVERT TO OCTAL
7848	037600	104401	001165			TYPE	,SCLF		:TYPE <CR><LF>
7849	037604	004737	031426		10\$:	JSR	PC,HLT000		:CHECK IF HALT ON ERROR
7850	037610	005237	001630			INC	CNTCNT		:INCREMENT CONTROLLER ERROR COUNT
7851	037614	022737	000012	001630		CMP	#T.CONT,CNTCNT		:CHECK IF THRESHOLD EXCEEDED
7852	037622	103006				BHIS	15\$:NO, CONTINUE
7853	037624	104401	061131			TYPE	,ERR000		:TYPE 'FATAL ERROR'
7854	037630	104401	061470			TYPE	,ERR008		:TYPE 'CONTROLLER ERROR THRESHOLD EXCEEDED'
7855	037634	000000				HALT			
7856	037636	000776				BR	.-2		
7857									
7858	037640	105765	000150		15\$:	TSTB	P.DSTT(R5)		:CHECK IF FIRST ERROR
7859	037644	100005				BPL	20\$:NO, REQUEUE COMMAND
7860	037646	042765	100200	000150		BIC	#BIT15!BIT7,P.DSTT(R5)		:CLEAR ERROR ENQUEUED AND FIRST ERROR
7861	037654	005037	001566			CLR	ERRPRO		:CLEAR ERROR BEING PROCESSED
7862	037660	004037	051564		20\$:	JSR	R0,Q.PUSH		:REQUEUE COMMAND IN COMMAND
7863	037664	001324				CINITQ			: INITIATION QUEUE
7864	037666	000200				RTS	R0		:RETURN

```

7865 .SBTTL TEST FOR DROP DRIVE ERRORS
7866
7867 037670 032765 000030 000036 TSTDRP: BIT #ACLO!SPDLSS,P.DS(R5) :CHECK IF AC LOW OR SPEED LOSS
7868 037676 001016 BNE 10$ :YES REPORT ERROR
7869 : CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET
7870 : UNSAFE
7871 : ILLEGAL DISK ADDRESS
7872 : WRITE LOCK ERROR
7873 : DRIVE TYPE ERROR
7874 : FORMAT ERROR
7875 : NON-EXECUTABLE DRIVE FUNCTION
7876 037700 032765 046064 000034 BIT #UNS!IDAE!WLE!DTYE!FMTE!NXF,P.ER(R5)
7877 BNE 12$ :YES, REPORT ERROR
7878 037706 001026 CHECK IF ANY OF THE FOLLOWING FLAGS ARE SET
7879 : DRIVE ERROR NOT REPORTED
7880 : DRIVE READY NOT SET AFTER START SPINDLE
7881 : VOLUME VALID NOT SET AFTER PACK ACKNOWLEDGE
7882 BIT #BIT3!BIT10!BIT11,P.ERR(R5)
7883 037710 032765 006010 000212 BNE 18$ :YES, GO REPORT ERROR
7884 037716 001072 CHECK IF ANY OF THE FOLLOWING FLAGS ARE SET
7885 : DRIVE HARD ERROR
7886 : DRIVE STATUS CHANGE DID NOT CLEAR
7887 : UNEXPECTED ATTN FROM DRIVE COMMAND
7888 : ATTENTION SET BUT DSC AND FAULT RESET
7889 BIT #DRVHRD!DRVDSC!UEXATT!NODSC,P.PRST(R5)
7890 037720 032765 004070 000014 BNE 32$ :YES REPORT ERROR
7891 037726 001116 TST (R0)+ :GET NORMAL RETURN ADDRESS
7892 037730 005720 RTS R0 :RETURN
7893 037732 000200
7894 037734 004737 030156 10$: JSR PC,PRI000 :PRINT ERROR HEADER
7895 037740 032765 000010 000036 BIT #ACLO,P.DS(R5) :CHECK IF AC LOW
7896 037746 001403 BEQ 11$ :NO, CONTINUE
7897 037750 104401 062501 TYPE ,ERR033 :TYPE "AC LOW"
7898 037754 000472 BR 30$ :GO GET ERROR QUALIFICATION
7899 037756 104401 062510 11$: TYPE ,ERR034 :TYPE "SPEED LOSS"
7900 037762 000467 BR 30$ :GET ERROR QUALIFICATION
7901 037764 004737 030156 12$: JSR PC,PRI000 :PRINT ERROR HEADER
7902 037770 032765 040000 000034 BIT #UNS,P.ER(R5) :CHECK IF DRIVE UNSAFE
7903 037776 001403 BEQ 13$ :NO, CONTINUE
7904 040000 104401 063220 TYPE ,ERR056 :TYPE "DRIVE UNSAFE"
7905 040004 000456 BR 30$ :GET ERROR QUALIFICATION
7906 040006 032765 000040 000034 13$: BIT #DTYE,P.ER(R5) :CHECK IF DRIVE TYPE ERROR
7907 040014 001403 BEQ 14$ :NO, CONTINUE
7908 040016 104401 062523 TYPE ,ERR035 :TYPE "DRIVE TYPE ERROR"
7909 040022 000447 BR 30$ :GET ERROR QUALIFIER BITS
7910 040024 032765 000020 000034 14$: BIT #FMTE,P.ER(R5) :CHECK IF FORMAT ERROR
7911 040032 001403 BEQ 15$ :NO, CONTINUE
7912 040034 104401 062540 TYPE ,ERR036 :TYPE FORMAT ERROR

```

7921	040040	000440				BR	30\$:GET ERROR QUALIFIER BITS
7922									
7923	040042	032765	000004	000034	15\$:	BIT	#NXF,P.ER(R5)		:CHECK IF NON-EXECUTABLE DRIVE FUNCTION
7924	040050	001403				BEQ	16\$:NO, CONTINUE
7925	040052	104401	062553			TYPE	ERR037		:TYPE ILLEGAL DRIVE FUNCTION
7926	040056	000431				BR	30\$:GET ERROR QUALIFIER BITS
7927									
7928	040060	032765	002000	000034	16\$:	BIT	#IDAE,P.ER(R5)		:CHECK IF ILLEGAL DISK ADDRESS
7929	040066	001403				BEQ	17\$:NO, CONTINUE
7930	040070	104401	062450			TYPE	ERR031		:TYPE "ILLEGAL DISK ADDRESS"
7931	040074	000422				BR	30\$:GET ERROR QUALIFIER BITS
7932									
7933	040076	104401	062464		17\$:	TYPE	ERR032		:TYPE "WRITE LOCK ERROR"
7934	040102	000417				BR	30\$:GET ERROR QUALIFIER BITS
7935									
7936	040104	004737	030156		18\$:	JSR	PC,PRI000		:PRINT ERROR HEADER
7937	040110	032765	002000	000212		BIT	#BIT10,P.ERR(R5)		:CHECK IF DRIVE NOT READY AFTER START SPINDLE
7938	040116	001403				BEQ	19\$:NO, CONTINUE
7939	040120	104401	063112			TYPE	ERR053+1		:PRINT "DRIVE NOT READY AFTER START SPINDLE"
7940	040124	000406				BR	30\$:GET ERROR QUALIFIER BITS
7941									
7942	040126	032765	004000	000212	19\$:	BIT	#BIT11,P.ERR(R5)		:CHECK IF VOLUME VALID DID NOT SET
7943									: AFTER PACK ACKNOWLEDGE
7944	040134	001404				BEQ	31\$:NO, CONTINUE
7945	040136	104401	063153			TYPE	ERR054+1		:PRINT "VOLUME VAID NOT SET AFTER
7946									: PACK ACKNOWLEDGE INDSTRUCTION"
7947	040142	104401	001165		30\$:	TYPE	\$CRLF		:TYPE <CR><LF>
7948	040146	032765	000010	000212	31\$:	BIT	#BIT3,P.ERR(R5)		:CHECK IF ERROR NOT INDICATED BY FAULT
7949	040154	001405				BEQ	42\$:NO, CHECK OTHER QUALIFIER BITS
7950	040156	104401	063233			TYPE	ERR057		:TYPE "ERROR NOT INDICATED BY FAULT"
7951	040162	000402				BR	42\$:CHECK OTHER QUALIFIER BITS
7952									
7953	040164	004737	030156		32\$:	JSR	PC,PRI000		:PRINT ERROR HEADER
7954	040170	032765	000020	000014	42\$:	BIT	#DRVHRD,P.PRST(R5)		:CHECK IF HARD DRIVE ERROR
7955	040176	001402				BEQ	43\$:NO, CHECK OTHER QUALIFIER BITS
7956	040200	104401	062321			TYPE	ERR027		:TYPE "DRIVE HARD ERROR"
7957	040204	032765	004000	000014	43\$:	BIT	#NODSC,P.PRST(R5)		:CHECK IF ATTENTION BUT
7958									: NO DRIVE STATUS CHANGE OR FAULT
7959	040212	001402				BEQ	44\$:NO, CHECK OTHER QUALIFIER BITS
7960	040214	104401	063274			TYPE	ERR058		:TYPE "ATTN BUT NO DSC OR FAULT"
7961	040220	032765	000040	000014	44\$:	BIT	#DRVDSC,P.PRST(R5)		:CHECK IF DRIVE STATUS CHANGE DID
7962									: NOT CLEAR
7963	040226	001402				BEQ	45\$:NO, CHECK OTHER ERROR QUALIFIER BITS
7964	040230	104401	062340			TYPE	ERR028		:TYPE "DRIVE STATUS CHANGE DID NOT CLEAR"
7965	040234	032765	000010	000014	45\$:	BIT	#UEXATT,P.PRST(R5)		:CHECK IF UNEXPECTED ATTENTION FROM
7966									: DRIVE COMMAND
7967	040242	001402				BEQ	46\$:NO, CHECK OTHER QUALIFIER BITS
7968	040244	104401	062430			TYPE	ERR030		:TYPE "UNEXPECTED ATTN FROM DRIVE COMMAND"
7969	040250	005265	000206		46\$:	INC	P.NER(R5)		:INCREMENT OTHER ERROR COUNT
7970	040254	032777	020000	140656		BIT	#SW13,@SWR		:CHECK IF INHIBIT PRINT OUT
7971	040262	001015				BNE	60\$:YES, CHECK IF HALT ON ERROR
7972	040264	005765	000150			TST	P.DSTT(R5)		:CHECK IF ERROR DURING ERROR RECOVERY
7973	040270	100402				BMI	51\$:NO, PRINT STATUS
7974	040272	104401	064317			TYPE	ERR202		:TYPE "ERROR DURING ERROR RECOVERY"
7975	040276	004737	026252		51\$:	JSR	PC,PRSTT		:PRINT CONTROLLER REGISTERS
7976	040302	004737	026076			JSR	PC,PRDSTT		:PRINT DRIVE STATUS

7977	040306	004737	030422		JSR	PC,PRI002	:PRINT CURRENT COMMAND
7978	040312	004737	030244		JSR	PC,PRI001	:PRINT PREVIOUS COMMAND
7979	040316	004737	031426		JSR	PC,HLT000	:CHECK IF HALT ON ERROR
7980	040322	032777	000100	140610	BIT	#SW6,@SWR	:CHECK FOR SWITCH 6
7981	040330	001431			BEQ	63\$:NO DROP DRIVE
7982	040332	032765	000030	000036	BIT	#ACLO!SPDLSS,P.DS(R5)	:CHECK IF UNSAFE, ACLOW AND SPEED LOSS
7983	040340	001004			BNE	61\$	
7984	040342	032765	040000	000034	BIT	#UNS,P.ER(R5)	
7985	040350	001421			BEQ	63\$	
7986	040352	105765	000120		61\$: TSTB	P.IERC(R5)	:CHECK IF INHIBIT ERROR RECOVERY
7987	040356	001012			BNE	62\$	
7988	040360	105265	000146		INCB	P.RECT(R5)	:INCREMENT ERROR RETRY COUNT
7989	040364	122765	000010	000146	CIPB	#8.,P.RECT(R5)	:CHECK IF RETRY THRESHOLD EXCEEDED
7990	040372	103404			BLO	62\$:YES, INDICATE UNSUCCESSFUL RECOVERY
7991	040374	004037	051564		JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
7992	040400	001324			CINITQ		: INITIATION QUEUE
7993	040402	000402			BR	57\$:RETURN
7994							
7995	040404	004737	031660		62\$: JSR	PC,UNRECY	:INDICATE UNSUCCESSFUL RECOVERY
7996	040410	011000			57\$: MOV	(R0),R0	
7997	040412	000200			RTS	R0	
7998							
7999	040414	004737	023020		63\$: JSR	PC,BUFREL	:RELEASE BUFFER
8000	040420	004737	013162		JSR	PC,DROP	:DROP DRIVE
8001	040424	004737	022520		JSR	PC,GETBUF	:GET NEW BUFFER
8002	040430	005037	001566		CLR	ERRPRO	:CLEAR ERROR BEING PROCESSED
8003	040434	011000			MOV	(R0),R0	:LOAD ERROR RETURN
8004	040436	000200			RTS	R0	:RETURN

```

8005 .SBTTL ERROR RECOVERY SEQUENCE NORMAL RETURN
8006
8007 040440 032765 004000 000150 ERCVY1: BIT #BIT11,P.DSTT(R5) ;CHECK IF RECALIBRATE ISSUED
8008 040446 001415 BEQ EC3$ ;NO, CONTINUE
8009 040450 042765 004000 000150 BIC #BIT11,P.DSTT(R5) ;CLEAR RECALIBRATE ISSUED
8010 040456 052765 010000 000150 15$: BIS #BIT12,P.DSTT(R5) ;SET SEEK ISSUED
8011 040464 112765 000117 000001 MOVB #SEEK,P.CMND(R5) ;ISSUE SEEK INSTRUCTION
8012 040472 004037 051564 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
8013 040476 001324 CINITQ ; INITIATION QUEUE
8014 040500 000207 RTS PC ;RETURN
8015
8016 040502 032765 010000 000150 EC3$: BIT #BIT12,P.DSTT(R5) ;CHECK IF SEEK ISSUED
8017 040510 001427 BEQ 10$ ;NO, CONTINUE
8018 040512 042765 010000 000150 BIC #BIT12,P.DSTT(R5) ;CLEAR SEEK ISSUED
8019 040520 105765 000211 TSTB P.SEEK(R5) ;CHECK IF SEEK TO PREVIOUS POSITION
8020 040524 001403 BEQ 5$ ;NO, PROCESS DATA TRANSFER
8021 040526 004737 036454 JSR PC,SURECY ;INDICATE SUCCESSFUL RECOVERY
8022 040532 000207 RTS PC ;RETURN
8023
8024 040534 122765 000021 000147 5$: CMPB #21,P.RERD(R5) ;CHECK IF OFFSET
8025 040542 101053 BHI 20$ ;NO, REISSUE COMMAND
8026 040544 052765 020000 000150 BIS #BIT13,P.DSTT(R5) ;SET OFFSET ISSUED
8027 040552 112765 000115 000001 MOVB #OFFSET,P.CMND(R5) ;ISSUE OFFSET
8028 040560 004037 051564 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
8029 040564 001324 CINITQ ; INITIATION QUEUE
8030 040566 000207 RTS PC ;RETURN
8031
8032 040570 032765 020000 000150 10$: BIT #BIT13,P.DSTT(R5) ;CHECK IF DO PREVIOUS OFFSET
8033 040576 001453 BEQ EC4$ ;NO, CONTINUE
8034 040600 042765 020000 000150 BIC #BIT13,P.DSTT(R5) ;CLEAR OFFSET ISSUED
8035 040606 122765 000021 000147 CMPB #21,P.RERD(R5) ;CHECK IF OFFSET +400 MICRO-INCHES
8036 040614 001424 BEQ 15$ ;YES, GO ISSUE READ
8037 040616 122765 000024 000147 CMPB #24,P.RERD(R5) ;CHECK IF OFFSET -400 MICRO-INCHES
8038 040624 001420 BEQ 15$ ;YES, GO ISSUE READ
8039 040626 122765 000027 000147 CMPB #27,P.RERD(R5) ;CHECK IF OFFSET +800 MICRO-INCHES
8040 040634 001414 BEQ 15$ ;YES, GO ISSUE READ
8041 040636 122765 000032 000147 CMPB #32,P.RERD(R5) ;CHECK IF OFFSET -800 MICRO-INCHES
8042 040644 001410 BEQ 15$ ;YES, GO ISSUE READ
8043 040646 122765 000035 000147 CMPB #35,P.RERD(R5) ;CHECK IF OFFSET +1200 MICRO-INCHES
8044 040654 001404 BEQ 15$ ;YES, GO ISSUE READ
8045 040656 122765 000040 000147 CMPB #40,P.RERD(R5) ;CHECK IF OFFSET -1200 MICRO-INCHES
8046 040664 001002 BNE 20$ ;NO, DO NOT INCREMENT REREAD COUNT
8047 040666 105265 000147 15$: INCB P.RERD(R5) ;INCREMENT REREAD COUNT
8048 040672 116565 000123 000001 20$: MOVB P.RCMD(R5),P.CMND(R5) ;LOAD COMMAND
8049 040700 122765 000131 000001 CMPB #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK
8050 040706 001003 BNE 25$ ;NO, ISSUE COMMAND
8051 040710 052765 000200 000014 BIS #W.WCK,P.PRST(R5) ;YES, ISSUE WRITE BEFOR WRITE CHECK
8052 040716 004037 051564 25$: JSR R0,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN COMMAND
8053 040722 001324 CINITQ ; INITIATION QUEUE
8054 040724 000207 RTS PC ;RETURN
8055
8056 040726 032765 000100 000150 EC4$: BIT #BIT6,P.DSTT(R5) ;CHECK IF WAITING FOR READ STATUS
8057 BEQ EC5$ ;FOR POSITIONING TIME OUT
8058 040734 001436 BEQ EC5$ ;NO, CONTINUE
8059 040736 042765 000100 000150 BIC #BIT6,P.DSTT(R5) ;CLEAR WAITING FOR READ STATUS
8060 040744 113765 001666 000001 MOVB POSCMD,P.CMND(R5) ;RESTORE COMMAND OF ERROR

```

8061	040752	032765	000200	000036		BIT	#DRDY,P.DS(R5)	:CHECK IF DRIVE READY
8062	040760	001004				BNE	10\$:YES, REISSUE COMMAND
8063	040762	004037	030642			JSR	R0,PRI100	:PRINT DRIVE TIME OUT DURING POSITIONING
8064	040766	063033				ERR052		: AND DROP DRIVE
8065	040770	000207			5\$:	RTS	PC	:RETURN
8066								
8067	040772	004037	031032		10\$:	JSR	R0,PRI200	:PRINT DRIVE TIME OUT DURING
8068	040776	063033				ERR052		: POSITIONING
8069	041000	004037	031456			JSR	R0,NERSTT	:TAKE OTHER ERROR STATICS
8070	041004	040770				5\$:NO RECOVERY RETURN
8071	041006	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	:SET RECALIBRATE ISSUED
8072	041014	112765	000113	000001		MOVB	#RECAL,P.CMND(R5)	:ISSUE RECALIBRATE
8073	041022	004037	051564			JSR	R0,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
8074	041026	001324				CINITQ		: INITIATION QUEUE
8075	041030	000207				RTS	PC	:RETURN
8076								
8077	041032	032765	000020	000150	EC5\$:	BIT	#BIT4,P.DSTT(R5)	:CHECK IF HEADER ERROR
8078	041040	001002				BNE	1\$:YES, GO PROCESS ERROR
8079	041042	000137	041556			JMP	EC6\$:NO, CONTINUE
8080								
8081	041046	042765	000020	000150	1\$:	BIC	#BIT4,P.DSTT(R5)	:CLEAR READ ALL HEADERS ISSUED
8082	041054	010346				MOV	R3,-(SP)	:STORE R3 ON THE STACK
8083	041056	016546	000056			MOV	P.B11(R5),-(SP)	:STORE SECTOR READ
8084	041062	042716	177017			BIC	#*C<M.SECT>,(SP)	:KEEP ONLY SECTOR COUNT
8085	041066	006216				ASR	(SP)	:SHIFT 4 BITS RIGHT
8086	041070	006216				ASR	(SP)	
8087	041072	006216				ASR	(SP)	
8088	041074	006216				ASR	(SP)	:GET SECTOR COUNT
8089	041076	005046				CLR	-(SP)	:MAKE ROOM ON THE STACK
8090	041100	116516	000004			MOVB	P.SECT(R5),(SP)	:GET DESIRED SECTOR
8091	041104	162616				SUB	(SP)+,(SP)	:CALCULATE DIFFERENCE
8092	041106	100410				BMI	2\$:BRANCH IF IN LOWER PART OF BUFFER
8093	041110	012703	066602			MOV	#BUFADD-6,R3	:LOAD BASE OF HIGHER PART OF BUFFER
8094	041114	006316				ASL	(SP)	:MULTIPLY SECTOR COUNT BY 6
8095	041116	011646				MOV	(SP),-(SP)	
8096	041120	006316				ASL	(SP)	
8097	041122	062616				ADD	(SP)+,(SP)	
8098	041124	162603				SUB	(SP)+,R3	:CALCULATE HEADER ADDRESS
8099	041126	000407				BR	3\$:CHECK HEADER
8100								
8101	041130	005416			2\$:	NEG	(SP)	:MAKE COUNT POSITIVE
8102	041132	006316				ASL	(SP)	:MULTIPLY DIFFERENCE BY 6
8103	041134	011603				MOV	(SP),R3	
8104	041136	006316				ASL	(SP)	
8105	041140	062603				ADD	(SP)+,R3	
8106	041142	062703	066376			ADD	#HDBUFF-6,R3	:CALCULATE HEADER ADDRESS
8107	041146	012337	001646		3\$:	MOV	(R3)+,HEAD1	:GET FIRST WORD OF HEADER
8108	041152	012337	001650			MOV	(R3)+,HEAD2	:GET SECOND WORD OF HEADER
8109	041156	011337	001652			MOV	(R3),HEAD3	:GET THIRD WORD OF HEADER
8110	041162	012603				MOV	(SP)+,R3	:RESTORE R3
8111	041164	004737	037154			JSR	PC,RSTAT	:RESTORE STATUS
8112	041170	013746	001646			MOV	HEAD1,-(SP)	:STORE FIRST WORD OF THE HEADER
8113	041174	043716	001650			BIC	HEAD2,(SP)	:HD1 & ^HD2
8114	041200	013746	001650			MOV	HEAD2,-(SP)	:STORE SECOND WORD OF HEADER
8115	041204	043716	001646			BIC	HEAD1,(SP)	:^HD1 & HD2
8116	041210	052616				BIS	(SP)+,(SP)	:GENERATE EXPECTED HEADER VRC

8117	041212	023726	001652			CMP	HEAD3,(SP)+	:CHECK IF HEADER VRC IS CORRECT
8118	041216	001501				BEQ	15\$:NO, CONTINUE
8119	041220	004037	031032			JSR	RO,PR1200	:PRINT HEADER VRC ERROR
8120	041224	062660				ERR043		
8121	041226	105765	000120			TSTB	P.IERC(R5)	:CHECK IF INHIBIT ERROR RECOVERY
8122	041232	001016				BNE	5\$:YES, INDICATE UNSUCCESSFUL RECOVERY
8123	041234	105265	000147		4\$:	INCB	P.RERD(R5)	:INCREMENT REREAD COUNT
8124	041240	122765	000043	000147		CMPB	#43,P.RERD(R5)	:CHECK IF UNRECOVERABLE
8125	041246	101410				BLOS	5\$:YES, INDICATE UNRECOVERABLE ERROR
8126	041250	122765	000121	000123		CMPB	#RDDATA,P.RCMD(R5)	:CHECK IF READ DATA COMMAND
8127	041256	001407				BEQ	6\$:YES, START ERROR RECOVERY
8128	041260	122765	000021	000147		CMPB	#21,P.RERD(R5)	:CHECK IF UNRECOVERABLE WRITE
8129	041266	101021				BHI	8\$:NO, START ERROR RECOVERY
8130	041270	004737	031660		5\$:	JSR	PC,UNRECY	:INDICATE UNRECOVERABLE ERROR
8131	041274	000207				RTS	PC	:RETURN
8132								
8133	041276	122765	000021	000147	6\$:	CMPB	#21,P.RERD(R5)	:CHECK IF BEGINNING OFFSETTING OPERATION
8134	041304	001007				BNE	7\$:NO, DETERMINE PROPER OFFSET
8135	041306	010446				MOV	R4,-(SP)	:STORE R4 ON THE STACK
8136	041310	116504	000000			MOVB	P.DRVN(R5),R4	:GET DRIVE NUMBER FOR INDEX
8137	041314	146437	001514	001513		BICB	INTMSK(R4),O.OVER	:DO NOT DO IMPLIED SEEKS
8138	041322	012604				MOV	(SP)+,R4	:RESTORE R4
8139	041324	004037	037204		7\$:	JSR	RO,DETOFF	:DETERMINE OFFSET VALUE
8140	041330	041420				12\$:OFFSET ISSUED RETURN
8141	041332	116565	000123	000001	8\$:	MOVB	P.RCMD(R5),P.CMND(R5)	:STORE COMMAND
8142	041340	013765	001642	000002		MOV	RTYCYL,P.CYLN(R5)	:GET CYLINDER
8143	041346	013765	001640	000004		MOV	RTYSEC,P.SECT(R5)	:GET TRACK AND SECTOR
8144	041354	013765	001634	000010		MOV	RTYBA,P.BALO(R5)	:GET BUS ADDRESS
8145	041362	013765	001636	000012		MOV	RTYWC,P.WC(R5)	:GET WORD COUNT
8146	041370	005465	000012			NEG	P.WC(R5)	:MAKE IT POSITIVE
8147	041374	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK COMMAND
8148	041402	001003				BNE	10\$:NO, GO ISSUE COMMAND
8149	041404	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	:ISSUE WRITE BEFORE WRITE CHECK
8150	041412	004037	051564		10\$:	JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK ON COMMAND
8151	041416	001324				CINITQ		: INITIATION QUEUE
8152	041420	000207			12\$:	RTS	PC	:RETURN
8153								
8154	041422	023765	001646	000030	15\$:	CMP	HEAD1,P.DCYL(R5)	:CHECK IF CYLINDER CORRECT
8155	041430	001420				BEQ	20\$:YES, CHECK SECTOR TRACK AND FORMAT
8156	041432	004037	031032			JSR	RO,PR1200	:PRINT MISPOSITIONING
8157	041436	063473				ERR064		
8158	041440	004037	031524			JSR	RO,OPISTT	:GO TAKE STATISTICS
8159	041444	041470				17\$:UNRECOVERABLE ERROR RETURN
8160	041446	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	:SET RECALIBRATE ISSUED
8161	041454	112765	000113	000001		MOVB	#RECAL,P.CMND(R5)	:ISSUE RECALIBRATE
8162	041462	004037	051564			JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN
8163	041466	001324				CINITQ		: COMMAND INITIATION QUEUE
8164	041470	000207			17\$:	RTS	PC	:RETURN
8165								
8166	041472	005046			20\$:	CLR	-(SP)	:MAKE ROOM ON STACK
8167	041474	113766	001641	000001		MOVB	RTYTRK,1(SP)	:GET TRACK
8168	041502	006216				ASR	(SP)	:SHIFT RIGHT 3 BITS
8169	041504	006216				ASR	(SP)	
8170	041506	006216				ASR	(SP)	
8171	041510	153716	001640			BISB	RTYSEC,(SP)	:GENERATE EXPECTED SECOND WORD
8172	041514	052716	140000			BIS	#140000,(SP)	:PUT IN GO TRACK BITS

Line	Address	Code	Param	Param	Param	Code	Comment
8173	041520	022637	001650			CMP (SP)+,HEAD2	:CHECK IF SECOND WORD IS CORRECT
8174	041524	001407				BEQ 25\$:CHECK FOR BAD SECTOR
8175	041526	004037	031032			JSR R0,PRI200	:INDICATE OPERATION INCOMPLETE
8176	041532	062641				ERR042	
8177	041534	004037	031524			JSR R0,OPISTT	:GO TAKE STATISTICS
8178	041540	041470				17\$:UNRECOVERABLE RETURN
8179	041542	000673				BR 8\$:GO ISSUE LAST COMMAND
8180							
8181	041544	004037	031032		25\$:	JSR R0,PRI200	:TYPE HEADER ERROR
8182	041550	064630				ERR209	
8183	041552	000137	041234			JMP 4\$:GO TRY TO RETRY COMMAND
8184							
8185	041556	122765	000021	000147	EC6\$:	CMPB #21,P.RERD(R5)	:CHECK IF REISSUE DATA TRANSFER
8186	041564	101036				BHI 10\$:YES, INDICATE SUCCESSFUL RECOVERY
8187	041566	001424				BEQ 5\$:NO, OFFSET = +400 MICRO-INCHES
8188							:ISSUE DATA TRANSFER
8189	041570	122765	000024	000147		CMPB #24,P.RERD(R5)	:CHECK IF OFFSET -400 MICRO-INCHES
8190	041576	001420				BEQ 5\$:YES, ISSUE DATA TRANSFER
8191	041600	122765	000027	000147		CMPB #27,P.RERD(R5)	:CHECK IF OFFSET +800 MICRO-INCHES
8192	041606	001414				BEQ 5\$:YES, ISSUE DATA TRANSFER
8193	041610	122765	000032	000147		CMPB #32,P.RERD(R5)	:CHECK IF OFFSET -800 MICRO-INCHES
8194	041616	001410				BEQ 5\$:YES, ISSUE DATA TRANSFER
8195	041620	122765	000035	000147		CMPB #35,P.RERD(R5)	:CHECK IF OFFSET +1200 MICRO-INCHES
8196	041626	001404				BEQ 5\$:YES, ISSUE DATA TRANSFER
8197	041630	122765	000040	000147		CMPB #40,P.RERD(R5)	:CHECK IF OFFSET -1200 MICRO-INCHES
8198	041636	001011				BNE 10\$:NO, INDICATE SUCCESSFUL RECOVERY
8199	041640	105265	000147		5\$:	INCB P.RERD(R5)	:INCREMENT REREAD COUNT
8200	041644	116565	000123	000001		MOVB P.RCMD(R5),P.CMND(R5)	:REISSUE COMMAND
8201	041652	004037	051564			JSR R0,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
8202	041656	001324				CINITQ	:INITIATION QUEUE
8203	041660	000207				RTS PC	:RETURN
8204							
8205	041662	032765	040000	000150	10\$:	BIT #BIT14,P.DSTT(R5)	:CHECK IF SERVICING BAD SECTOR
8206	041670	001403				BEQ 11\$:NO, CONTINUE
8207	041672	005765	000146			TST P.RECT(R5)	:CHECK IF IN RETRY SEQUENCE
8208	041676	001472				BEQ EC7\$:NO, GO TO BAD SECTOR SERVICE ROUTINE
8209	041700	105765	000210		11\$:	TSTB P.ASSN(R5)	:CHECK IF DRIVE IS BEING ASSIGNED
8210	041704	001403				BEQ 12\$:NO, CHECK IF SEEK TO PREVIOUS CYLINDER
8211	041706	004737	036454			JSR PC,SURECY	:INDICATE SUCCESSFUL RECOVERY
8212	041712	000207				RTS PC	:RETURN
8213							
8214	041714	122765	000117	000001	12\$:	CMPB #SEEK,P.CMND(R5)	:CHECK IF SEEK TO PREVIOUS CYLINDER
8215	041722	001404				BEQ 14\$:YES, INDICATE SUCCESSFUL RECOVERY
8216	041724	122765	000140	000001		CMPB #RELEAS,P.CMND(R5)	:CHECK IF RELEASE
8217	041732	001003				BNE 16\$:NO, PROCESS DATA TRANSFER
8218	041734	004737	036454		14\$:	JSR PC,SURECY	:INDICATE SUCCESSFUL RECOVERY
8219	041740	000207				RTS PC	:RETURN
8220							
8221	041742	004037	025210		16\$:	JSR R0,CHKADD	:CHECK SECTOR, TRACK, CYLINDER, BUS ADDRESS
8222							:AND WORD COUNT
8223	041746	042004				20\$:ERROR RETURN
8224	041750	005037	177776			CLR PS	:ALLOW RK06 INTERRUPTS
8225	041754	122765	000121	000001		CMPB #RDATA,P.CMND(R5)	:CHECK IF READ DATA
8226	041762	001021				BNE 30\$:NO, CHECK IF WRITE CHECK
8227	041764	004037	023462			JSR R0,BUFER1	:FIND FIRST DATA ERROR
8228	041770	042006				25\$:ERROR RETURN

8229	041772	013737	001452	177776	18\$:	MOV	RKPRI,PS	:LOCK OUT RK06 INTERRUPTS
8230	042000	004737	036454			JSR	PC,SURECY	:INDICATE SUCCESSFUL RECOVERY
8231	042004	000207			20\$:	RTS	PC	:RETURN
8232								
8233	042006	013737	001452	177776	25\$:	MOV	RKPRI,PS	:LOCK OUT RK06 INTERRUPTS
8234	042014	052765	000004	000212		BIS	#BIT2,P.ERR(R5)	:INDICATE DATA COMPARE ERROR
8235	042022	000137	035466			JMP	E10\$:PROCESS DATA COMPARE ERROR
8236								
8237	042026	122765	000131	000001	30\$:	CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK
8238	042034	001356				BNE	18\$:NO, INDICATE SUCCESSFUL RECOVERY
8239	042036	032765	000200	000014		BIT	#W.WCK,P.PRST(R5)	:CHECK IF WRITE CHECK ISSUED
8240	042044	001752				BEQ	18\$:YES, INDICATE SUCCESSFUL RECOVERY
8241	042046	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:CLEAR WRITE BEFORE WRITE CHECK
8242	042054	004037	051564			JSR	RO,Q.PUSH	:ENQUEUE COMMAND IN COMMAND INITIATION QUEUE
8243	042060	001324				CINITQ		
8244	042062	000207				RTS	PC	:RETURN
8245								
8246	042064	004037	025210		EC7\$:	JSR	RO,CHKADD	:CHECK IF BUS ADD, WORD COUNT, SECTOR, : TRACK AND CYLINDER ARE CORRECT
8247								:ERROR RETURN
8248	042070	042156				9\$		
8249	042072	042765	040000	000150		BIC	#BIT14,P.DSTT(R5)	:CLEAR SERVICING BAD SECTOR
8250	042100	004737	042320			JSR	PC,BSTRAN	:CALCULATE NUMBER OF WORDS TRANSFERRED
8251	042104	013765	001660	000012		MOV	BSWC,P.WC(R5)	:LOAD BUS ADDRESS FOR NEXT DATA TRANSFER
8252	042112	100022				BPL	10\$:CHECK IF DATA TRANSFER FINISHED
8253	042114	013765	001654	000004		MOV	BSSECT,P.SECT(R5)	:LOAD SECTOR AND TRACK FOR NEXT COMMAND
8254	042122	013765	001656	000002		MOV	BSCYLN,P.CYLN(R5)	:LOAD CYLINDER FOR NEXT COMMAND
8255	042130	013765	001662	000010		MOV	BSBA,P.BALO(R5)	:LOAD BUS ADDRESS FOR NEXT DATA TRANSFER
8256	042136	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	:GO ISSUE WRITE
8257	042144	004037	051564		5\$:	JSR	RO,Q.PUSH	:ENQUEUE COMMAND IN COMMAND INITIATION : QUEUE
8258	042150	001324				CINITQ		
8259	042152	005037	001566			CLR	ERRPRO	:CLEAR ERROR PROCESSING FLAG
8260	042156	000207			9\$:	RTS	PC	:RETURN
8261								
8262	042160	105765	000210		10\$:	TSTB	P.ASSN(R5)	:CHECK IF PACK IS BEING WRITTEN
8263	042164	001012				BNE	13\$:YES, CHECK IF SEEK TO NEXT CYLINDER
8264	042166	004737	023020			JSR	PC,BUFREL	:RELEASE BUFFER
8265	042172	004037	051564			JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN DRIVE
8266	042176	001314				AVAILQ		: AVAILIABLE QUEUE
8267	042200	004737	022520			JSR	PC,GETBUF	:GET NEXT BUFFER FOR DAT TRANSFER
8268	042204	005037	001566			CLR	ERRPRO	:CLEAR ERROR BEING PROCESSING
8269	042210	000207				RTS	PC	:RETURN
8270								
8271	042212	005037	001566		13\$:	CLR	ERRPRO	:CLEAR ERROR BEING PROCESSED
8272	042216	013765	001654	000026		MOV	BSSECT,P.DTS(R5)	:LOAD NEXT SECTOR AND TRACK
8273	042224	013765	001656	000030		MOV	BSCYLN,P.DCYL(R5)	:LOAD NEXT CYLINDER
8274	042232	032777	040000	136700		BIT	#SW14,ASWR	:CHECK IF LOOP ON CURRENT OPERATION
8275	042240	001002				BNE	14\$:YES, ISSUE SEEK TO PREVIOUS CYLINDER
8276	042242	000137	015454			JMP	WV0\$:NO, CONTINUE TO WRITE PACK
8277								
8278	042246	004737	023020		14\$:	JSR	PC,BUFREL	:RELEASE BUFFER
8279	042252	032765	002000	000014		BIT	#DRPDRV,P.PRST(R5)	:CHECK IF DROP DRIVE
8280	042260	001402				BEQ	15\$:NO ISSUE SEEK TO PREVIOUS CYLINDER
8281	042262	000137	016216			JMP	WV1\$:DROP DRIVE
8282								
8283	042266	112765	177777	000211	15\$:	MOVB	#-1,P.SEEK(R5)	:SET SEEK TO PREVIOUS CYLINDER
8284	042274	016565	000136	000002		MOV	P.LCYL(R5),P.CYLN(R5)	:GET LAST CYLINDER

8285	042302	016565	000140	000004
8286	042310	112765	000117	000001
8287	042316	000712		

MOV	P.LSEC(R5),P.SECT(R5) ;GET LAST TRACK AND SECTOR
MOVB	#SEEK,P.CMND(R5) ;LOAD COMMAND
BR	5\$;GO ISSUE SEEK COMMAND

```

8288      .SBTTL  CALCULATE WORDS TRANSFERRED UNTIL BAD SECTOR
8289
8290 042320 016537 000026 001654 BSTRAN: MOV      P.DTS(R5),BSSECT ;STORE ADDRESS OF BAD SECTOR
8291 042326 016537 000030 001656      MOV      P.DCYL(R5),BSCYLN
8292 042334 105237 001654      INCB     BSSECT      ;ADVANCE TO NEXT SECTOR ON PACK
8293 042340 122737 000025 001654      CMPB    #21.,BSSECT ;CHECK FOR SECTOR OVERFLOW
8294 042346 103014      BHIS     5$          ;NO, CONTINUE
8295 042350 105037 001654      CLRB    BSSECT     ;CLEAR SECTOR
8296 042354 105237 001655      INCB    BSTRCK     ;INCREMENT TRACK
8297 042360 122737 000002 001655      CMPB    #2,BSTRCK  ;CHECK FOR TRACK OVERFLOW
8298 042366 103004      BHIS     5$          ;NO, CONTINUE
8299 042370 105037 001655      CLRB    BSTRCK     ;CLEAR TRACK
8300 042374 005237 001656      INC     BSCYLN     ;INCREMENT CYLINDER
8301 042400 013746 001656      5$:  MOV     BSCYLN,-(SP) ;STORE CYLINDER ON STACK
8302 042404 166516 000002      SUB     P.CYLN(R5),(SP) ;SUBTRACT STARTING CYLINDER
8303 042410 011646      MOV     (SP),-(SP)    ;SAVE DIFFERENCE
8304 042412 006316      ASL     (SP)         ;MULTIPLY CYLINDER BY 3
8305 042414 062616      ADD     (SP)+,(SP)   ;
8306 042416 005046      CLR     -(SP)        ;MAKE ROOM ON STACK
8307 042420 113716 001655      MOVB    BSTRCK,(SP)  ;PLACE TRACK ON STACK
8308 042424 062616      ADD     (SP)+,(SP)   ;ADD TRACK
8309 042426 005046      CLR     -(SP)        ;MAKE ROOM ON THE STACK
8310 042430 116516 000005      MOVB    P.TRCK(R5),(SP) ;STORE TRACK
8311 042434 162616      SUB     (SP)+,(SP)   ;SUBTRACT INITIAL TRACK
8312 042436 012746 000026      MOV     #22,-(SP)    ;STORE 22 ON STACK FOR MULTIPLICATION
8313 042442 004737 055210      JSR     PC,$MULT     ;MULTIPLY BY 22
8314 042446 005046      CLR     -(SP)        ;MAKE ROOM ON THE STACK
8315 042450 113716 001654      MOVB    BSSECT,(SP)  ;STORE BAD SECTOR
8316 042454 062616      ADD     (SP)+,(SP)   ;ADD BAD SECTOR ADDRESS
8317 042456 005046      CLR     -(SP)        ;MAKE ROOM ON THE STACK
8318 042460 116516 000004      MOVB    P.SECT(R5),(SP) ;STORE ORIGINAL SECTOR ADDRESS
8319 042464 162616      SUB     (SP)+,(SP)   ;SUBTRACT ORIGINAL SECTOR ADDRESS
8320 042466 111666 000001      MOVB    (SP),1(SP)   ;KEEP NUMBER OF SECTORS TRANSFERRED
8321 042472 105016      CLRB    (SP)
8322 042474 011637 001664      MOV     (SP),BSWORD ;LOAD WORD COUNT
8323 042500 006316      ASL     (SP)         ;MULTIPLY BY 2
8324 042502 066516 000010      ADD     P.BALO(R5),(SP) ;GET STARTING ADDRESS FOR NEW DATA
8325 042506 011637 001662      MOV     (SP),BSBA    ;GET NEW BUS ADDRESS
8326 042512 166516 000132      SUB     P.RBAL(R5),(SP) ;GET NUMBER OF BYTES TRANSFERRED
8327 042516 006216      ASR     (SP)         ;CONVERT TO WORDS (DIVIDE BY 2)
8328 042520 066516 000130      ADD     P.RWC(R5),(SP) ;DETERMINE REMAINING WORD COUNT
8329 042524 012637 001660      MOV     (SP)+,BSWC   ;STORE CALCULATED WORD COUNT
8330 042530 005726      TST     (SP)+        ;THROW AWAY MOST SIGNIFICANT BITS
8331 042532 000207      RTS     PC           ;RETURN

```

```

8332          .SBTTL  BAD SECTOR HANDLING
8333
8334 042534 032765 040000 000150 BDSECT: BIT  #BIT14,P.DSTT(R5) ;CHECK IF PRESENTLY SERVICING BAD SECTOR
8335 042542 001401          BEQ  5$          ;NO, CONTINUE
8336 042544 000000          HALT
8337 042546 132765 000010 000210 5$: BITB  #BIT3,P.ASSN(R5) ;CHECK IF READING PACK SERIAL NUMBER
8338 042554 001157          BNE  40$          ;YES, DETERMINE NEXT SECTOR
8339 042556 004737 042320          JSR  PC,BSTRAN    ;CALCULATE WORDS TRANSFERRED
8340 042562 122765 000121 000001 CMPB  #RDDATA,P.CMND(R5) ;CHECK IF READ COMMAND
8341 042570 001076          BNE  25$          ;NO, DO NO CHECK BUS ADDRESS AND WORD COUNT
8342 042572 013746 001662          MOV  BSBA,-(SP)    ;STORE BUS ADDRESS
8343 042576 162716 001000          SUB  #256,*2,(SP) ;SUBTRACT ADDITIONAL SECTOR
8344 042602 026526 000024          CMP  P.BAR(R5),(SP)+ ;CHECK IF BUS ADDRESS CORRECT
8345 042606 001404          BEQ  10$          ;YES, CONTINUE
8346 042610 004037 037500          JSR  R0,CONERR   ;PRINT 'BUS ADDRESS INVALID'
8347 042614 063612          ERR071
8348 042616 000207          RTS  PC          ;RETURN
8349
8350 042620 016546 000022          10$: MOV  P.WCR(R5),-(SP) ;STORE WORD COUNT FOR COMPARISON
8351 042624 062716 000400          ADD  #256,(SP)    ;SUBTRACT OFF BAD SECTOR
8352 042630 023726 001660          CMP  BSWC,(SP)+  ;CHECK IF WORD COUNT CORRECT
8353 042634 001404          BEQ  15$          ;YES, CONTINUE
8354 042636 004037 037500          JSR  R0,CONERR   ;PRINT 'WORD COUNT INVALID'
8355 042642 063567          ERR070
8356 042644 000207          RTS  PC          ;RETURN
8357
8358 042646 042765 100200 000150 15$: BIC  #BIT15!BIT7,P.DSTT(R5) ;CLEAR FIRST ERROR AND ERROR ENQUEUED
8359 042654 022737 000400 001664          CMP  #256.,BSWORD ;CHECK IF NO WORDS TRANSFERRED
8360 042662 001413          BEQ  20$          ;YES, DO NOT CHECK DATA
8361 042664 004037 023462          JSR  R0,BUFER1   ;CHECK DATA READ
8362 042670 043224          50$          ;ERROR RETURN
8363 042672 063765 001664 000012          ADD  BSWORD,P.WC(R5) ;CALCULATE NEW WORD COUNT
8364 042700 100410          BMI  22$          ;CHECK IF DATA TRANSFER NOT COMPLETE
8365 042702 000464          BR   30$          ;DATA TRANSFER COMPLETE
8366
8367 042704 052765 000200 000014 19$: BIS  #W.WCK,P.PRST(R5) ;ISSUE WRITE BEFORE WRITE CHECK
8368 042712 013765 001660 000012 20$: MOV  BSWC,P.WC(R5)  ;GET NUMBER OF WORDS TO BE TRANSFERRED
8369 042720 100055          BPL  30$          ;CHECK IF DATA TRANSFER COMPLETE
8370 042722 013765 001656 000002 22$: MOV  BSCYLN,P.CYLN(R5) ;GET NEW CYLINDER
8371 042730 013765 001654 000004          MOV  BSSECT,P.SECT(R5) ;GET NEW SECTOR
8372 042736 013765 001662 000010          MOV  BSBA,P.BALO(R5) ;GET NEW BUS ADDRESS
8373 042744 004037 051564          JSR  R0,Q.PUSH   ;PUT PARAMETER IN COMMAND
8374 042750 001324          CINITQ          ; INITIATION QUEUE
8375 042752 005765 000146          TST  P.RECT(R5)  ;CHECK IF ERROR RECOVERY IN PROGRESS
8376 042756 001002          BNE  23$          ;YES, DO NOT RESET ERROR PROCESSING IN PROGRESS
8377 042760 005037 001566          CLR  ERRPRO      ;RESET ERROR PROCESSING IN PROGRESS
8378 042764 000207          23$: RTS  PC          ;RETURN
8379
8380 042766 042765 100200 000150 25$: BIC  #BIT15!BIT7,P.DSTT(R5) ;CLEAR FIRST ERROR AND ERROR ENQUEUED
8381 042774 122765 000131 000001          CMPB #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
8382 043002 001343          BNE  20$          ;NO, CONTINUE
8383 043004 022737 000400 001664          CMP  #256.,BSWORD ;CHECK IF NO WORDS TRANSFERRED
8384 043012 001734          BEQ  19$          ;YES, ISSUE NEXT DATA TRANSFER
8385 043014 012746 000400          MOV  #256.,-(SP) ;STORE EXCESS WORD COUNT ON STACK
8386 043020 163716 001664          SUB  BSWORD,(SP) ;CALCULATE NEW WORD COUNT
8387 043024 012665 000012          MOV  (SP)+,P.WC(R5) ;STORE NEW WORD COUNT

```

8388	043030	052765	040000	000150		BIS	#BIT14,P.DSTT(R5)	:INDICATE THAT BAD
8389								:SECTOR IS BEING PROCESSED
8390	043036	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:DO WRITE CHECK PART OF COMMAND
8391	043044	004037	051564			JSR	RO,Q.PUSH	:PUT PARAMETER BLOCK IN COMMAND
8392	043050	001324				CINITQ		:INITIATION QUEUE
8393	043052	000207				RTS	PC	:RETURN
8394								
8395	043054	004737	023020		30\$:	JSR	PC,BUFREL	:RELEASE BUFFER
8396	043060	005765	000146			TST	P.RECT(R5)	:DETERMINE IF CURRENTLY UNDERGOING ERROR
8397								:RECOVERY
8398	043064	001403				BEQ	35\$:YES, PUT IN AVAILIABLE QUEUE
8399	043066	004737	036454			JSR	PC,SURECY	:INDICATE SUCCESSFUL RECOVERY
8400	043072	000207				RTS	PC	:RETURN
8401								
8402	043074	004037	051564		35\$:	JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN
8403	043100	001314				AVAILQ		:AVAILIABLE QUEUE
8404	043102	004737	022520			JSR	PC,GETBUF	:GET NEXT BUFFER
8405	043106	005037	001566			CLR	ERRPRO	:CLEAR ERROR BEING PROCESSED
8406	043112	000207				RTS	PC	:RETURN
8407								
8408	043114	042765	100200	000150	40\$:	BIC	#BIT15!BIT7,P.DSTT(R5)	:CLEAR FIRST ERROR AND ERROR ENQUEUED
8409	043122	122737	000010	000126		CMPB	#8.,P.RSEC	:CHECK IF ALL SECTORS HAVE BEEN READ
8410	043130	101422				BLOS	45\$:YES, DROP DRIVE FROM TEST SEQUENCE
8411	043132	116565	000123	000135		MOVB	P.RCMD(R5),P.LCMD(R5)	:STORE PREVIOUS COMMAND
8412	043140	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	
8413	043146	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	
8414	043154	016537	000130	000142		MOV	P.RWC(R5),P.L:IC	
8415	043162	004037	051564			JSR	RO,Q.PUSH	:ENQUEUE PARAMETER BLOCK IN COMMAND
8416	043166	001324				CINITQ		:INITIATION QUEUE
8417	043170	005037	001566			CLR	ERRPRO	:CLEAR ERROR RECOVERY IN PROGRESS
8418	043174	000207				RTS	PC	:RETURN
8419								
8420	043176	004737	023020		45\$:	JSR	PC,BUFREL	:GO RELEASE BUFFER
8421	043202	104401	064410			TYPE	,ERR204	
8422	043206	004737	013162			JSR	PC,DROP	:DROP DRIVE FROM TEST SEQUENCE
8423	043212	004737	022520			JSR	PC,GETBUF	:GET NEW BUFFER
8424	043216	005037	001566			CLR	ERRPRO	:CLEAR ERROR RECOVERY IN PROGRESS
8425	043222	000207				RTS	PC	:RETURN
8426								
8427	043224	052765	000004	000212	50\$:	BIS	#BIT2,P.ERR(R5)	:SET DATA TYPE ERROR
8428	043232	000137	035466			JMP	E10\$:GO PROCESS DATA ERROR

```

8429          .SBTTL  ERROR RECOVERY SEQUENCE ABNORMAL RETURN
8430
8431 043236 032765 000120 000150 ERCVY2: BIT      #BIT4!BIT6,P.DSTT(R5) ;CHECK IF DIAGNOSING SEQUENCE
8432 043244 001524          BEQ      35$          ;NO, GO TO ERROR RECOVERY SEQUENCE
8433 043246 004737 030156          JSR     PC,PRI000      ;PRINT HEADER
8434 043252 104401 064577          TYPE   ,ERR208        ;TYPE 'ERROR WHILE DIAGNOSING ----'
8435 043256 032765 000020 000150 BIT      #BIT4,P.DSTT(R5) ;CHECK IF HEADER TYPE ERROR
8436 043264 001005          BNE     2$          ;YES, PRINT MESSAGE
8437 043266 104401 063034          TYPE   ,ERR052+1     ;TYPE 'TIME OUT WHILE DRIVE POSITION'
8438 043272 104401 001165          TYPE   $CRLF         ;TYPE <CR><LF>
8439 043276 000402          BR      3$          ;GO PRINT STATUS
8440
8441 043300 104401 064631          2$:    TYPE   ,ERR209+1     ;PRINT HEADER TYPE ERROR
8442 043304 032777 020000 135626 3$:    BIT      #SW13,@SWR    ;CHECK IF INHIBIT PRINT OUT
8443 043312 001066          BNE     30$         ;YES, DO NOT PRINT STATUS
8444 043314 032765 000020 000212 BIT      #BIT4,P.ERR(R5) ;CHECK IF DRIVE SEIZED TIME OUT
8445 043322 001402          BEQ     4$
8446 043324 104401 063351          TYPE   ,ERR059
8447 043330 032765 000040 000212 4$:    BIT      #BIT5,P.ERR(R5) ;CHECK IF TIME OUT WHILE DRIVE POSITIONING
8448 043336 001402          BEQ     5$
8449 043340 104401 063034          TYPE   ,ERR052+1
8450 043344 032765 000010 000014 5$:    BIT      #UEXATT,P.PRST(R5) ;CHECK IF UNEXPECTED ATTENTION
8451 043352 001402          BEQ     6$
8452 043354 104401 062430          TYPE   ,ERR030
8453 043360 032765 000020 000014 6$:    BIT      #DRVHRD,P.PRST(R5) ;CHECK IF DRIVE HARD ERROR
8454 043366 001402          BEQ     7$
8455 043370 104401 062321          TYPE   ,ERR027
8456 043374 032765 000040 000014 7$:    BIT      #DRVDSC,P.PRST(R5) ;CHECK IF DSC DID NOT CLEAR
8457 043402 001402          BEQ     8$
8458 043404 104401 062340          TYPE   ,ERR028
8459 043410 032765 004000 000014 8$:    BIT      #NODSC,P.PRST(R5) ;CHECK IF ATTN BUT NO DSC OR FAULT
8460 043416 001402          BEQ     11$
8461 043420 104401 063274          TYPE   ,ERR058
8462 043424 004737 026252          11$:   JSR     PC,PRISTT    ;PRINT CONTROLLER STATUS
8463 043430 004737 026076          JSR     PC,PRDSTT    ;PRINT DRIVE STATUS
8464 043434 004737 030422          JSR     PC,PRI002    ;PRINT RETRY COMMAND
8465 043440 010546          MOV     R5,-(SP)     ;STORE R5 ON STACK
8466 043442 012705 004516          MOV     #PARM10,R5  ;LOAD R5 FOR PREVIOUS STATUS
8467 043446 104401 064650          TYPE   ,ERR210      ;TYPE 'FIRST ERROR INFO'
8468 043452 004737 026252          JSR     PC,PRISTT    ;PRINT CONTROLLER STATUS
8469 043456 004737 026076          JSR     PC,PRDSTT    ;PRINT DRIVE STATUS
8470 043462 004737 030422          JSR     PC,PRI002    ;PRINT INITIAL COMMAND
8471 043466 012605          MOV     (SP)+,R5    ;RESTORE R5
8472 043470 005265 000206          30$:   INC     P.NER(R5)  ;INCREMENT OTHER ERRORS
8473 043474 004737 031426          JSR     PC,HLT000    ;CHECK IF HALT ON ERROR
8474 043500 004737 023020          JSR     PC,BUFREL    ;RELEASE BUFFER
8475 043504 004737 013162          JSR     PC,DROP      ;DROP DRIVE
8476 043510 004737 022520          JSR     PC,GETBUF    ;GO ALLOCATE BUFFERS
8477 043514 000405          BR      40$         ;CHECK IF ERROR QUEUE EMPTY
8478
8479 043516 004737 032160          35$:   JSR     PC,ERCVY    ;GO DO ERROR RECOVERY
8480 043522 005737 001566          TST     ERRPRO       ;CHECK IF ERROR RECOVERY ON LAST DRIVE
8481
8482 043526 001007          BNE     45$         ; IS FINISHED
8483 043530 004037 051644          40$:   JSR     R0,Q.POP    ;GET NEXT ELEMENT FROM
8484 043534 001330          ERRPRQ              ; ERROR PROCESSING QUEUE

```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 PAGE 193
ERROR RECOVERY SEQUENCE ABNORMAL RETURN

I 15

SEQ 0190

8485 043536 012605
8486 043540 010537 001566
8487 043544 001364
8488
8489 043546 000207

MOV (SP)+,R5
MOV R5,ERRPRO
BNE 35\$
45\$: RTS PC

:LOAD PARAMETER BLOCK
:LOAD CURRENT PROCESSING ERROR
:IF ERROR PROCESSING QUEUE NOT EMPTY GO
: PROCESS NEXT ERROR
:RETURN


```
8490 .SBTTL DETERMINE IF CLOCK IS PRESENT
8491
8492 043550 012737 043600 000004 CHKCLK: MOV #5$,ERRVEC ;SET UP ERROR VECTOR FOR NON-EXISTENT MEMORY
8493 043556 012737 000340 000006 MOV #PR7,ERRVEC+2 ;LOCK ALL INTERRUPTS
8494 043564 005777 135546 TST @SLKCSB ;CHECK FOR P CLOCK
8495 043570 112737 177777 001374 MOVB #-1,CLKFLG ;INDICATE KW11-P PRESENT
8496 043576 000413 BR 20$ ;LOAD COMMON CLOCK PARAMETERS
8497
8498 043600 022626 5$: CMP (SP)+,(SP)+ ;ADJUST STACK
8499 043602 012737 043624 000004 MOV #15$,ERRVEC ;SET UP ERROR VECTOR FOR NON-EXISTENT MEMORY
8500 043610 005777 135526 TST @SLKS ;CHECK FOR L CLOCK
8501 043614 112737 000001 001374 MOVB #1,CLKFLG ;SET KW11-L PRESENT
```

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 K 15 PAGE 195
DETERMINE IF CLOCK IS PRESENT

SEQ 0192

8502	043622	000401			BR	20\$:REINSTATE TRAP CATCHER
8503								
8504	043624	022626			15\$: CMP	(SP)+,(SP)+		:ADJUST STACK
8505	043626	012737	000006	000004	20\$: MOV	#ERRVEC+2,ERRVEC		:REINSTATE TRAP CATCHER
8506	043634	005037	000006		CLR	ERRVEC+2		
8507	043640	000207			RTS	PC		:RETURN

```
8508      .SBTTL  SET UP CLOCK INTERRUPT
8509
8510 043642 105737 001374      CLKINT: TSTB   CLKFLG   ;CHECK IF A CLOCK IS ON SYSTEM
8511 043646 001431              BEQ     20$      ;NO, RETURN
8512 043650 100011              BPL     5$      ;CHECK IF KW11-L
8513 043652 013701 001340      MOV     $LPVEC,R1 ;LOAD VECTOR ADDRESS
8514 043656 012777 177777 135452 MOV     #-1,@$LKCSB ;LOAD COUNT BUFFER WITH 1'S
8515 043664 012777 000135 135442 MOV     #135,@$LKCSR ;SET COUNT UP, 16 MS, CONT
8516 043672 000405              BR      10$     ;LOAD COMMON CLOCK PARAMETERS
8517
8518 043674 013701 001344      5$:    MOV     $LLVEC,R1 ;LOAD VECTOR ADDRESS
8519 043700 012777 000100 135434 MOV     #100,@$LKS ;SET UP L CLOCK FOR INTERRUPT MODE
8520 043706 012721 043734      10$:   MOV     #CLOCK,(R1)+ ;LOAD ADDRESS OF INTERRUPT ROUTINE
8521 043712 012711 000300      MOV     #PR6,(R1)  ;SET UP FOR PRIORITY 6
8522 043716 113737 001352 001353 MOVB    HZ,CLKFRQ  ;SET UP CLOCK FREQUENCY
8523 043724 113737 001360 001375 MOVB    PERINV,PERIOD ;SET UP COUNTER FOR INITIAL STATISTIC SUMMARY
8524 043732 000207              RTS     PC      ;RETURN
```

```
8525      .SBTTL  KW11-L AND KW11-P INTERRUPT HANDLER
8526
8527 043734 105337 001353      CLOCK:  DECB  CLKFRQ      ;CHECK IF ONE SECOND PASTED
8528 043740 001041              BNE  10$      ;NO, RETURN
8529 043742 113737 001352 001353      MOVB  HZ,CLKFRQ ;REINITIALIZE SYSTEM CLOCK
8530 043750 005237 001372              INC  SECOND    ;INCREMENT SECOND COUNT
8531 043754 022737 000074 001372      CMP   #60.,SECOND ;CHECK IF MINUTE HAS OCCURRED
8532 043762 001030              BNE  10$      ;NO, RETURN
8533 043764 005037 001372      CLR  SECOND    ;CLEAR SECOND
8534 043770 105737 001360      TSTB PERINV    ;CHECK IF INTERVAL STATISTICS
8535 043774 001411              BEQ  5$      ;NO, UPDATE MINUTE COUNT
8536 043776 105337 001375      DECB PERIOD    ;DECREMENT PERIOD COUNT
8537 044002 001006              BNE  5$      ;CHECK IF END OF PERIOD, IF NOT UPDATE MINUTE COUNT
8538 044004 113737 001360 001375      MOVB PERINV,PERIOD ;INITIALIZE PERIOD
8539 044012 112737 177777 001633      MOVB #-1,STATIS ;INDICATE TIME FOR STATISTIC TYPE OUT
8540 044020 005237 001370      5$:  INC  MINUTE    ;INCREMENT MINUTE COUNTER
8541 044024 022737 000074 001370      CMP   #60.,MINUTE ;CHECK IF HOUR HAS OCCURED
8542 044032 001004              BNE  10$      ;NO, RETURN
8543 044034 005037 001370      CLR  MINUTE    ;CLEAR MINUTE COUNT
8544 044040 005237 001366      INC  HOUR      ;INCREMENT HOUR COUNT
8545 044044 000002      10$: RTI      ;RETURN
```

```

8546          .SBTTL PRINT TIME ROUTINE
8547
8548 044046 105737 001374      PRITIM: TSTB   CLKFLG      :CHECK IF CLOCK ON SYSTEM
8549 044052 001455              BEQ     50$      :NO, RETURN
8550 044054 010046              MOV    RO,-(SP)  :SAVE RO
8551 044056 013746 177776      MOV    PS,-(SP)  :SAVE PSW
8552 044062 012737 000340 177776  MOV    #PR7,PS   :LOCK OUT CLOCK
8553 044070 013737 001366 001376  MOV    HOUR,TIMHR :SAVE TIME FOR PRINT OUT
8554 044076 013737 001370 001400  MOV    MINUTE,TIMMIN
8555 044104 013737 001372 001402  MOV    SECOND,TIMSEC
8556 044112 012637 177776      MOV    (SP)+,PS   :RESTORE PSW
8557 044116 012700 065341      MOV    #TIM001,RO :LOAD RO FOR INDEX
8558 044122 105010              CLRB   (RO)      :DETERMINE HUNDREDS OF HOURS
8559 044124 013746 001376      MOV    TIMHR,-(SP) :STORE HOURS FOR CONVERSION
8560 044130 162716 000144      1$:   SUB    #100.,(SP) :CONVERT HUNDREDS
8561 044134 100402              BMI    2$      :CHECK IF TOO SMALL
8562 044136 105210              INCB  (RO)
8563 044140 000773              BR     1$
8564
8565 044142 062716 000144      2$:   ADD    #100.,(SP)
8566 044146 152720 000060      BISB  #60,(RO)+  :MAKE DIGIT ASCII
8567 044152 004737 044210      JSR   PC,CONHUN
8568 044156 013716 001400      MOV   TIMMIN,(SP) :LOAD STACK FOR MINUTES
8569 044162 004737 044210      JSR   PC,CONHUN
8570 044166 013716 001402      MOV   TIMSEC,(SP) :LOAD STACK FOR SECONDS
8571 044172 004737 044210      JSR   PC,CONHUN
8572 044176 005726              TST   (SP)+      :ADJUST STACK
8573 044200 104401 065334      TYPE  ,TIM000    :TYPE TIME:
8574 044204 012600              MOV   (SP)+,RO   :RESTORE RO
8575 044206 000207      50$:  RTS     PC     :RETURN
8576
8577          .SBTTL CONVERT DECIMAL LESS THAN 100
8578
8579 044210 105010              CONHUN: CLRB   (RO)      :ZERO DIGIT
8580 044212 162766 000012 000002  1$:   SUB    #10.,2(SP) :CONVERT TENS
8581 044220 100402              BMI    5$
8582 044222 105210              INCB  (RO)
8583 044224 000772              BR     1$
8584
8585 044226 062766 000012 000002  5$:   ADD    #10.,2(SP)
8586 044234 152720 000060      BISB  #60,(RO)+  :MAKE ASCII
8587 044240 105010              CLRB  (RO)      :ZERO DIGIT
8588 044242 005766 000002      TST   2(SP)
8589 044246 001404              BEQ   7$
8590 044250 105210              6$:   INCB  (RO)      :CONVERT ONES
8591 044252 005366 000002      DEC   2(SP)
8592 044256 001374              BNE   6$
8593 044260 152720 000060      7$:   BISB  #60,(RO)+  :MAKE ASCII
8594 044264 105720              TSTB (RO)+      :SKIP OVER COLON
8595 044266 000207      RTS     PC

```

8596
8597
8598
8599
8600
8601
8602
8603
8604
8605
8606
8607
8608
8609
8610
8611
8612
8613
8614
8615
8616
8617
8618
8619
8620
8621
8622
8623
8624
8625
8626
8627
8628
8629
8630
8631
8632
8633
8634
8635
8636
8637
8638
8639
8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651

.SBTTL RK611/RK06-RK07 UNIBUS DRIVER FOR QUEUED OPERATIONS (REV. 0.11)

:*COPYRIGHT (C) 1975
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MA. 01754
:*AUTHOR: ROY SPITZER

.SBTTL *WATCH-DOG TIMER

*
* THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06/07 UNIBUS
* SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
* REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
* THE RK06 DRIVER WILL USE THE LOCATION W.MTIM FOR
* MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
* WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
* DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
* (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
* IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
* TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
* REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
*
* THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
* OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
* THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
* LIMIT FOR ALL OTHER COMMANDS.
*
* FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
* WATCH UP TO 8 OPERATIONS SIMULTEOUSLY. FOR SEQUENTIAL
* OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
*
*CALL JSR PC,W.WTCH
* RETURN IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
*
* OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
* BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
* IN THE PROGRAM DEVICE STATUS REGISTER OF THE
* APPROPRIATE PARAMETER BLOCK WILL BE SET.
*

W.WTCH: MOV R5,-(SP) ;SAVE R5 ON THE STACK
MOV R4,-(SP) ;SAVE R4 ON THE STACK
MOV R3,-(SP) ;SAVE R3 ON THE STACK
MOV R2,-(SP) ;SAVE R2 ON STACK
MOV PS,-(SP) ;SAVE PROGRAM STATUS WORD ON STACK
DEC W.MTIM ;DECREMENT MILLISECOND TIMER
BNE 20\$;IF NOT ZERO RETURN
MOV W.MILI,W.MTIM ;REINITIALIZE MILLISECOND TIMER
MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS
MOV RKBAS,R2 ;LOAD BASE OF RK06 REGISTERS
CLR R4 ;CLEAR INDEX OF DRIVE MASKS
CLR R3 ;CLEAR INDEX OF DRIVE TIMES
; AND PARAMETER BLOCK ADDRESSES
BITB INTMSK(R4),W.TIME ;CHECK IF DRIVE IS BEING TIMED

044270 010546
044272 010446
044274 010346
044276 010246
044300 013746 177776
044304 005337 001466
044310 001046
044312 013737 001470 001466
044320 013737 001452 177776
044326 013702 001446
044332 005004
044334 005003
044336 136437 001514 001512 1\$:

8652	044344	001422			BEG	6\$:NO, GO TO NEXT DRIVE
8653	044346	005363	001544		DEC	W.DRV(R3)		:DECREMENT DRIVE COUNT
8654	044352	001017			BNE	6\$:IF NOT TIME OUT, GO TO
8655								: NEXT DRIVE
8656	044354	016305	001524		MOV	PBLKT(R3),R5		:LOAD ADDRESS OF PARAMETER
8657								: BLOCK TABLE FOR INDEXING
8658	044360	146437	001514	001512	BICB	INTMSK(R4),W.TIME		:RESET TIMING INDICATOR FOR DRIVE
8659	044366	052765	000100	000014	BIS	#CMDTO,P.PRST(R5)		:SET COMMAND TIME OUT
8660	044374	020537	001464		CMP	R5,O.WAIT		:CHECK IF DRIVER IS WAITING FOR
8661								: COMMAND COMPLETION
8662	044400	001002			BNE	5\$:NO, DO NOT ALTER WAITING FOR
8663								: COMMAND COMPLETION
8664	044402	005037	001464		CLR	O.WAIT		:CLEAR WAIT FOR COMMAND COMPLETION
8665	044406	004777	135044	5\$:	JSR	PC,@A.ABNL		:BRANCH TO ERROR ROUTINE
8666	044412	062703	000002	6\$:	ADD	#2,R3		:ADD 2 TO INDEX OF DRIVE TIMES
8667								: AND PARAMETER BLOCK ADDRESSES
8668	044416	005204			INC	R4		:INCREMENT INDEX OF DRIVE MASK
8669	044420	022704	000010		CMP	#10,R4		:CHECK IF ALL DRIVES ARE TESTED
8670	044424	001344			BNE	1\$:NO, CHECK NEXT DRIVE
8671	044426	012637	177776	20\$:	MOV	(SP)+,PS		:RESTORE PSW
8672	044432	012602			MOV	(SP)+,R2		:RESTORE R2
8673	044434	012603			MOV	(SP)+,R3		:RESTORE R3
8674	044436	012604			MOV	(SP)+,R4		:RESTORE R4
8675	044440	012605			MOV	(SP)+,R5		:RESTORE R5
8676	044442	000207			RTS	PC		:RETURN

8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732

.SBTTL *RK06-RK07 INTERRUPT SERVICE ROUTINE

THIS ROUTINE WILL SERVICE ALL RK06/07 INTERRUPTS.

UPON RECEIVING AN INTERRUPT, THIS ROUTINE WILL PERFORM ONE OF THE FOLLOWING SERVICES:

- 1.) SERVICE PORT WAS SEIZED BY OTHER PORT
- 2.) SERVICE DRIVER IS WAIT FOR COMMAND COMPLETION
- 3.) SERVICE POSITIONING COMPLETION
- 4.) REQUEUE COMMAND IF DRIVE WAS RELEASED FOR THE QUEUED RK06 DRIVER.
- 5.) IF NO SERVICE IS REQUIRED, THE COMMAND WILL BE ISSUED FOR THE QUEUED RK06 DRIVER.

THREE LINKS ARE PROVIDED TO THE DRIVING PROGRAM. THEY ARE:

- 1.) A.NORM ADDRESS OF NORMAL RETURN (SUCESSFUL COMPLETION OF COMMAND)
- 2.) A.ABNL ADDRESS OF ABNORMAL RETURN (UNSUCCESSFUL COMPLETION OF COMMAND)
- 3.) A.CONT ADDRESS OF CONTROL ERROR RETURN

FOR NORMAL AND ABNORMAL RETURNS, THE ADDRESS OF THE APPROPRIATE PARAMETER BLOCK WILL BE IN R5.

FOR THE CONTROLLER ERROR RETURN, THE LOCATION E.CONT CONTAINS THE REASON FOR THE CONTROLLER ERROR.

ROUTINES USED:

- C.OPT (QUEUED ONLY)
- Q.PUSH (QUEUED ONLY)
- Q.RMOV (QUEUED ONLY)
- R.CONT (SEQUENTIAL ONLY)
- R.NORM (SEQUENTIAL ONLY)
- R.ABNL (SEQUENTIAL ONLY)
- I.CSTS
- I.STAT
- I.ISSU
- I.CCLR

044444	010546		
044446	010446		
044450	010346		
044452	010246		
044454	010146		
044456	010046		
044460	013702	001446	
044464	016237	000010	001412
044472	032737	001000	001412
044500	001407		
044502	052737	100000	001462
044510	004777	134744	

```

I.INTR: MOV R5,-(SP) ;STORE R5 ON THE STACK
        MOV R4,-(SP) ;STORE R4 ON THE STACK
        MOV R3,-(SP) ;STORE R3 ON THE STACK
        MOV R2,-(SP) ;STORE R2 ON THE STACK
        MOV R1,-(SP) ;STORE R1 ON THE STACK
        MOV R0,-(SP) ;STORE R0 ON THE STACK
        MOV RKBAS,R2 ;LOAD R2 TO ADDRESS RK06 REGISTER
        MOV RKCS2(R2),T.CS2 ;STORE CS2
        BIT #MDS,T.CS2 ;CHECK IF MULTIPLE DRIVE SELECT
        BEQ 1$ ;NO, CONTINUE PROCESSING
        BIS #E.MDS,E.CONT ;SET MULTIPLE DRIVE SELECT
        JSR PC,@A.CONT ;REPORT ERROR

```



```

8733 044514 000137 047052          JMP      I.RTRN          ;RETURN
8734
8735 044520 105737 001506          1$:    TSTB      I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
8736 044524 001410                    BEQ      6$              ;NO, CHECK IF DRIVE AVAILABLE
8737 044526 100403                    BMI      5$              ;CHECK IF RELEASE COMMAND
8738 044530 105037 001506          CLRB      I.ISRL          ;YES, CLEAR FLAG
8739 044534 000425                    BR       I.I00          ;CONTINUE PROCESSING INTERRUPT
8740
8741 044536 105037 001506          5$:    CLRB      I.ISRL          ;CLEAR FLAG
8742 044542 000137 045576          JMP      I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
8743
8744 044546 032737 010400 001412 6$:    BIT      #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
8745                                ; UNIT FIELD ERROR
8746 044554 001415                    BEQ      7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
8747 044556 013704 001412          MOV      T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
8748 044562 042704 177770          BIC      #*C<DRVMSK>,R4 ;KEEP DRIVE BITS
8749 044566 010403                    MOV      R4,R3          ;STORE DRIVE NUMBER FOR INDEX
8750 044570 006303                    ASL      R3              ;MULTIPLY BY 2
8751 044572 016305 001524          MOV      PBLKT(R3),R5    ;STORE PARAMETER BLOCK ADDRESS
8752 044576 016237 000000 001410  MOV      RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
8753 044604 000137 044760          JMP      I.ERRC          ;REPORT ERROR
8754
8755 044610          7$:
8756
8757 044610 013705 001464          I.I00: MOV      O.WAIT,R5    ;LOAD PARAMETER BLOCK ADDRESS INTO R5
8758 044614 001002                    BNE      2$              ;IS COMMAND WAITING PROCESSING
8759                                ; YES, DO PROCESSING
8760 044616 000137 045576          JMP      I.ATTN          ;NO, PROCESS ATTENTION
8761
8762 044622 013704 001412          2$:    MOV      T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
8763 044626 042704 177770          BIC      #*C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
8764
8765
8766 044632 010403                    MOV      R4,R3          ;STORE DRIVE NUMBER FOR INDEX
8767 044634 006303                    ASL      R3              ;MULTIPLY BY 2 FOR INDEX
8768
8769 044636 126504 000000          CMPB     P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
8770 044642 001401                    BEQ      3$              ;YES, CONTINUE
8771 044644 000000                    HALT                                ;NO, DRIVER ERROR
8772 044646 122765 000164 000001 3$:    CMPB     #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
8773 044654 001002                    BNE      10$             ;NO, EXECUTE NORMAL DATA TRANSFER
8774 044656 000137 045236          JMP      I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
8775
8776 044662 005037 001464          10$:   CLR      O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
8777 044666 005063 001544          CLR      W.DRV(R3)       ;CLEAR WATCH-DOG TIME
8778 044672 146437 001514 001512  BICB     INTMSK(R4),W.TIME ;RESET TIMING ON THIS DRIVE
8779 044700 016237 000000 001410  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
8780 044706 032737 100000 001410  BIT      #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
8781 044714 001021                    BNE      I.ERRC          ;YES, PROCESS ERROR
8782 044716 016237 000016 001424  MOV      RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
8783 044724 136437 001514 001425  BITB     INTMSK(R4),T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
8784 044732 001004                    BNE      15$             ;YES, REPORT ERROR
8785 044734 004777 134514          JSR      PC,@A.NORM      ;INDICATE NORMAL RETURN
8786 044740 000137 047052          JMP      I.RTRN          ;RESTORE REGISTERS
8787
8788 044744 052765 000010 000014 15$:   BIS      #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION

```

```

8789
8790 044752 004737 047542      I.ERRA: JSR   PC,I.CSTS      ;STORE CONTROLLER STATUS
8791 044756 000405              BR      I.ERR              ;STORE PATTERN AND POSITION INFORMATION
8792
8793 044760 013765 001410 000016 I.ERRC: MOV   T.CS1,P.CS1(R5) ;GET ERROR RKCS1
8794 044766 004737 047564              JSR   PC,I.CS1            ;GET REST OF CONTROLLER STATUS
8795 044772 016265 000032 000062 I.ERR:  MOV   RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
8796 045000 016265 000030 000060      MOV   RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
8797 045006 004037 047070              JSR   RO,I.CCLR          ;CLEAR CONTROLLER
8798 045012 047052              I.RTRN                    ;ERROR RETURN
8799 045014 032765 010400 000020      BIT   #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
8800                                     ; UNIT FIELD ERROR
8801 045022 001056              BNE   5$                  ;YES, REPORT ERROR
8802 045024 032765 000001 000036      BIT   #DRA,P.DS(R5)      ;SEE IF DRIVE IS AVAIL
8803 045032 001004              BNE   4$                  ;BR IF YES
8804 045034 052765 010000 000014      BIS   #DRVSZD,P.PRST(R5) ;SET "DRIVE SIEZED" BIT
8805 045042 000471              BR    10$                 ;REPORT ERROR
8806 045044 004037 047646      4$:   JSR   RO,I.STAT        ;GATHER DRIVE STATUS
8807 045050 047052              I.RTRN                    ;ERROR RETURN
8808 045052 112737 000005 001410      MOVB  #DR.CLR,T.CS1      ;LOAD COMMAND
8809 045060 004037 047152              JSR   RO,I.ISSU          ;ISSUE DRIVE CLEAR
8810 045064 047052              I.RTRN                    ;ERROR RETURN
8811 045066 136437 001514 001425      BITB  INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION RESET
8812 045074 001407              BEQ   2$                  ;NO, INDICATE DRIVE ERROR
8813 045076 052737 000020 001462      BIS   #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
8814                                     ; WITH CLEAR
8815 045104 004777 134350              JSR   PC,@A.CONT        ;REPORT CONTROLLER ERROR
8816 045110 000137 047052              JMP   I.RTRN            ;GO RESTORE REGISTERS
8817
8818 045114 032737 040000 001434 2$:   BIT   #S.DSC,T.MR2       ;CHECK IF DRIVE STATUS CHANGE CLEARED
8819 045122 001403              BEQ   3$                  ;YES, CHECK FAULT
8820 045124 052765 000040 000014      BIS   #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
8821 045132 032737 001000 001436 3$:   BIT   #S.PAR,T.MR3       ;CHECK IF DRIVE PARITY ERROR
8822 045140 001407              BEQ   5$                  ;NO, INDICATE ABNORMAL TERMINATION
8823 045142 052737 002000 001462      BIS   #E.DPAR,E.CONT     ;SET DRIVE PARITY ERROR
8824 045150 004777 134304              JSR   PC,@A.CONT        ;INDICATE CONTROLLER ERROR
8825 045154 000137 047052              JMP   I.RTRN            ;RETURN
8826
8827 045160 032765 000020 000014 5$:   BIT   #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
8828 045166 001017              BNE   10$                 ;YES, GO REPORT ERROR
8829 045170 032737 020000 001434      BIT   #S.PIP,T.MR2       ;CHECK IF DRIVE IS CYCLING DOWN
8830 045176 001413              BEQ   10$                 ;NO, REPORT ERROR
8831 045200 052765 020000 000014      BIS   #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
8832 045206 156437 001514 001512      BISB  INTMSK(R4),W.TIME ;SET UP 8 SECONDS FOR
8833 045214 013763 001474 001544      MOV   W.8SEC,W.DRV(R3)   ;DRIVE TO CYCLE UP
8834 045222 000137 047052              JMP   I.RTRN            ;GO RESTORE REGISTERS
8835
8836 045226 004777 134224      10$:  JSR   PC,@A.ABNL        ;GO REPORT ERROR
8837 045232 000137 047052              JMP   I.RTRN            ;GO RESTORE REGISTERS
8838
8839                                     .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
8840
8841 045236 016237 000000 001410 I.HDAL: MOV   RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
8842                                     ; ERROR
8843 045244 032737 100000 001410      BIT   #CERR,T.CS1       ;CHECK IF CONTROLLER ERROR
8844 045252 001423              BEQ   5$                  ;NO, CHECK FOR ATTENTION

```

```

8845
8846 045254 005037 001464          CLR      O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETE
8847 045260 146437 001514 001512  BICB    INTMSK(R4),W.TIME ;RESET TIMING ON DRIVE
8848 045266 005063 001544          CLR      W.DRV(R3)      ;CLEAR TIME OUT COUNT
8849 045272 013765 001410 000016  MOV     T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
8850 045300 004737 047564          JSR     PC,I.CS1        ;STORE CONTROLLER REGISTERS
8851 045304 004037 047070          JSR     RO,I.CCLR       ;CLEAR CONTROLLER
8852 045310 047052          I.RTRN  ;ERROR RETURN
8853 045312 004777 134140          JSR     PC,QA.ABNL      ;INDICATE ERROR RETURN
8854 045316 000137 047052          JMP     I.RTRN         ;RESTORE REGISTERS
8855
8856 045322 016237 000016 001424 5$:  MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
8857 045330 136437 001514 001425  BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
8858 045336 001411          BEQ     7$             ;NO, CHECK IF READ ALL HEADERS
8859 045340 005037 001464          CLR      O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETION
8860 045344 146437 001514 001512  BICB    INTMSK(R4),W.TIME ;RESET TIMING ON DRIVE
8861 045352 005063 001544          CLR      W.DRV(R3)      ;CLEAR TIME OUT COUNT
8862 045356 000137 044752          JMP     I.ERRA         ;GO REPORT ERROR
8863
8864 045362 013701 001502          7$:  MOV     HDR.AD,R1        ;GET MAIN MEMORY ADDRESS
8865 045366 016221 000024          MOV     RKDB(R2),(R1)+  ;GET FIRST WORD OF HEADER
8866 045372 016221 000024          MOV     RKDB(R2),(R1)+  ;GET SECOND WORD OF HEADER
8867 045376 016221 000024          MOV     RKDB(R2),(R1)+  ;GET THIRD WORD OF HEADER
8868 045402 010137 001502          MOV     R1,HDR.AD       ;STORE ADDRESS FOR NEXT HEADER
8869 045406 016237 000010 001412  MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
8870 045414 032737 100000 001412  BIT     #DLT,T.CS2      ;CHECK FOR DATA LATE
8871 045422 001056          BNE     35$           ;YES, REPORT ERROR
8872 045424 005337 001504          DEC     HDR.CT         ;DECREMENT NUMBER OF HEADER YET TO READ
8873 045430 001027          BNE     25$           ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
8874 045432 005037 001464          CLR      O.WAIT          ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
8875 045436 005063 001544          CLR      W.DRV(R3)      ;CLEAR TIME OUT COUNT FOR THIS DRIVE
8876 045442 146437 001514 001512  BICB    INTMSK(R4),W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
8877 045450 012762 000003 000026  MOV     #3,RKMR1(R2)    ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
8878 045456 112737 000001 001410  MOVB    #DR.SEL,T.CS1   ;LOAD SELECT COMMAND
8879 045464 004037 047152          JSR     RO,I.ISSU       ;GET SECTOR COUNT
8880 045470 047052          I.RTRN  ;ERROR RETURN
8881 045472 013765 001436 000056  MOV     T.MR3,P.B11(R5) ;LOAD SECTOR COUNT
8882 045500 004777 133750          JSR     PC,QA.NORM      ;INDICATE NORMAL TERMINATION
8883 045504 000137 047052          JMP     I.RTRN         ;RESTORE REGISTERS
8884
8885 045510 016562 000002 000020 25$:  MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
8886 045516 016562 000004 000006  MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
8887 045524 116565 000007 000017  MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
8888 045532 042765 165777 000016  BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
8889                                     ; DRIVE TYPE
8890 045540 112765 000125 000016  MOVB    #RDHEAD,P.CS1(R5) ;STORE COMMAND ISSUED
8891 045546 016562 000016 000000  MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
8892 045554 000137 047052          JMP     I.RTRN         ;RESTORE REGISTERS
8893
8894 045560 052737 000400 001462 35$:  BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
8895 045566 004777 133666          JSR     PC,QA.CONT      ;REPORT ERROR
8896 045572 000137 047052          JMP     I.RTRN         ;RESTORE REGISTERS
8897
8898                                     .SBTTL  *DRIVE ATTENTION SCANNER
8899
8900 045576 016237 000000 001410  I.ATTN: MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS

```

```

8901                                     : REGISTER 1 FOR COMPARISON
8902 045604 032737 100000 001410      BIT   #CERR,T.CS1      :CHECK IF CONTROLLER ERROR OCCURRED
8903 045612 001444                                     BEQ   5$              :NO, CHECK IF ATTENTION
8904
8905                                     :CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
8906 045614 032737 164000 001412      BIT   #DLT!WCE!UPE!NEM,T.CS2
8907
8908 045622 001007      BNE   1$              :INDICATE ERROR
8909 045624 016237 000014 001426      MOV   RKER(R2),T.ER  :STORE ERROR REGISTER
8910
8911                                     :
8912 045632 032737 125700 001426      BIT   #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
8913
8914 045640 001407      BEQ   2$              :NO DATA TRANSFER ERROR
8915
8916 045642 052737 000010 001462 1$:   BIS   #E.UDAT,E.CONT :SET UNEXPECTED DATA TYPE ERROR
8917 045650 004777 133604             JSR   PC,@A.CONT     :REPORT ERROR
8918 045654 000137 047052             JMP   I.RTRN         :RESTORE REGISTERS
8919
8920 045660 013704 001412             2$:   MOV   T.CS2,R4      :SAVE CS2 FOR REGISTER NUMBER
8921 045664 042704 177770             BIC   #^C<DRVMSK>,R4 :STRIP OFF JUNK
8922 045670 010403             MOV   R4,R3         :STORE DRIVE NUMBER IN R3
8923 045672 006303             ASL   R3             :MULTIPLY DRIVE NUMBER BY 2
8924 045674 146437 001514 001512      BICB  INTMSK(R4),W.TIME :CLEAR WATCH-DOG TIMER FOR DRIVE
8925 045702 005063 001544             CLR   W.DRV(R3)     :RESET WATCH DOG TIME
8926 045706 016305 001524             MOV   PBLKT(R3),R5  :STORE PARAMETER BLOCK ADDRESS IN R5
8927
8928                                     :
8929                                     :
8930 045712 042765 000006 000014      BIC   #DRVPOS!DRVPDT,P.PRST(R5)
8931
8932 045720 000137 044760             JMP   I.ERRC        :GO REPORT ERROR
8933
8934 045724 032737 040000 001410 5$:   BIT   #DI,T.CS1     :CHECK IF ANY DRIVE ATTENTION
8935 045732 001004             BNE   6$            :YES, PROCESS INTERRUPT
8936 045734 004737 052116             JSR   PC,C.OPT      :CALL COMMAND OPTIMIZER
8937 045740 000137 047052             JMP   I.RTRN         :RESTORE REGISTERS
8938
8939 045744 016237 000016 001424 6$:   MOV   RKASOF(R2),T.ASOF :STORE ATTENTION SUMMARY
8940 045752 105737 001425             TSTB  T.ASOF+1      :CHECK IF ANY ATTENTIONS SET
8941 045756 001007             BNE   7$            :YES GO PROCESS INTERRUPT
8942 045760 052737 000002 001462      BIS   #E.NOAT,E.CONT :SET NO ATTENTION IN ATTENTION SUMMARY
8943 045766 004777 133466             JSR   PC,@A.CONT     :GO REPORT ERROR
8944 045772 000137 047052             JMP   I.RTRN         :GO RESTORE REGISTERS
8945
8946 045776 005004             7$:   CLR   R4            :CLEAR DRIVE NUMBER
8947 046000 136437 001514 001425 8$:   BITB  INTMSK(R4),T.ASOF+1 :CHECK IF ATTENTION OF THIS DRIVE
8948 046006 001002             BNE   9$            :YES, PROCESS INTERRUPT
8949 046010 005204             INC   R4            :INCREMENT DRIVE NUMBER
8950 046012 000772             BR    8$            :CHECK ATTENTION ON NEXT DRIVE
8951
8952 046014 010403             9$:   MOV   R4,R3         :STORE DRIVE NUMBER
8953 046016 006303             ASL   R3            :MULTIPLY DRIVE NUMBER BY 2
8954 046020 016305 001524             MOV   PBLKT(R3),R5  :STORE PARAMETER BLOCK ADDRESS
8955 046024 032765 010000 000014      BIT   #DRVSZD,P.PRST(R5) :CHECK IF DRIVE WAS SEIZED
8956 046032 001402             BEQ   10$           :NO, PROCESS NORMAL ATTENTION

```

```

8957 046034 000137 046664          JMP      I.DUAL          ;PROCESS THE RELEASE OF DRIVE
8958
8959 046040          10$:
8960 046040 032765 020000 000014          BIT      #E.UNLD,P.PRST(R5) ;CHECK IF DRIVE UNLOADING
8961 046046 001402          BEQ     11$              ;NO, CONTINUE
8962 046050 000137 046560          JMP      I.UNLD          ;SERVICE DRIVE IN POSITION AFTER ERROR
8963
8964 046054 042765 000002 000014 11$:          BIC     #DRVPOS,P.PRST(R5) ;RESET DRIVE POSITIONING
8965 046062 005062 000026          CLR     RKMR1(R2)        ;CLEAR MAINTENANCE REGISTER 1
8966 046066 112737 000001 001410          MOVB   #DR.SEL,T.CS1     ;LOAD COMMAND
8967 046074 004037 047152          JSR     RO,I.ISSU        ;SELECT DRIVE WITH ATTENTION HIGH
8968 046100 047052          I.RTRN                    ;ERROR RETURN
8969 046102 013765 001436 000042          MOV     T.MR3,P.B00(R5)  ;STORE STATUS BYTE 00 MESS B
8970 046110 032765 000200 000042          BIT     #S.FLT,P.B00(R5) ;CHECK IF DRIVE FAULT
8971 046116 001401          BEQ     12$              ;NO, CHECK FOR DRIVE STATUS CHANGE
8972 046120 000473          BR     I.AERR            ;PROCESS ERROR
8973
8974 046122 013765 001434 000040 12$:          MOV     T.MR2,P.A00(R5)  ;STORE MAINTENANCE REGISTER 2
8975 046130 032765 040000 000040          BIT     #S.DSC,P.A00(R5) ;CHECK FOR DRIVE STATUS CHANGE
8976 046136 001004          BNE     13$              ;YES, PROCESS DRIVE STATUS CHANGE
8977 046140 052765 004000 000014          BIS     #NODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
8978 046146 000460          BR     I.AERR            ;PROCESS ERROR
8979
8980 046150 112737 000005 001410 13$:          MOVB   #DR.CLR,T.CS1     ;LOAD COMMAND
8981 046156 004037 047152          JSR     RO,I.ISSU        ;CLEAR DRIVE STATUS CHANGE
8982 046162 047052          I.RTRN                    ;ERROR RETURN
8983 046164 013765 001424 000032          MOV     T.ASOF,P.ASOF(R5) ;STORE ATTENTION SUMMARY
8984 046172 136465 001514 000033          BITB   INTMSK(R4),P.ASOF+1(R5) ;CHECK IF ATTENTION RESET
8985 046200 001407          BEQ     15$              ;YES, CONTINUE INTERRUPT PROCESSING
8986 046202 052737 000020 001462          BIS     #E.CLAT,E.CONT   ;SET ATTENTION DID NOT RESET
8987                                ; WITH DRIVE CLEAR
8988 046210 004777 133244          JSR     PC,@A.CONT       ;FLAG ERROR
8989 046214 000137 047052          JMP     I.RTRN           ;RESTORE REGISTERS
8990
8991 046220 013765 001434 000040 15$:          MOV     T.MR2,P.A00(R5)  ;STORE MAINTENANCE REGISTER 2
8992 046226 032765 040000 000040          BIT     #S.DSC,P.A00(R5) ;CHECK IF DRIVE STATUS CHANGE
8993                                ; RESET
8994 046234 001404          BEQ     16$              ;YES, CONTINUE INTERRUPT PROCESSING
8995 046236 052765 000040 000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
8996 046244 000421          BR     I.AERR            ;GO PROCESS ERROR
8997
8998 046246 146437 001514 001512 16$:          BICB   INTMSK(R4),W.TIME ;RESET TIMING ON THIS DRIVE
8999 046254 005063 001544          CLR     W.DRV(R3)        ;CLEAR DRIVE TIMING COUNT
9000 046260 032765 000004 000014          BIT     #DRVPDT,P.PRST(R5) ;CHECK IF DRIVE IS POSITIONING
9001                                ; FOR DATA TRANSFER
9002 046266 001004          BNE     17$              ;YES, CALL COMMAND OPTIMIZER
9003 046270 004777 133160          JSR     PC,@A.NORM       ;REPORT SUCCESSFUL COMMAND
9004                                ; COMPLETION
9005 046274 000137 047052          JMP     I.RTRN           ;RESTORE REGISTERS
9006
9007 046300 004737 052116 17$:          JSR     PC,C.OPT         ;CALL COMMAND OPTIMIZER
9008 046304 000137 047052          JMP     I.RTRN           ;RESTORE REGISTERS
9009
9010                                .SBTTL *ATTENTION ERROR HANDLER
9011
9012 046310 042765 000004 000014 I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE

```

```
9013                                     : OF DATA TRANSFER
9014 046316 146437 001514 001512 BICB INTMSK(R4),W.TIME :CLEAR TIMING FOR THIS DRIVE
9015 046324 005063 001544 CLR W.DRV(R3) :RESET WATCH-DOG TIME
9016 046330 042765 177741 000016 BIC #177741,P.CS1(R5) :KEEP COMMAND ISSUED
9017 046336 042737 000036 001410 BIC #36,T.CS1 :KEEP CURRENT CONTROLLER STATUS
9018 046344 053765 001410 000016 BIS T.CS1,P.CS1(R5) :MAKE GOOD MESSAGE
9019 046352 013765 001412 000020 MOV T.CS2,P.CS2(R5) :STORE CONTROLLER REGISTERS
9020 046360 013765 001414 000022 MOV T.WCR,P.WCR(R5)
9021 046366 013765 001416 000024 MOV T.BA,P.BAR(R5)
9022 046374 013765 001420 000026 MOV T.DA,P.DTS(R5)
9023 046402 013765 001422 000030 MOV T.DC,P.DCYL(R5)
9024 046410 013765 001424 000032 MOV T.ASOF,P.ASOF(R5)
9025 046416 013765 001426 000034 MOV T.ER,P.ER(R5)
9026 046424 013765 001430 000036 MOV T.DS,P.DS(R5)
9027 046432 004037 047646 JSR RO,I.STAT :GATHER DRIVE STATUS
9028 046436 047052 I.RTRN :ERROR RETURN
9029 046440 112737 000005 001410 MOVB #DR.CLR,T.CS1 :LOAD COMMAND
9030 046446 004037 047152 JSR RO,I.ISSU :CLEAR DRIVE ERRORS
9031 046452 047052 I.RTRN :ERROR RETURN
9032 046454 136437 001514 001425 BITB INTMSK(R4),T.ASOF+1 :CHECK IF ATTENTION RESET
9033 046462 001407 BEQ 2$ :YES, FLAG DRIVE ERROR
9034 046464 052737 000020 001462 BIS #E.CLAT,E.CONT :SET ATTENTION DID NOT RESET
9035 046472 004777 132762 JSR PC,@A.CONT :REPORT ERROR
9036 046476 000137 047052 JMP I.RTRN :RESTORE REGISTERS
9037
9038 046502 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) :CHECK IF AWARD DRIVE ERROR
9039 046510 001017 BNE 10$ :YES, REPORT ERROR
9040 046512 032737 020000 001434 BIT #S.PIP,T.MR2 :CHECK IF DRIVE IS UNLOADING
9041 046520 001413 BEQ 10$ :NO, REPORT ERROR
9042 046522 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) :SET DRIVE UNLOADING DUE TO ERROR
9043 046530 156437 001514 001512 BISB INTMSK(R4),W.TIME :SET TIMING ON THIS DRIVE
9044 046536 013763 001474 001544 MOV W.8SEC,W.DRV(R3) :LOAD 8 SECONDS FOR CYCLE UP
9045 046544 000137 047052 JMP I.RTRN :RESTORE REGISTERS
9046
9047 046550 004777 132702 10$: JSR PC,@A.ABNL :REPORT ERROR
9048 046554 000137 047052 JMP I.RTRN :RESTORE REGISTERS
9049
9050 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
9051
9052 046560 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) :CLEAR DRIVE UNLOADING BECAUSE OF ERROR
9053 046566 112737 000005 001410 MOVB #DR.CLR,T.CS1 :LOAD IN DRIVE CLEAR
9054 046574 004037 047152 JSR RO,I.ISSU :GO ISSUE DRIVE CLEAR
9055 046600 047052 I.RTRN :ERROR RETURN
9056 046602 136437 001514 001425 BITB INTMSK(R4),T.ASOF+1 :CHECK IF ATTENTION CLEARED
9057 046610 001406 BEQ 15$ :YES, CONTINUE
9058 046612 012737 000020 001462 MOV #E.CLAT,E.CONT :SET ATTENTION DID NOT RESET
9059 046620 004777 132634 JSR PC,@A.CONT :REPORT ERROR
9060 046624 000512 BR I.RTRN :RESTORE REGISTERS
9061
9062 046626 032737 040000 001434 15$: BIT #S.DSC,T.MR2 :CHECK IF DRIVE STAUUS CHANGE RESET
9063 046634 001403 BEQ 20$ :YES, CONTINUE
9064 046636 052765 000040 000014 BIS #DRVDSC,P.PRST(R5) :SET DRIVE STAUUS CHANGE DID NOT CLEAR
9065 046644 146437 001514 001512 20$: BICB INTMSK(R4),W.TIME :RESET TIMING ON THIS DRIVE
9066 046652 005063 001544 CLR W.DRV(R3) :CLEAR TIME COUNT
9067 046656 004777 132574 JSR PC,@A.ABNL :REPORT ERROR
9068 046662 000473 BR I.RTRN :RESTORE REGISTER
```

```

9069
9070      .SBTTL      *DUAL ACCESS INTERRUPT HANDLER
9071
9072 046664 112737 000001 001410 I.DUAL: MOVB   #DR.SEL,T.CS1   ;LOAD COMMAND
9073 046672 004037 047152          JSR    RO,I.ISSU   ;SELECT DRIVE
9074 046676 047052          I.RTRN          ;ERROR RETURN
9075 046700 032737 000200 001436 BIT    #S.FLT,T.MR3   ;CHECK IF DRIVE FAULT
9076 046706 001402          BEQ    10$        ;NO, PROCESS INTERRUPT
9077 046710 000137 046310          JMP    I.AERR     ;PROCESS ATTENTION ERROR
9078
9079 046714 032737 040000 001434 10$:  BIT    #S.DSC,T.MR2   ;CHECK IF DRIVE STATUS CHANGE SET
9080 046722 001005          BNE    11$        ;YES, PROCESS INTERRUPT
9081 046724 052765 004000 000014 BIS    #NODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
9082 046732 000137 046310          JMP    I.AERR     ;PROCESS ATTENTION ERROR
9083
9084 046736 042765 010000 000014 11$:  BIC    #DRVSZD,P.PRST(R5) ;RESET DRIVE SEIZED
9085 046744 112737 000005 001410 MOVB   #DR.CLR,T.CS1   ;LOAD COMMAND
9086 046752 004037 047152          JSR    RO,I.ISSU   ;CLEAR DRIVE DSC
9087 046756 047052          I.RTRN          ;ERROR RETURN
9088 046760 136437 001514 001425 BITB   INTMSK(R4),T.ASO+1 ;CHECK IF ATTENTION RESET
9089 046766 001406          BEQ    15$        ;YES, CONTINUE
9090 046770 052737 000020 001462 BIS    #E.CLAT,E.CONT   ;SET ATTENTION DID NOT RESET
9091 046776 004777 132456          JSR    PC,@A.CONT   ;INDICATE CONTROLLER ERROR
9092 047002 000423          BR     I.RTRN     ;RESTORE REGISTERS
9093
9094 047004 032737 040000 001434 15$:  BIT    #S.DSC,T.MR2   ;CHECK IF DRIVE STATUS CHANGE IS STILL SET
9095 047012 001405          BEQ    20$        ;NO, GO ENQUEUE COMMAND
9096 047014 052765 000040 000014 BIS    #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
9097 047022 000137 046310          JMP    I.AERR     ;REPORT ATTENTION ERROR
9098
9099 047026 146437 001514 001512 20$:  BICB   INTMSK(R4),W.TIME ;STOP TIMING ON THIS DRIVE
9100 047034 005063 001544          CLR    W.DRV(R3)   ;CLEAR WATCH DOG TIME OF THIS DRIVE
9101 047040 004037 051564          JSR    RO,Q.PUSH   ;PUT PARAMETER BLOCK ADDRESS ON
9102 047044 001324          CINITQ          ;COMMAND INITATION QUEUE
9103 047046 004737 052116          JSR    PC,C.OPT   ;CALL COMMAND OPTIMIZER
9104
9105      I.RTRN: MOV    (SP)+,R0   ;RESTORE R0
9106      MOV    (SP)+,R1   ;RESTORE R1
9107      MOV    (SP)+,R2   ;RESTORE R2
9108      MOV    (SP)+,R3   ;RESTORE R3
9109      MOV    (SP)+,R4   ;RESTORE R4
9110      MOV    (SP)+,R5   ;RESTORE R5
9111      RTI              ;RETURN
9112

```

9113
9114
9115
9116
9117
9118
9119
9120
9121
9122
9123
9124
9125
9126
9127
9128
9129
9130
9131
9132
9133
9134
9135
9136
9137
9138
9139
9140
9141
9142
9143
9144
9145
9146
9147

.SBTTL *CONTROLLER CLEAR ROUTINE

```

:*****
:*
:* THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
:* AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT
:* CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
:* E.CCLR SET IN E.CONT.
:*
:* REGISTER      USE
:* -----      ---
:*
:* R2            ADDRESS OF RK06 REGISTERS
:* R5            ADDRESS OF PARAMETER BLOCK
:*
:*CALL JSR      R0,I.CCLR
:*      <ADDRESS OF ERROR RETURN>
:*      RETURN
:*
:*****

```

```

9134 047070 012762 100000 000000 I.CCLR: MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
9135 047076 016237 000000 001410      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
9136 047104 032737 100000 001410      BIT      #CERR,T.CS1 ;CHECK IF CONTROLLER CLEAR DID
9137                                     ; CLEAR ERROR
9138 047112 001407                                     BEQ      $$ ;YES, RETURN TO DRIVER PROCESSING
9139 047114 052737 000001 001462      BIS      #E.CCLR,E.CONT ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
9140 047122 004777 132332      JSR      PC,@A.CONT ;REPORT CONTROLLER ERROR
9141 047126 011000      MOV      (R0),R0 ;SET UP ERROR RETURN
9142 047130 000200      RTS      R0 ;RETURN
9143
9144 047132 012762 000100 000000 $$:  MOV      #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
9145 047140 112737 177777 001506      MOVB    #-1,I.ISRL ;SET INTERRUPT ENABLE ISSUED
9146 047146 005720      TST     (R0)+ ;ADJUST FOR NORMAL RETURN
9147 047150 000200      RTS     R0 ;RETURN

```


9148
9149
9150
9151
9152
9153
9154
9155
9156
9157
9158
9159
9160
9161
9162
9163
9164
9165
9166
9167
9168
9169
9170
9171
9172
9173
9174
9175
9176
9177
9178
9179
9180
9181
9182
9183
9184
9185
9186
9187
9188
9189
9190
9191
9192
9193
9194
9195
9196
9197
9198
9199
9200
9201
9202
9203

.SBTTL *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

```

*****
*
* THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
* AND CHECK IF A CONTROLLER ERROR OCCURRED. IF A CONTROLLER
* ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
* CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
* ADDRESS IN A.CONT.
*
* REGISTER      USE
* -----      ---
*
* R2            ADDRESS OF RK06 REGISTERS
* R5            ADDRESS OF PARAMETER BLOCK
*
*CALL JSR      RO,I.ISSU
*      <ADDRESS OF ERROR RETURN>
*      RETURN
*
* ROUTINES USED:
* -----
*
*           I.CCLR
*           I.STOR
*****

```

```

I.ISSU: MOV      T.CS1,-(SP)      ;STORE COMMAND ISSUED
        CLR      T.CS2          ;CLEAR TEMPORARY CS2
        MOVB     P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
        MOV      T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
        MOVB     P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
        BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
        ;      FORMAT AND DRIVE TYPE
1$:     MOV      T.CS1,RKCS1(R2) ;ISSUE COMMAND
        TSTB    RKCS1(R2)      ;WAIT FOR READY
        BPL     1$
        JSR     PC,I.STOR      ;GO STORE REGISTERS
        BIT     #CERR,T.CS1    ;CHECK IF CONTROLLER ERROR OCCURED
        BEQ     5$            ;NO, RETURN
        BIT     #MDS,T.CS2     ;CHECK IF MULTIPLE DRIVE SELECT
        BEQ     2$            ;NO, CHECK FOR OTHER CONTROLLER ERRORS
        BIS     #E.MDS,E.CONT  ;SET MULTIPLE DRIVE SELECT FLAG
        JSR     PC,@A.CONT    ;REPORT CONTROLLER ERROR
        BR      10$          ;RETURN

;CHECK IF ANY CONTROLLER ERROR IS SET
2$:     CMP     #1,OPTFLG      ;SEE IF CURRENTLY IN CMND OPTIMIZER
        BNE     4$            ;BR IF NOT
        BIT     #NED,T.CS2    ;SEE IF NED ERROR SET
        BNE     3$            ;BR IF YES, TO ISSUE CLEAR AND EXIT
4$:     BIT     #CTO!SPAR,T.CS1
        BNE     7$
        BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
        BNE     7$

```

```

047152 013746 001410
047156 005037 001412
047162 116537 000000 001412
047170 013762 001412 000010
047176 116537 000007 001411
047204 142737 177753 001411
047212 013762 001410 000000
047220 105762 000000
047224 100375
047226 004737 047414
047232 032737 100000 001410
047240 001447
047242 032737 001000 001412
047250 001406
047252 052737 100000 001462
047260 004777 132174
047264 000450
047266 022737 000001 001500 2$:
047274 001004
047276 032737 010000 001412
047304 001022
047306 032737 024000 001410 4$:
047314 001027
047316 032737 176400 001412
047324 001023

```

9204	047326	032737	131761	001426		BIT	#ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
9205	047334	001017				BNE	7\$
9206							
9207	047336	122716	000005			CMPB	#DR.CLR,(SP) ;CHECK IF CLEAR DRIVE
9208	047342	001003				BNE	3\$;NO, DO NOT SET DRIVE HARD ERROR
9209	047344	052765	000020	000014		BIS	#DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
9210	047352	004037	047070		3\$:	JSR	RO,I.CCLR ;GO ISSUE A CONTROLLER CLEAR
9211	047356	047406				10\$;ERROR RETURN
9212	047360	012762	000100	000000	5\$:	MOV	#IE,RKCS1(R2) ;SET INTERRUPT ENABLE
9213	047366	005726				TST	(SP)+ ;ADJUST STACK
9214	047370	005720				TST	(RO)+ ;ADJUST RO FOR NORMAL RETURN
9215	047372	000200				RTS	RO ;RETURN
9216							
9217	047374	052737	001000	001462	7\$:	BIS	#E.CERR,E.CONT ;SET CONTROLLER ERROR DURING
9218							; DRIVER SERVICING
9219	047402	004777	132052			JSR	PC,@A.CONT ;REPORT ERROR
9220	047406	005726			10\$:	TST	(SP)+ ;ADJUST STACK
9221	047410	011000				MOV	(RO),RO ;ADJUST RO FOR ERROR RETURN
9222	047412	000200				RTS	RO ;RETURN

9223
9224
9225
9226
9227
9228
9229
9230
9231
9232
9233
9234
9235
9236
9237
9238
9239
9240 047414 016237 000000 001410
9241 047422 016237 000010 001412
9242 047430 016237 000002 001414
9243 047436 016237 000004 001416
9244 047444 016237 000006 001420
9245 047452 016237 000012 001430
9246 047460 016237 000014 001426
9247 047466 016237 000016 001424
9248 047474 016237 000020 001422
9249 047502 016237 000026 001432
9250 047510 016237 000034 001434
9251 047516 016237 000036 001436
9252 047524 016237 000030 001440
9253 047532 016237 000032 001442
9254 047540 000207

.SBTTL *STORE RK611 UNIBUS REGISTERS

```
*****  
*  
* THIS SUBROUTINE IS CALLED BY THE RK06/07 DRIVER TO STORE ALL  
* RK611 REGISTER IN TEMPORARY LOCATIONS.  
*  
*CALL JSR PC,I.STOR  
* RETURN  
*  
* REGISTER USE  
* -----  
*  
* R2 ADDRESS OF RK611 REGISTERS  
*  
*****
```

```
I.STOR: MOV RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS  
MOV RKCS2(R2),T.CS2 ; EXCEPT DATA BUFFER  
MOV RKWC(R2),T.WCR  
MOV RKBA(R2),T.BA  
MOV RKDA(R2),T.DA  
MOV RKDS(R2),T.DS  
MOV RKER(R2),T.ER  
MOV RKASOF(R2),T.ASOF  
MOV RKDCYL(R2),T.DC  
MOV RKMR1(R2),T.MR1  
MOV RKMR2(R2),T.MR2  
MOV RKMR3(R2),T.MR3  
MOV RKECPS(R2),T.POS  
MOV RKECPT(R2),T.PAT  
RTS PC ;RETURN
```

.SBTTL *STORE CONTROLLER STATUS

* THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
* THE FOLLOWING REGISTERS WILL BE STORED:

- COMMAND AND STATUS REGISTER 2
- WORD COUNT REGISTER
- BUS ADDRESS REGISTER
- DESIRED TRACK AND SECTOR
- STATUS REGISTER
- ERROR REGISTER
- ATTENTION SUMMARY/OFFSET REGISTER
- CYLINDER ADDRESS REGISTER

*CALL JSR PC,I.CSTS
* RETURN

THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:

REGISTER	CONTENTS
R2	RK06/07 BASE ADDRESS
R5	ADDRESS OF PARAMETER BLOCK

```
9255  
9256  
9257  
9258  
9259  
9260  
9261  
9262  
9263  
9264  
9265  
9266  
9267  
9268  
9269  
9270  
9271  
9272  
9273  
9274  
9275  
9276  
9277  
9278  
9279  
9280  
9281  
9282  
9283  
9284  
9285 047542 042765 177741 000016 I.CSTS: BIC #177741,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FUNCTION  
9286 ;OF LAST COMMAND ISSUED  
9287 047550 042737 000036 001410 BIC #36,T.CS1 ;CLEAR FUNCTION OF CS1 STATUS  
9288 047556 053765 001410 000016 BIS T.CS1,P.CS1(R5) ;GENERATE CS1 STATUS INFORMATION  
9289 047564 016265 000010 000020 I.CST1: MOV RKCS2(R2),P.CS2(R5) ;STORE COMMAND AND STATUS REGISTER 2  
9290 047572 016265 000002 000022 MOV RKWC(R2),P.WCR(R5) ;STORE WORD COUNT REGISTER  
9291 047600 016265 000004 000024 MOV RKBA(R2),P.BAR(R5) ;STORE BUS ADDRESS REGISTER  
9292 047606 016265 000006 000026 MOV RKDA(R2),P.DTS(R5) ;STORE DESIRED TRACK AND SECTOR  
9293 047614 016265 000012 000036 MOV RKDS(R2),P.DS(R5) ;STORE DRIVE STATUS REGISTER  
9294 047622 016265 000014 000034 MOV RKER(R2),P.ER(R5) ;STORE ERROR REGISTER  
9295 047630 016265 000016 000032 MOV RKASOF(R2),P.ASOF(R5) ;STORE ATTENTION SUMMARY AND  
9296 ; OFFSET  
9297 047636 016265 000020 000030 MOV RKDCYL(R2),P.DCYL(R5) ;STORE CYLINDER ADDRESS  
9298 047644 000207 RTS PC ;RETURN  
9299
```

9300
9301
9302
9303
9304
9305
9306
9307
9308
9309
9310
9311
9312
9313
9314
9315
9316
9317
9318
9319
9320
9321
9322
9323
9324
9325
9326
9327
9328
9329
9330
9331
9332
9333
9334
9335
9336
9337
9338
9339
9340
9341
9342
9343
9344
9345
9346
9347
9348
9349
9350
9351
9352
9353
9354
9355

.SBTTL *GATHER DRIVE STATUS

```

*****
*
* THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
* BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
* HAS PREVIOUSLY BEEN SEIZED. IT RUNS AT PRIORITY 7.
*
*CALL JSR R0,I.STAT
*      <ADDRESS OF ERROR RETURN>
*      RETURN
*
* THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
*
*          REGISTER          CONTENTS
*          -----          -
*          R2                RK06 BASE ADDRESS
*          R5                ADDRESS OF PARAMETER BLOCK
*
* ROUTINES USED:
*          I.ISSU
*****

```

```

9326 047646 012762 000001 000026 I.STAT: MOV #1,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
9327 ; FOR STATUS BYTE 01
9328 047654 112737 000001 001410 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
9329 047662 004037 047152 JSR R0,I.ISSU ;GET STATUS BYTES 01
9330 047666 050056 3$ ;ERROR RETURN
9331 047670 013765 001434 000044 MOV T.MR2,P.A01(R5) ;STORE STATUS BYTE 01 MESS A
9332 047676 013765 001436 000046 MOV T.MR3,P.B01(R5) ;STORE STATUS BYTE 01 MESS B
9333 047704 012762 000002 000026 MOV #2,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
9334 ; FOR STATUS BYTE 10
9335 047712 112737 000001 001410 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
9336 047720 004037 047152 JSR R0,I.ISSU ;GET STATUS BYTES 10
9337 047724 050056 3$ ;ERROR RETURN
9338 047726 013765 001434 000050 MOV T.MR2,P.A10(R5) ;STORE STATUS BYTE 10 MESS A
9339 047734 013765 001436 000052 MOV T.MR3,P.B10(R5) ;STORE STATUS BYTE 10 MESS B
9340 047742 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER
9341 ; FOR STATUS BYTE 11
9342 047750 112737 000001 001410 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
9343 047756 004037 047152 JSR R0,I.ISSU ;GET STATUS BYTES 11
9344 047762 050056 3$ ;ERROR RETURN
9345 047764 013765 001434 000054 MOV T.MR2,P.A11(R5) ;STORE STATUS BYTE 11 MESS A
9346 047772 013765 001436 000056 MOV T.MR3,P.B11(R5) ;STORE STATUS BYTE 11 MESS B
9347 050000 005062 000026 CLR RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
9348 ; FOR STATUS BYTE 00
9349 050004 112737 000001 001410 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
9350 050012 004037 047152 JSR R0,I.ISSU ;GET STATUS BYTES 00
9351 050016 050056 3$ ;ERROR RETURN
9352 050020 013765 001434 000040 MOV T.MR2,P.A00(R5) ;STORE STATUS BYTE 00 MESS A
9353 050026 013765 001436 000042 MOV T.MR3,P.B00(R5) ;STORE STATUS BYTE 00 MESS B
9354 050034 032737 001000 001436 BIT #S.PAR,T.MR3 ;CHECK IF BAD PARITY DETECTED BY DRIVE
9355 050042 001407 BEQ 5$ ;NO, RETURN NORMALLY

```


9365
9366
9367
9368
9369
9370
9371
9372
9373
9374
9375
9376
9377
9378
9379
9380
9381
9382
9383
9384
9385
9386
9387
9388
9389
9390
9391
9392
9393
9394
9395
9396
9397
9398
9399
9400
9401
9402
9403
9404
9405
9406
9407
9408
9409
9410
9411
9412
9413
9414
9415
9416
9417
9418
9419
9420

.SBTTL *COMMAND INITATOR

THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
SPECIAL COMMAND ARE ALSO EXECUTED:

RELEASE
CONROLLER CLEAR
SUBSYSTEM CLEAR
READ ALL DRIVE STATUS
READ SPECIFIED HEADER

THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS

*CALL JSR PC,C.INIT
<ADDRESS OF PARAMETER BLOCK>
RETURN

FOR THE SEQUENTIAL OPERATIONS, THE DRIVER WILL LOAD THE
LOCATIONS, PBLKT AND INTMSK.

ROUTINES USED:
W.WTCH
I.CSTS
I.STAT
I.CCLR

C.INIT: MOV R5,-(SP) ;STORE R5 ON STACK
MOV R4,-(SP) ;STORE R4 ON STACK
MOV R3,-(SP) ;STORE R3 ON STACK
MOV R2,-(SP) ;STORE R2 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV R0,-(SP) ;STORE R0 ON STACK
MOV PS,-(SP) ;STORE PSW ON STACK
MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS
MOV @16(SP),R5 ;STORE PARAMETER BLOCK ADDRESS
ADD #2,16(SP) ;ADJUST RETURN
MOV P.DRVN(R5),R4 ;STORE DRIVE NUMBER
BIC #^C<DRVMSK>,R4 ;MASK OUT JUNK
BISB INTMSK(R4),W.TIME ;SET WATCH DOG TIME
MOV R4,R3 ;STORE DRIVE NUMBER
ASL R3 ;MULTIPLY DRIVE NUMBER BY 2
MOV W.SEC,W.DRV(R3) ;LOAD WATCH-DOG TIME

MOV RKBAS,R2 ;LOAD R2 WITH RK06 ADDRESS BASE

RESET ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
DRIVE IN USE
WRITE FOR WRITE CHECK
NO CHECK
DROP DRIVE FROM TEST SEQUENCE
INHIBIT BUS ADDRESS INCREMENT

9421	050170	042765	075176	000014		BIC	#^C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)	
9422								
9423	050176	010500				MOV	R5,R0 ;STORE PARAMETER BLOCK ADDRESS	
9424	050200	062700	000016			ADD	#P.CS1,R0 ;CALCULATE FIRST LOCATION TO BE CLEARED	
9425	050204	010501				MOV	R5,R1 ;STORE PARAMETER BLOCK ADDRESS	
9426	050206	062701	000062			ADD	#P.EPAT,R1 ;CALCULATE LAST LOCATION TO BE CLEARED	
9427								
9428	050212	005020			1\$:	CLR	(R0)+ ;CLEAR RETURN PARAMETER	
9429	050214	020001				CMP	R0,R1 ;CHECK IF FINISHED	
9430	050216	101775				BLOS	1\$;NO, CLEAR NEXT RETURN PARAMETER	
9431	050220	105037	001506			CLRB	I.ISRL ;CLEAR RELEASE OR INTERRUPT ISSUED	
9432	050224	010465	000020			MOV	R4,P.CS2(R5) ;STORE DRIVE NUMBER	
9433	050230	005062	000026			CLR	RKMR1(R2) ;CLEAR RK06 MAINTENANCE REGISTER 1	
9434	050234	132765	000040	000001		BITB	#BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND	
9435	050242	001402				BEQ	3\$;NO, PROCESS	
9436	050244	000137	050624			JMP	C.SPEC ;JUMP TO SPECIAL COMMAND PROCESSOR	
9437								
9438	050250	122765	000107	000001	3\$:	CMPB	#UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND	
9439							: START SPINDLE	
9440							: RECALIBRATE	
9441							: OFFSET	
9442							: SEEK	
9443							: UNLOAD	
9444								
9445	050256	101125				BHI	25\$;NO, DRIVE COMMAND	
9446							: SELECT DRIVE	
9447							: PACK ACKNOWLEDGE	
9448							: CLEAR	
9449								
9450	050260	122765	000117	000001		CMPB	#SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER	
9451	050266	103471				BLO	20\$;YES, DATA TRANSFER COMMAND	
9452							: READ DATA	
9453							: WRITE DATA	
9454							: READ HEADER	
9455							: WRITE HEADER	
9456							: WRITE CHECK	
9457	050270	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER	
9458	050276	052765	000002	000014		BIS	#DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING	
9459	050304	005037	001464			CLR	O.WAIT ;CLEAR WAIT FOR COMMAND	
9460	050310	122765	000117	000001		CMPB	#SEEK,P.CMND(R5) ;CHECK IF SEEK	
9461	050316	001007				BNE	5\$;NO, CHECK FOR OFFSET	
9462	050320	016562	000002	000020		MOV	P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS	
9463	050326	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK	
9464	050334	000431				BR	8\$;GO ISSUE COMMAND	
9465								
9466	050336	122765	000115	000001	5\$:	CMPB	#OFFSET,P.CMND(R5) ;CHECK IF OFFSET	
9467	050344	001007				BNE	6\$;NO, CHECK FOR UNLOAD	
9468	050346	116565	000006	000032		MOVB	P.OFST(R5),P.ASOF(R5) ;STORE OFFSET	
9469	050354	016562	000032	000016		MOV	P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER	
9470	050362	000416				BR	8\$;GO ISSUE COMMAND	
9471								
9472	050364	122765	000111	000001	6\$:	CMPB	#SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE	
9473	050372	001003				BNE	7\$;NO, CHECK IF RECAL	
9474	050374	013763	001476	001544		MOV	W.MIN,W.DRV(R3) ;LOAD WATCH DOG TIME FOR 1 MINUTE	
9475	050402	122765	000113	000001	7\$:	CMPB	#RECAL,P.CMND(R5) ;CHECK IF RECAL	
9476	050410	001003				BNE	8\$;NO, CONTINUE	


```
9477 050412 013763 001474 001544      MOV      W.8SEC,W.DRV(R3) ;LOAD RECAL TIME TO 8 SECONDS
9478 050420 116565 000007 000017 8$:     MOVB     P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9479 050426 042765 165777 000016      BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
9480                                     ; AND DRIVE TYPE
9481 050434 116565 000001 000016      MOVB     P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
9482 050442 042765 000200 000014      BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
9483 050450 000460                                     BR      30$ ;GO, ISSUE COMMAND
9484
9485 050452 016562 000010 000004 20$:     MOV      P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
9486 050460 016562 000012 000002      MOV      P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
9487 050466 016562 000002 000020      MOV      P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
9488 050474 016562 000004 000006      MOV      P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
9489 050502 122765 000131 000001      CMPB     #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
9490 050510 001010      BNE     25$ ;NO, GO ISSUE THE COMMAND
9491 050512 032765 000200 000014      BIT      #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
9492 050520 001404      BEQ     25$ ;NO, GO ISSUE THE COMMAND
9493 050522 012765 000123 000016      MOV      #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
9494 050530 000406      BR      26$ ;GO ISSUE COMMAND
9495
9496 050532 116565 000001 000016 25$:     MOVB     P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
9497 050540 042765 000200 000014      BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
9498 050546 116565 000007 000017 26$:     MOVB     P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9499 050554 142765 177750 000017      BICB     #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
9500                                     ; FORMAT, DRIVE TYPE, AND BUS ADDRESS
9501                                     ; BITS 16-17
9502 050562 010537 001464      MOV      R5,O.WAIT ;LOAD WAITING FOR COMMAND
9503 050566 032765 100000 000014      BIT      #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
9504 050574 001403      BEQ     27$ ;NO, LOAD CS2
9505 050576 052765 000020 000020      BIS      #BAI,P.CS2(R5) ;SET INHIBIT BUS ADDRESS INCREMENT
9506 050604 016562 000020 000010 27$:     MOV      P.CS2(R5),RKCS2(R2) ;LOAD CS2
9507 050612 016562 000016 000000 30$:     MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
9508 050620 000137 051542      JMP      C.RTRN ;RESTORE REGISTERS
9509
9510                                     .SBTTL  *SPECIAL COMMAND PROCESSING
9511
9512 050624 122765 000141 000001  C.SPEC: CMPB     #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
9513 050632 001132      BNE     10$ ;NO, PROCESS OTHER COMMANDS
9514 050634 016562 000020 000010      MOV      P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
9515 050642 116565 000007 000017      MOVB     P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9516 050650 042765 165777 000016      BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
9517                                     ; AND DRIVE TYPE
9518 050656 112765 000001 000016      MOVB     #DR.SEL,P.CS1(R5) ;STORE COMMAND
9519 050664 016562 000016 000000      MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
9520 050672 004737 044270 2$:      JSR      PC,W.WTCH ;CALL WATCH-DOG TIMER
9521 050676 016265 000000 000016      MOV      RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
9522 050704 032765 000200 000016      BIT      #RDY,P.CS1(R5) ;WAIT FOR READY
9523 050712 001767      BEQ     2$
9524 050714 004737 047564      JSR      PC,I.CST1 ;STORE CONTROLLER REGISTERS
9525 050720 016265 000034 000040      MOV      RKMR2(R2),P.A00(R5) ;STORE STATUS BYTE 00 MESSAGE A
9526 050726 016265 000036 000042      MOV      RKMR3(R2),P.B00(R5) ;STORE STATUS BYTE 00 MESSAGE B
9527 050734 032765 100000 000016      BIT      #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
9528 050742 001443      BEQ     6$ ;NO, GATHER DRIVE STATUS
9529 050744 146437 001514 001512      BICB     INTMSK(R4),W.TIME ;RESET WATCH DOG TIMING ON THIS DRIVE
9530 050752 005063 001544      CLR      W.DRV(R3) ;CLEAR WATCH DOG COUNT
9531 050756 032765 001000 000020      BIT      #MDS,P.CS2(R5) ;CHECK IF MULTIPLE DRIVE SELECT
9532 050764 001046      BNE     9$ ;YES, INDICATE CONTROLLER ERROR
```

```
9533 050766 004037 047070 JSR RO,I.CCLR ;CLEAR ERROR
9534 050772 051542 C.RTRN ;ERROR RETURN
9535 050774 032765 010400 000020 BIT #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
9536 051002 001017 BNE 5$ ;REPORT ERROR
9537 051004 032765 000001 000036 BIT #DRA,P.DS(R5) ;CHECK IF DRIVE AVAILIABLE
9538 051012 001013 BNE 5$ ;YES, REPORT ERROR
9539 051014 156437 001514 001512 BISB INTMSK(R4),W.TIME ;SET TIMING FOR THIS DRIVE
9540 051022 013763 001476 001544 MOV W.MIN,W.DRV(R3) ;WAIT 1 MINUTE FOR RELEASE
9541 051030 052765 010000 000014 BIS #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED
9542 051036 000137 051542 JMP C.RTRN ;RESTORE REGISTERS
9543
9544 051042 004777 130410 5$: JSR PC,@A.ABNL ;REPORT ERROR
9545 051046 000137 051542 JMP C.RTRN ;RESTORE REGISTERS
9546
9547 051052 004037 047646 6$: JSR RO,I.STAT ;GATHER DRIVE STATUS
9548 051056 051542 C.RTRN ;ERROR RETURN
9549 051060 146437 001514 001512 BICB INTMSK(R4),W.TIME ;STOP WATCH-DOG TIMING ON DRIVE
9550 051066 005063 001544 CLR W.DRV(R3) ;RESET WATCH-DOG TIME
9551 051072 004777 130356 JSR PC,@A.NORM ;REPORT COMMAND COMPLETE
9552 051076 000137 051542 JMP C.RTRN ;RESTORE REGISTERS
9553
9554 051102 052737 100000 001462 9$: BIS #E.MDS,E.CONT ;SET MULTIPLE DRIVE SELECT
9555 051110 004777 130344 JSR PC,@A.CONT ;INDICATE CONTROLLER ERROR
9556 051114 000137 051542 JMP C.RTRN
9557
9558 051120 122765 000140 000001 10$: CMPB #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
9559 051126 001031 BNE 13$ ;NO, CHECK IF READ ALL HEADERS
9560 051130 010537 001464 MOV R5,O.WAIT ;STORE PARAMETER BLOCK ADDRESS IN
9561 ; WAIT FOR COMMAND
9562 051134 052765 000010 000020 BIS #RLS,P.CS2(R5) ;SET RELEASE BIT
9563 051142 016562 000020 000010 MOV P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
9564 051150 112737 000001 001506 MOVB #1,I.ISRL ;SET FLAG FOR RELEASE COMMAND
9565 051156 116565 000007 000017 MOVB P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9566 051164 042765 165777 000016 BIC #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
9567 ; AND DRIVE TYPE
9568 051172 112765 000101 000016 MOVB #SELDIV,P.CS1(R5) ;STORE COMMAND
9569 051200 016562 000016 000000 11$: MOV P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
9570 051206 000137 051542 JMP C.RTRN ;RESTORE REGISTERS
9571
9572 051212 122765 000164 000001 13$: CMPB #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
9573 051220 001047 BNE 30$ ;NO, CHECK IF CONTROLLER CLEAR
9574 051222 010537 001464 MOV R5,O.WAIT ;SET WAITING FOR COMMAND COMPLETION
9575 051226 016537 000010 001502 MOV P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
9576 051234 132765 000020 000007 BITB #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
9577 051242 001404 BEQ 14$ ;YES, LOAD 22 IN HEADER COUNT
9578 051244 012737 000024 001504 MOV #20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
9579 051252 000403 BR 22$ ;GO ISSUE READ HEADER COMMAND
9580
9581 051254 012737 000026 001504 14$: MOV #22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
9582 051262 016562 000002 000020 22$: MOV P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
9583 051270 016562 000004 000006 MOV P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
9584 051276 016562 000020 000010 MOV P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
9585 051304 116565 000007 000017 MOVB P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9586 051312 042765 165777 000016 BIC #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
9587 ; AND FORMAT
9588 051320 112765 000125 000016 MOVB #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
```

```

9589 051326 016562 000016 000000      MOV  P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
9590 051334 000137 051542      JMP  C.RTRN                ;RESTORE REGISTERS
9591
9592 051340 122765 000176 000001 30$:  CMPB #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
9593 051346 001004      BNE 32$                   ;NO, CHECK IF SUBSYSTEM CLEAR
9594 051350 004037 047070      JSR  RO,I.CCLR            ;CLEAR CONTROLLER
9595 051354 051542      C.RTRN                    ;ERROR RETURN
9596 051356 000467      BR   40$                 ;INDICATE NORMAL RETURN
9597
9598 051360 122765 000177 000001 32$:  CMPB #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
9599 051366 001406      BEQ 36$                   ;YES, CLEAR SUBSYSTEM
9600 051370 052737 000100 001462 34$:  BIS  #E.ILLD,E.CONT      ;SET ILLEGAL DRIVER COMMAND
9601 051376 004777 130056      JSR  PC,@A.CONT          ;REPORT ERROR
9602 051402 000457      BR   C.RTRN              ;RESTORE REGISTERS
9603
9604 051404 012762 000040 000010 36$:  MOV  #SCLR,RKCS2(R2)     ;ISSUE SUBSYSTEM CLEAR
9605 051412 016265 000000 000016      MOV  RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
9606 051420 032765 100000 000016      BIT  #CERR,P.CS1(R5)     ;CLEAR IF CONTROLLER ERROR RESET
9607 051426 001406      BEQ 37$                   ;NO, FINISH COMMAND
9608 051430 052737 000001 001462      BIS  #BIT0,E.CONT        ;SET CLEAR SUBSYSTEM DID NOT CLEAR
9609                                ; CONTROLLER ERROR
9610 051436 004777 130016      JSR  PC,@A.CONT          ;REPORT ERROR
9611 051442 000437      BR   C.RTRN              ;RESTORE REGISTERS
9612
9613 051444 013746 001470      MOV  W.MILI,-(SP)        ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
9614                                ; TO DISAPPEAR
9615 051450 016265 000000 000016 38$:  MOV  RKCS1(R2),P.CS1(R5) ;STORE CS1
9616 051456 032765 040000 000016      BIT  #DI,P.CS1(R5)       ;CHECK IF ATTENTIONS CLEARED
9617 051464 001411      BEQ 39$                   ;YES, FINISH COMMAND
9618 051466 005316      DEC  (SP)                 ;DECREMENT 16 MILLISECOND COUNT
9619 051470 001367      BNE 38$                   ;CHECK DRIVE INTERRUPT AGAIN
9620 051472 005726      TST  (SP)+                ;ADJUST STACK
9621 051474 052737 000040 001462      BIS  #E.SCLR,E.CONT      ;SET SUBSYSTEM CLEAR DID NOT CLEAR
9622                                ; DRIVE ATTENTIONS
9623 051502 004777 127752      JSR  PC,@A.CONT          ;REPORT ERROR
9624 051506 000415      BR   C.RTRN              ;RESTORE REGISTER
9625
9626 051510 005726 000400 000014 39$:  TST  (SP)+                ;ADJUST STACK
9627 051512 032765 000400 000014      BIT  #NOCHK,P.PRST(R5)   ;CHECK IF NO CHECK MODE
9628 051520 001010      BNE  C.RTRN               ;YES, RESTORE REGISTERS
9629 051522 112737 177777 001506      MOVB #-1,I.ISRL          ;SET INTERRUPT ENABLE SET
9630 051530 012762 000100 000000      MOV  #IE,RKCS1(R2)       ;SET INTERRUPT ENABLE
9631 051536 004777 127712 40$:  JSR  PC,@A.NORM          ;INDICATE NORMAL TERMINATION
9632
9633 051542 012637 177776      C.RTRN: MOV (SP)+,PS      ;RESTORE PSW
9634 051546 012600      MOV  (SP)+,R0            ;RESTORE R0
9635 051550 012601      MOV  (SP)+,R1            ;RESTORE R1
9636 051552 012602      MOV  (SP)+,R2            ;RESTORE R2
9637 051554 012603      MOV  (SP)+,R3            ;RESTORE R3
9638 051556 012604      MOV  (SP)+,R4            ;RESTORE R4
9639 051560 012605      MOV  (SP)+,R5            ;RESTORE R5
9640 051562 000207      RTS  PC                  ;RETURN
  
```

.SBTTL *ENQUEUE ROUTINE

```
*****  
*  
* THIS ROUTINE WILL BUT THE ADDRESS OF THE PARAMETER IN THE  
* QUEUE SPECIFIED BY THE LOCATION FOLLOWING THE CALL.  
*  
*CALL JSR R0,Q.PUSH  
* <ADDRESS OF QUEUE>  
* RETURN  
*****
```

9641
9642
9643
9644
9645
9646
9647
9648
9649
9650
9651
9652
9653
9654
9655
9656
9657
9658
9659
9660
9661
9662
9663
9664
9665
9666
9667
9668
9669
9670
9671
9672
9673
9674

051564	010446		
051566	010346		
051570	013746	177776	
051574	013737	001452	177776
051602	012004		
051604	005065	000064	
051610	005724		
051612	001405		
051614	011403		
051616	010563	000064	
051622	010514		
051624	000402		
051626	010514		
051630	010544		
051632	012637	177776	
051636	012603		
051640	012604		
051642	000200		

```
Q.PUSH: MOV R4,-(SP) :STORE R4 ON STACK  
MOV R3,-(SP) :STORE R3 ON STACK  
MOV PS,-(SP) :STORE PSW ON STACK  
MOV RKPRI,PS :LOCK OUT RK06 INTERRUPTS  
MOV (R0)+,R4 :STORE QUEUE ADDRESS  
CLR P.QLNK(R5) :CLEAR QUEUE LINK  
TST (R4)+ :CHECK IF QUEUE IS EMPTY  
BEQ 1$ :YES, PUT PARAMETER BLOCK ADDRESS  
: AT HEAD AND TAIL OF THE QUEUE  
MOV (R4),R3 :STORE TAIL OF QUEUE  
MOV R5,P.QLNK(R3) :UPDATE QUEUE LINK  
MOV R5,(R4) :UPDATE QUEUE TAIL  
BR 2$  
  
1$: MOV R5,(R4) :LOAD TAIL OF QUEUE  
MOV R5,-(R4) :LOAD HEAD OF QUEUE  
  
2$: MOV (SP)+,PS :RESTORE PSW  
MOV (SP)+,R3 :RESTORE R3  
MOV (SP)+,R4 :RESTORE R4  
RTS R0 :RETURN
```

9675
9676
9677
9678
9679
9680
9681
9682
9683
9684
9685
9686
9687
9688
9689
9690
9691
9692
9693
9694
9695
9696
9697
9698
9699
9700
9701
9702
9703
9704
9705
9706
9707
9708

.SBTTL *POP QUEUE ROUTINE

```
*****  
*  
* THIS ROUTINE WILL PUT THE ADDRESS OF THE FIRST PARAMTER  
* BLOCK ADDRESS OF THIS QUEUE ON THE STACK AND REMOVE IT  
* FROM THE QUEUE DESIGNATED BY THE CALL. A RETURN OF ZERO  
* ON THE STACK INDICATES THAT THE QUEUE WAS EMPTY. OTHERWISE,  
* THE STACK WILL CONTAIN THE ADDRESS OF THE FIRST  
* PARAMETER BLOCK ON THE QUEUE.  
*  
*CALL JSR R0,Q.POP  
* <ADDRESS OF QUEUE>  
* RETURN  
*  
*****
```

```
Q.POP: MOV (SP),-(SP) ;CREATE SPACE ON THE STACK  
MOV R5,-(SP) ;STORE R5 ON THE STACK  
MOV R4,-(SP) ;STORE R4 ON THE STACK  
MOV PS,-(SP) ;STORE PSW ON THE STACK  
MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS  
MOV (R0)+,R4 ;STORE QUEUE ADDRESS  
MOV (R4),R5 ;STORE PARAMETER BLOCK ADDRESS  
MOV R5,10(SP) ;LOAD ADDRESS ON STACK  
BEQ 1$ ;IF QUEUE IS EMPTY, RETURN  
MOV P,QLNK(R5),(R4)+ ;READJUST HEAD OF THE QUEUE  
BNE 1$ ;IF NOT LAST ELEMENT OF THE QUEUE, RETURN  
CLR (R4) ;CLEAR TAIL POINTER IF LAST ELEMENT OF QUEUE  
  
1$: MOV (SP)+,PS ;RESTORE PSW  
MOV (SP)+,R4 ;RESTORE R4  
MOV (SP)+,R5 ;RESTORE R5  
RTS R0 ;RETURN
```

177776
001452 177776
000010
000064
177776

9709
9710
9711
9712
9713
9714
9715
9716
9717
9718
9719
9720
9721
9722
9723
9724
9725
9726 051720 012704 001324
9727
9728 051724 011405
9729 051726 001425
9730 051730 032765 000002 000014
9731 051736 001404
9732 051740 010504
9733 051742 062704 000064
9734 051746 000766
9735
9736 051750 016514 000064
9737 051754 001012
9738
9739 051756 022704 001324
9740 051762 001405
9741 051764 162704 000064
9742 051770 010437 001326
9743 051774 000207
9744
9745 051776 005024
9746 052000 005014
9747 052002 000207

.SBTTL *SEARCH QUEUE FOR DRIVE NOT POSITIONING

* THIS ROUTINE WILL SEARCH THROUGH THE COMMAND INITIATION
* QUEUE FOR THE FIRST PARAMETER BLOCK WHOSE DRIVE IS
* NOT POSITIONING. IF THE QUEUE IS EMPTY OF ALL DRIVES
* ARE POSITIONING ZERO IS PLACED IN R5. OTHERWISE,
* THE ADDRESS OF THE PARAMETER BLOCK IS PLACED IN R5.
*
*CALL JSR PC,Q.SRCH
* RETURN
*
*NOTE: THIS ROUTINE DESTROYS R4.
*

Q.SRCH: MOV #CINITQ,R4 ;LOAD ADDRESS OF COMMAND INITIATION
; QUEUE IN R4
1\$: MOV (R4),R5 ;GET NEXT QUEUE ELEMENT
BEQ 5\$;IF END OF QUEUE, RETURN
BIT #DRVPOS,P.PRST(R5) ;CHECK IF DRIVE IS POSITIONING
BEQ 2\$;NO, RETURN PARAMETER BLOCK ADDRESS
MOV R5,R4 ;UPDATE LINK ADDRESS POINTER
ADD #P.QLNK,R4 ;CALCULATE LINK ADDRESS
BR 1\$;GET NEXT ELEMENT OF QUEUE
2\$: MOV P.QLNK(R5),(R4) ;UPDATE QUEUE LINK
BNE 5\$;CHECK IF LAST ELEMENT OF QUEUE
; IF NOT, DO NOT UPDATE QUEUE TAIL
CMP #CINITQ,R4 ;CHECK IF ONLY ELEMENT IN QUEUE
BEQ 4\$;YES, CLEAR HEAD AND TAIL
SUB #P.QLNK,R4 ;ADJUST R4 FOR TAIL POINTER
MOV R4,CINITQ+2 ;LOAD TAIL
RTS PC ;RETURN
4\$: CLR (R4)+ ;CLEAR HEAD OF QUEUE
CLR (R4) ;CLEAR TAIL OF QUEUE
5\$: RTS PC ;RETURN

```
9748 .SBTTL *REMOVE PARAMETER BLOCK FROM COMMAND INITIATION QUEUE
9749
9750 :*****
9751 :*
9752 :* THIS ROUTINE WILL CHECK IF THE CURRENT PARAMETER BLOCK
9753 :* IS IN THE COMMAND INITIATION QUEUE. IF IT IS, THIS
9754 :* ROUTINE WILL REMOVE IT FROM THE COMMAND INITIATION QUEUE.
9755 :*
9756 :*CALL JSR PC,Q.RMOV
9757 :* RETURN
9758 :*
9759 :*NOTE: R5 CONTAINS ADDRESS OF CURRENT PARAMETER BLOCK.
9760 :*
9761 :*****
9762
9763 052004 032765 040000 000014 Q.RMOV: BIT #Q.INIT,P.PRST(R5) :CHECK IF PARAMETER BLICK ENQUEUED IN
9764 : INITIATION QUEUE
9765 052012 001001 BNE 5$ :YES, REMOVE PARAMETER BLOCK
9766 : FROM COMMAND INITIATION QUEUE
9767 052014 000207 RTS PC :RETURN
9768
9769 052016 010446 5$: MOV R4,-(SP) :STORE R4 ON STACK
9770 052020 010346 MOV R3,-(SP) :STORE R3 ON THE STACK
9771 052022 012704 001324 MOV #CINITQ,R4 :LOAD ADDRESS OF COMMAND INITIATION
9772 : QUEUE IN R4
9773 052026 011403 10$: MOV (R4),R3 :GET NEXT QUEUE ELEMENT
9774 052030 001001 BNE 11$ :CHECK IF NO MORE ELEMENTS
9775 052032 000000 HALT : *** FLAGS DO NOT AGREE WITH
9776 : COMMAND INITIATION QUEUE
9777 052034 020305 11$: CMP R3,R5 :CHECK IF CORRECT PARAMETER BLOCK
9778 052036 001404 BEQ 12$ :YES, ADJUST QUEUE
9779 052040 010304 MOV R3,R4 :UPDATE LINK ADDRESS POINTER
9780 052042 062704 000064 ADD #P.QLNK,R4 :CALCULATE LINK ADDRESS
9781 052046 000767 BR 10$ :GET NEXT ELEMENT OF QUEUE
9782
9783 052050 016314 000064 12$: MOV P.QLNK(R3),(R4) :UPDATE QUEUE LINK
9784 052054 001012 BNE 15$ :CHECK IF LAST ELEMENT OF QUEUE
9785 : IF NOT, DO NOT UPDATE QUEUE TAIL
9786 052056 022704 001324 CMP #CINITQ,R4 :CHECK IF ONLY ELEMENT IN QUEUE
9787 052062 001405 BEQ 14$ :YES, CLEAR HEAD AND TAIL
9788 052064 162704 000064 SUB #P.QLNK,R4 :ADJUST R4 FOR TAIL POINTER
9789 052070 010437 001326 MOV R4,CINITQ+2 :LOAD TAIL
9790 052074 000402 BR 15$ :RETURN
9791
9792 052076 005024 14$: CLR (R4)+ :CLEAR HEAD OF QUEUE
9793 052100 005014 CLR (R4) :CLEAR TAIL OF QUEUE
9794 052102 042765 040006 000014 15$: BIC #DRVPT!DRVPOS!Q.INIT,P.PRST(R5) :CLEAR DRIVE POSITIONING,
9795 : DRIVE POSITIONING FOR DATA TRANSFER, AND
9796 : PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
9797
9798 052110 012603 MOV (SP)+,R3 :RESTORE R3
9799 052112 012604 MOV (SP)+,R4 :RESTORE R4
9800 052114 000207 RTS PC :RETURN
```

9801
9802
9803
9804
9805
9806
9807
9808
9809
9810
9811
9812
9813
9814
9815
9816
9817
9818
9819
9820
9821
9822
9823
9824
9825
9826
9827
9828
9829
9830
9831
9832
9833
9834
9835
9836
9837
9838
9839
9840
9841
9842
9843
9844
9845
9846
9847
9848
9849
9850
9851
9852
9853
9854
9855
9856

.SBTTL *COMMAND OPTIMIZER

THIS ROUTINE WILL INITIATE THE COMMAND AS SPECIFIED IN THE
PARAMETER BLOCK OF THE FIRST DRIVE WHICH IS NOT POSITIONING
IN THE COMMAND INITIATION QUEUE.

*CALL JSR PC,C.OPT
RETURN

ROUTINES USED:
C.INIT
Q.PUSH
Q.SRCH

C.OPT: MOV R5,-(SP) ;STORE R5 ON STACK
MOV R4,-(SP) ;STORE R4 ON STACK
MOV R3,-(SP) ;STORE R3 ON STACK
MOV R2,-(SP) ;STORE R2 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV R0,-(SP) ;STORE R0 ON STACK
MOV PS,-(SP) ;STORE PSW ON STACK
MOV #1,OPTFLG ;SET FLAG FOR CMND OPTIMIZER
MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS
TST O.WAIT ;CHECK IF WAITING FOR COMMAND COMPLETION
BNE 12\$;YES, RESTORE REGISTERS
MOV RKBAS,R2 ;LOAD R2 WITH BASE ADDRESS OF THE RK06
; REGISTERS
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
BIT #RDY,T.CS1 ;CHECK IF READY SET
BEQ 12\$;NO, WAIT FOR COMMAND COMPLETION
JSR PC,Q.SRCH ;SEARCH FOR PARAMETER BLOCK ADDRESS
TST R5 ;CHECK IF NO COMMAND CAN BE ISSUED
BNE 1\$;YES, ISSUE COMMAND
BIT #DI,T.CS1 ;CHECK IF DRIVE INTERRUPT WAITING
BEQ 12\$;NO, RETURN
MOVB #-1,I.ISRL ;SET FLAG THAT INTERRUPT WAS FORCED
MOV #INTR,RKCS1(R2) ;FORCE INTERRUPT TO SCAN DRIVE ATTENTIONS
JMP O.RTRN ;RESTORE REGISTERS

12\$: MOVB #DR_SEL,T.CS1 ;SET DRIVE SELECT BIT
JSR RO,I.ISSU ;ISSUE DRIVE SELECT COMMAND
O.RTRN ;ERROR RETURN ADDRESS
BIT #NED,T.CS2 ;SEE IF NED ERROR SET
BNE 24\$;BR IF YES
BIT #DRA,T.DS ;SEE IF DRIVE IS AVAILABLE
BNE 24\$;BR IF AVAILABLE
MOV T.CS2,R3 ;GET RKCS2
BIC #177770,R3 ;CLEAR ALL BUT DRIVE NO.
BISB INTMSK(R3),W.TIME ;SET BIT FOR THIS DRIVE
ASL R3
MOV W.MIN,W.DRV(R3) ;SET TIMER FOR 1 MINUTE FOR RELEASE
CLR O.WAIT ;CLEAR WAITING FOR CMND COMPLETION

1\$: MOVB #DR_SEL,T.CS1 ;SET DRIVE SELECT BIT
JSR RO,I.ISSU ;ISSUE DRIVE SELECT COMMAND
O.RTRN ;ERROR RETURN ADDRESS
BIT #NED,T.CS2 ;SEE IF NED ERROR SET
BNE 24\$;BR IF YES
BIT #DRA,T.DS ;SEE IF DRIVE IS AVAILABLE
BNE 24\$;BR IF AVAILABLE
MOV T.CS2,R3 ;GET RKCS2
BIC #177770,R3 ;CLEAR ALL BUT DRIVE NO.
BISB INTMSK(R3),W.TIME ;SET BIT FOR THIS DRIVE
ASL R3
MOV W.MIN,W.DRV(R3) ;SET TIMER FOR 1 MINUTE FOR RELEASE
CLR O.WAIT ;CLEAR WAITING FOR CMND COMPLETION

9857	052330	004037	047070			JSR	R0,I.CCLR	:ISSUE CONTROLLER CLEAR
9858	052334	052622				O.RTRN		:ERROR RETURN ADDRESS
9859	052336	042765	040006	000014		BIC	#DRVPOS!DRVPDT!Q.INIT,P.PRST(R5)	
9860	052344	052765	010000	000014	10\$:	BIS	#DRVSZD,P.PRST(R5)	:SET DRIVE SIEZED FLAG
9861	052352	000137	052622			JMP	O.RTRN	:EXIT
9862	052356				24\$:			
9863	052356	005065	000212			CLR	P.ERR(R5)	:CLEAR ERROR STATUS INFORMATION
9864	052362	122765	000164	000001		CMPB	#RDALHD,P.CMND(R5)	:CHECK IF READ ALL HEADERS
9865	052370	001410				BEQ	2\$:YES, CHECK IF IMPLIED SEEK
9866	052372	122765	000121	000001		CMPB	#RDDATA,P.CMND(R5)	:CHECK IF DATA TRANSFER
9867	052400	101021				BHI	3\$:NO, GO TO THE COMMAND INITIATOR
9868	052402	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF SPECIAL COMMAND
9869	052410	103415				BLO	3\$:YES, GO TO THE COMMAND INITIATOR
9870	052412	116504	000000		2\$:	MOVB	P.DRVN(R5),R4	:STORE DRIVE NUMBER
9871	052416	136437	001514	001513		BITB	INTMSK(R4),O.OVER	:CHECK IF OVERLAPPED OPERATIONS FOR THIS DRIVE
9872	052424	001407				BEQ	3\$:NO, GO TO THE COMMAND INITIATOR
9873	052426	032765	000004	000014		BIT	#DRVPDT,P.PRST(R5)	:CHECK IF POSITIONING FOR
9874								: DATA TRANSFER OCCURED
9875	052434	001411				BEQ	5\$:NO, ISSUE SEEK
9876	052436	042765	040004	000014		BIC	#DRVPDT!Q.INIT,P.PRST(R5)	:CLEAR POSITIONING IN PROGRESS FOR
9877								: DATA TRANSFER AND PARAMETER BLOCK ENQUEUED
9878								: IN INITIATION QUEUE
9879								
9880	052444	010537	052454		3\$:	MOV	R5,4\$:LOAD PARAMETER BLOCK ADDRESS
9881	052450	004737	050074			JSR	PC,C.INIT	:ISSUE COMMAND
9882	052454	000000			4\$:	.WORD	0	:ADDRESS OF PARAMETER BLOCK
9883								: PLACED HERE
9884	052456	000461				BR	O.RTRN	:RESTORE REGISTERS
9885								
9886	052460	156437	001514	001512	5\$:	BISB	INTMSK(R4),W.TIME	:SET WATCH DOG TIMING
9887	052466	010403				MOV	R4,R3	:STORE DRIVE NUMBER
9888	052470	006303				ASL	R3	:MULTIPLY DRIVE NUMBER BY 2
9889	052472	013763	001472	001544		MOV	W.SEC,W.DRV(R3)	:LOAD WATCH-DOG TIME
9890	052500	005037	001464			CLR	O.WAIT	:CLEAR WAIT FOR COMMAND
9891	052504	105037	001506			CLRB	I.ISRL	:CLEAR RELEASE OF INTERRUPT ISSUED
9892								
9893					:		RESET ALL THE BITS IN THE PROGRAM STATUS REGISTER. EXCEPT	
9894					:		DRIVE IN USE	
9895					:		WRITE FOR WRITE CHECK	
9896					:		NO CHECK	
9897					:		DROP DRIVE FROM TEST SEQUENCE	
9898					:		INHIBIT BUS ADDRESS INCREMENT	
9899	052510	042765	075176	000014		BIC	#*C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)	
9900					:		SET DRIVE POSITIONING, DRIVE POSITIONING BECAUSE OF DATA TRANSFER,	
9901					:		AND PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE	
9902					:			
9903	052516	052765	040006	000014		BIS	#DRVPOS!DRVPDT!Q.INIT,P.PRST(R5)	
9904								
9905	052524	010500				MOV	R5,R0	:STORE PARAMTER BLOCK ADDRESS
9906	052526	062700	000016			ADD	#P.CS1,R0	:CALCULATE FIRST LOCATION TO BE CLEARED
9907	052532	010501				MOV	R5,R1	:STORE PARAMETER BLOCK ADDRESS
9908	052534	062701	000062			ADD	#P.EPAT,R1	:CALCULATE LAST LOCATION TO BE CLEARED
9909								
9910	052540	005020			6\$:	CLR	(R0)+	:CLEAR RETURN PARAMETERS
9911	052542	020001				CMP	R0,R1	:CHECK IF FINISHED
9912	052544	101775				BLOS	6\$:NO, CLEAR NEXT RETURN PARAMETER

9913	052546	005062	000026		CLR	RKMR1(R2)	:CLEAR MAINTENANCE REGISTER 1
9914	052552	016562	000002	000020	MOV	P.CYLN(R5),RKDCYL(R2)	:LOAD CYLINDER ADDRESS
9915	052560	010462	000010		MOV	R4,RKCS2(R2)	:LOAD DEVICE NUMBER
9916	052564	116565	000007	000017	MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
9917	052572	042765	165777	000016	BIC	#^C<CFMT!CDT>,P.CS1(R5)	:CLEAR BITS EXCEPT FORMAT AND
9918							: DRIVE TYPE
9919	052600	112765	000117	000016	MOVB	#SEEK,P.CS1(R5)	:STORE COMMAND
9920	052606	016562	000016	000000	MOV	P.CS1(R5),RKCS1(R2)	:ISSUE SEEK
9921	052614	004037	051564		JSR	R0,Q.PUSH	:REQUEUE COMMAND IN COMMAND
9922	052620	001324			CINITQ		: INITIATION QUEUE
9923							
9924	052622	005037	001500		O.RTRN: CLR	OPTFLG	:CLEAR OPTIMIZER FLAG
9925	052626	012637	177776		MOV	(SP)+,PS	:RESTORE PSW
9926	052632	012600			MOV	(SP)+,R0	:RESTORE R0
9927	052634	012601			MOV	(SP)+,R1	:RESTORE R1
9928	052636	012602			MOV	(SP)+,R2	:RESTORE R2
9929	052640	012603			MOV	(SP)+,R3	:RESTORE R3
9930	052642	012604			MOV	(SP)+,R4	:RESTORE R4
9931	052644	012605			MOV	(SP)+,R5	:RESTORE R5
9932	052646	000207			RTS	PC	:RETURN

CZR6PDO RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 PAGE 228
MEMORY CHECK ENABLE TEST

SEQ 0225

9933
9934
9935 052650 104401 065276
9936 052654 011646
9937 052656 004737 053224
9938 052662 104401 001165
9939 052666 000002

.SBTTL MEMORY CHECK ENABLE TEST

MEMERR: TYPE .ERR400 :TYPE UNEXPECTED MEMORY PARITY TRAP
MOV (SP),-(SP) :TYPE PC VALUE
JSR PC,BINOCT
TYPE .SCRLF
RTI :RETURN

```
.SBTTL TYPE ROUTINE
*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
*      TYPE
*      MESADR
*
9957 052670 105737 001157 $TYPE: TSTB $TPFLG      ;;IS THERE A TERMINAL?
9958 052674 100002          BPL 1$              ;;BR IF YES
9959 052676 000000          HALT              ;;HALT HERE IF NO TERMINAL
9960 052700 000430          BR 3$              ;;LEAVE
9961 052702 010046          1$: MOV RO,-(SP)      ;;SAVE RO
9962 052704 017600 000002  MOV @2(SP),RO      ;;GET ADDRESS OF ASCIZ STRING
9963 052710 122737 000001 001210  CMPB #APTENV,$ENV      ;;RUNNING IN APT MODE
9964 052716 001011          BNE 62$          ;;NO,GO CHECK FOR APT CONSOLE
9965 052720 132737 000100 001211  BITB #APTSPOOL,$ENVM    ;;SPOOL MESSAGE TO APT
9966 052726 001405          BEQ 62$          ;;NO,GO CHECK FOR CONSOLE
9967 052730 010037 052740  MOV RO,61$          ;;SETUP MESSAGE ADDRESS FOR APT
9968 052734 004737 055720  JSR PC,$ATY3        ;;SPOOL MESSAGE TO APT
9969 052740 000000          61$: .WORD 0          ;;MESSAGE ADDRESS
9970 052742 132737 000040 001211  62$: BITB #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
9971 052750 001003          BNE 60$          ;;YES,SKIP TYPE OUT
9972 052752 112046          2$: MOVB (RO)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
9973 052754 001005          BNE 4$          ;;BR IF IT ISN'T THE TERMINATOR
9974 052756 005726          TST (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
9975 052760 012600          60$: MOV (SP)+,RO      ;;RESTORE RO
9976 052762 062716 000002  3$: ADD #2,(SP)      ;;ADJUST RETURN PC
9977 052766 000002          RTI              ;;RETURN
9978 052770 122716 000011  4$: CMPB #HT,(SP)      ;;BRANCH IF <HT>
9979 052774 001430          BEQ 8$          ;;BRANCH IF NOT <CRLF>
9980 052776 122716 000200  CMPB #CRLF,(SP)
9981 053002 001006          BNE 5$          ;;POP <CR><LF> EQUIV
9982 053004 005726          TST (SP)+          ;;TYPE A CR AND LF
9983 053006 104401          TYPE
9984 053010 001165          $CRLF
9985 053012 105037 053220  CLRB $CHARCNT      ;;CLEAR CHARACTER COUNT
9986 053016 000755          BR 2$          ;;GET NEXT CHARACTER
9987 053020 004737 053102  5$: JSR PC,$TYPEC      ;;GO TYPE THIS CHARACTER
9988 053024 123726 001156  6$: CMPB $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
9989 053030 001350          BNE 2$          ;;IF NO GO GET NEXT CHAR.
9990 053032 013746 001154  MOV $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
9991                                ;;AND THE NULL CHAR.
9992 053036 105366 000001  7$: DECB 1(SP)        ;;DOES A NULL NEED TO BE TYPED?
9993 053042 002770          BLT 6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
9994 053044 004737 053102  JSR PC,$TYPEC      ;;GO TYPE A NULL
9995 053050 105337 053220  DECB $CHARCNT      ;;DO NOT COUNT AS A COUNT
```

```
9996 053054 000770 BR 7$ ::LOOP
9997
9998
9999
10000 053056 112716 000040
10001 053062 004737 053102
10002 053066 132737 000007 053220
10003 053074 001372
10004 053076 005726
10005 053100 000724
10006 053102
10007 053102 105777 126036
10008 053106 100022
10009 053110 017746 126032
10010 053114 042716 177600
10011 053120 122716 000023
10012 053124 001012
10013 053126
10014 053126 105777 126012
10015 053132 100375
10016 053134 117716 126006
10017 053140 042716 177600
10018 053144 122716 000021
10019 053150 001366
10020 053152
10021 053152 005726
10022 053154
10023 053154 105777 125770
10024 053160 100375
10025 053162 116677 000002 125762
10026 053170 122766 000015 000002
10027 053176 001003
10028 053200 105037 053220
10029 053204 000406
10030 053206 122766 000012 000002
10031 053214 001402
10032 053216 105227
10033 053220 000000
10034 053222 000207
10035
10036
10037
10038
10039
10040
10041
10042
10043
10044
10045
10046
10047
10048
10049
10050 053224
10051 053224 010346

      BR 7$ ::LOOP
      ;HORIZONTAL TAB PROCESSOR
8$:   MOVB #' (SP) ::REPLACE TAB WITH SPACE
9$:   JSR PC,$TYPEC ::TYPE A SPACE
      BITB #7,$CHARCNT ::BRANCH IF NOT AT
      BNE 9$ ::TAB STOP
      TST (SP)+ ::POP SPACE OFF STACK
      BR 2$ ::GET NEXT CHARACTER
$TYPEC:
      TSTB @STKS ::CHAR IN KYBD BUFFER?
      BPL 10$ ::BR IF NOT
      MOV @STKB,-(SP) ::GET CHAR
      BIC #177600,(SP) ::STRIP EXTRANEIOUS BITS
      CMPB #$XOFF,(SP) ::WAS CHAR XOFF
      BNE 102$ ::BR IF NOT
101$:
      TSTB @STKS ::WAIT FOR CHAR
      BPL 101$
      MOVB @STKB,(SP) ::GET CHAR
      BIC #177600,(SP) ::STRIP IT
      CMPB #$XON,(SP) ::WAS IT XON?
      BNE 101$ ::BR IF NOT
102$:
      TST (SP)+ ::FIX STACK
10$:
      TSTB @STPS ::WAIT UNTIL PRINTER IS READY
      BPL 10$
      MOVB 2(SP),@STPB ::LOAD CHAR TO BE TYPED INTO DATA REG.
      CMPB #CR,2(SP) ::IS CHARACTER A CARRIAGE RETURN?
      BNE 1$ ::BRANCH IF NO
      CLRB $CHARCNT ::YES--CLEAR CHARACTER COUNT
      BR $TYPEX ::EXIT
1$:   CMPB #LF,2(SP) ::IS CHARACTER A LINE FEED?
      BEQ $TYPEX ::BRANCH IF YES
      INCB (PC)+ ::COUNT THE CHARACTER
      $CHARCNT: .WORD 0 ::CHARACTER COUNT STORAGE
      $TYPEX: RTS PC

.SBTTL BINARY TO OCTAL (ASCII) CONVERSION
*****
*
* THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A
* 6-DIGIT OCTAL (ASCII) NUMBER.
*
*CALL MOV NUM,-(SP)
* JSR PC,BINOCT
* RETURN ::NUMBER IN OCTSTG
*****
BINOCT:
MOV R3,-(SP) ::PUSH R3 ON STACK
```

```

10052 053226 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
10053 053230 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
10054 053232 012704 000006  MOV      #6,R4         ;;STORE COUNT
10055 053236 016605 000010  MOV      10(SP),R5      ;;PICKUP INPUT NUMBER
10056 053242 012703 053320  MOV      #OCTSTG,R3     ;;LOAD ADDRESS OF CONVERTTED STRING
10057 053246 005013      CLR      (R3)          ;;CLEAR OUTPUT BYTE
10058 053250 006105      ROL      R5            ;;ROTATE MSB INTO 'C'
10059 053252 000404      BR       3$           ;;GO DO MSB
10060
10061 053254 006105      2$:     ROL      R5            ;;FORM THIS DIGIT
10062 053256 006105      ROL      R5
10063 053260 006105      ROL      R5
10064 053262 110513      MOVB     R5,(R3)
10065 053264 106113      3$:     ROLB     (R3)          ;;GET LSB OF THIS DIGIT
10066 053266 142713 177770  BICB     #177770,(R3)   ;;GET RID OF JUNK
10067 053272 152723 000060  BISB     #'0,(R3)+     ;;MAKE IT ASCII
10068 053276 005304      DEC      R4           ;;CHECK IF DONE
10069 053300 001365      BNE      2$           ;;NO CONVERT NEXT DIGIT
10070 053302 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
10071 053304 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
10072 053306 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
10073 053310 012616      MOV      (SP)+,(SP)    ;;SETUP STACK FOR RETURN
10074 053312 104401 053320  TYPE     ,OCTSTG       ;;TYPE NUMBER
10075 053316 000207      RTS      PC           ;;RETURN
10076
10077 053320 000006      OCTSTG: .BLKB    6
10078 053326      000      .BYTE    0,0
10079
10080      .SBTTL  TTY INPUT ROUTINE
10081
10082      ;*****
10083      .ENABL  LSB
10084
10085      .DSABL  LSB
10086
10087
10088      ;*****
10089      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
10090      ;*CALL:
10091      ;*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
10092      ;*      RETURN HERE   ;;CHARACTER IS ON THE STACK
10093      ;*                  ;;WITH PARITY BIT STRIPPED OFF
10094      ;*
10095      ;*
10096 053330 011646      $RDCHR: MOV      (SP),-(SP)  ;;PUSH DOWN THE PC
10097 053332 016666 000004 000002  MOV      4(SP),2(SP)   ;;SAVE THE PS
10098 053340 105777 125600      1$:     TSTB     @TKS     ;;WAIT FOR
10099 053344 100375      BPL      1$           ;;A CHARACTER
10100 053346 117766 125574 000004  MOVB     @TKB,4(SP)    ;;READ THE TTY
10101 053354 042766 177600 000004  BIC      #'^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
10102 053362 026627 000004 000023  CMP      4(SP),#23    ;;IS IT A CONTROL-S?
10103 053370 001013      BNE      3$           ;;BRANCH IF NO
10104 053372 105777 125546      2$:     TSTB     @TKS     ;;WAIT FOR A CHARACTER
10105 053376 100375      BPL      2$           ;;LOOP UNTIL ITS THERE
10106 053400 117746 125542      MOVB     @TKB,-(SP)   ;;GET CHARACTER
10107 053404 042716 177600      BIC      #'^C177,(SP) ;;MAKE IT 7-BIT ASCII

```

```

10108 053410 022627 000021      CMP      (SP)+,#21      ::IS IT A CONTROL-Q?
10109 053414 001366              BNE      2$            ::IF NOT DISCARD IT
10110 053416 000750              BR       1$            ::YES, RESUME
10111 053420 026627 000004 000021 3$:    CMP      4(SP),#$XON   ::IS IT A RANDOM XON?      :RAN001
10112 053426 001744              BEQ      1$            ::BRANCH IF YES           :RAN001
10113 053430 026627 000004 000140      CMP      4(SP),#140   ::IS IT UPPER CASE?
10114 053436 002407              BLT      4$            ::BRANCH IF YES
10115 053440 026627 000004 000175      CMP      4(SP),#175   ::IS IT A SPECIAL CHAR?
10116 053446 003003              BGT      4$            ::BRANCH IF YES
10117 053450 042766 000040 000004      BIC      #40,4(SP)    ::MAKE IT UPPER CASE
10118 053456 000002      4$:      RTI                ::GO BACK TO USER
10119                               ::*****
10120                               ::*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
10121                               ::*CALL:
10122                               ::*      RDLIN          ::INPUT A STRING FROM THE TTY
10123                               ::*      RETURN HERE   ::ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
10124                               ::*                               ::TERMINATOR WILL BE A BYTE OF ALL 0'S
10125
10126 053460 010346      $RDLIN: MOV      R3,-(SP)    ::SAVE R3
10127 053462 005046      CLR      -(SP)        ::CLEAR THE RUBOUT KEY
10128 053464 012703 053714      1$:      MOV      #$TTYIN,R3  ::GET ADDRESS
10129 053470 022703 053745      2$:      CMP      #$TTYIN+25.,R3 ::BUFFER FULL?
10130 053474 101456      BLOS     4$            ::BR IF YES
10131 053476 104402      RDCHR   ::GO READ ONE CHARACTER FROM THE TTY
10132 053500 112613      MOVB    (SP)+,(R3)    ::GET CHARACTER
10133 053502 122713 000177      10$:    CMPB    #177,(R3)    ::IS IT A RUBOUT
10134 053506 001022      BNE     5$            ::BR IF NO
10135 053510 005716      TST     (SP)         ::IS THIS THE FIRST RUBOUT?
10136 053512 001007      BNE     6$            ::BR IF NO
10137 053514 112737 000134 053712      MOVB    #'\\,9$      ::TYPE A BACK SLASH
10138 053522 104401 053712      TYPE    ,9$
10139 053526 012716 177777      MOV     #-1,(SP)     ::SET THE RUBOUT KEY
10140 053532 005303      6$:      DEC     R3           ::BACKUP BY ONE
10141 053534 020327 053714      CMP     R3,$$TTYIN  ::STACK EMPTY?
10142 053540 103434      BLO     4$            ::BR IF YES
10143 053542 111337 053712      MOVB    (R3),9$     ::SETUP TO TYPEOUT THE DELETED CHAR.
10144 053546 104401 053712      TYPE    ,9$
10145 053552 000746      BR      2$            ::GO TYPE
10146 053554 005716      5$:      TST     (SP)         ::GO READ ANOTHER CHAR.
10147 053556 001406      BEQ     7$            ::RUBOUT KEY SET?
10148 053560 112737 000134 053712      MOVB    #'\\,9$      ::BR IF NO
10149 053566 104401 053712      TYPE    ,9$
10150 053572 005016      CLR     (SP)         ::TYPE A BACK SLASH
10151 053574 122713 000025      7$:      CMPB    #25,(R3)    ::CLEAR THE RUBOUT KEY
10152 053600 001003      BNE     8$            ::IS CHARACTER A CTRL U?
10153 053602 104401 053745      TYPE    ,SCNTLU     ::BR IF NO
10154 053606 000726      BR      1$            ::TYPE A CONTROL 'U'
10155 053610 122713 000022      8$:      CMPB    #22,(R3)    ::GO START OVER
10156 053614 001011      BNE     3$            ::IS CHARACTER A "'R'?
10157 053616 105013      CLRB   (R3)         ::BRANCH IF NO
10158 053620 104401 001165      TYPE    ,SCRLF      ::CLEAR THE CHARACTER
10159 053624 104401 053714      TYPE    ,$$TTYIN   ::TYPE A "CR" & "LF"
10160 053630 000717      BR      2$            ::TYPE THE INPUT STRING
10161 053632 104401 001164      4$:      TYPE    ,QUES      ::GO PICKUP ANOTHER CHACTER
10162 053636 000712      BR      1$            ::TYPE A '?'
10163 053640 111337 053712      3$:      MOVB    (R3),9$     ::CLEAR THE BUFFER AND LOOP
                               ::ECHO THE CHARACTER

```

10164	053644	104401	053712		TYPE	9\$	
10165	053650	122723	000015		CMPB	#15,(R3)+	::CHECK FOR RETURN
10166	053654	001305			BNE	2\$::LOOP IF NOT RETURN
10167	053656	105063	177777		CLRB	-1(R3)	::CLEAR RETURN (THE 15)
10168	053662	104401	001166		TYPE	,SLF	::TYPE A LINE FEED
10169	053666	005726			TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK
10170	053670	012603			MOV	(SP)+,R3	::RESTORE R3
10171	053672	011646			MOV	(SP),-(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE
10172	053674	016666	000004	000002	MOV	4(SP),2(SP)	:: FIRST ASCII CHARACTER ON IT
10173	053702	012766	053714	000004	MOV	#STTYIN,4(SP)	
10174	053710	000002			RTI		::RETURN
10175	053712	000			9\$: .BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE
10176	053713	000			.BYTE	0	::TERMINATOR
10177	053714	000031			\$TTYIN: .BLKB	25.	::RESERVE 25. BYTES FOR TTY INPUT
10178	053745	136	006525	000012	\$CNTLU: .ASCIZ	/'^U/<15><12>	::CONTROL 'U'
10179	053752	043536	005015	000	\$CNTLG: .ASCIZ	/'^G/<15><12>	::CONTROL 'G'
10180	053757	015	051412	051127	\$MSWR: .ASCIZ	<15><12>/SWR = /	
10181	053764	036440	000040				
10182	053770	020040	042516	020127	\$MNEW: .ASCIZ	/ NEW = /	
10183	053776	020075	000				
10184		054002			.EVEN		
10185					.SBTTL	TK INITIALIZE ROUTINE	
10186							
10187							
10188					::*****		
10189					::*		
10190					::*	THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD TO ALL THE	
10191					::*	OPERATOR TO COMMUNICATE WITH THE PROGRAM.	
10192					::*		
10193					::*CALL		
10194					::*	JSR PC,\$TKINT	
10195					::*	RETURN	
10196					::*		
10197					::*****		
10198							
10199	054002	012737	054032	000060	\$TKINT: MOV	#\$TKSRV,TKVEC	::INITIALIZE THE KEY BOARD VECTOR
10200	054010	013737	001452	000062	MOV	RKPRI,TKVEC+2	::LOCK RK06 AND TTY INTERUPTS
10201	054016	005777	125124		TST	@\$TKB	::CLEAR DONE FLAG
10202	054022	012777	000100	125114	MOV	#IE,@\$TKS	::ENABLE INTERRUPT
10203	054030	000207			RTS	PC	::RETURN
10204							
10205					.SBTTL	TK SERVICE ROUTINE (ONLY CONTROL C IS REGNIZED)	
10206							
10207	054032	117737	125110	054374	\$TKSRV: MOV	@\$TKB,2\$::PICK UP CHARACTER AND STORE FOR
10208							:: PRINT OUT
10209	054040	142737	000200	054374	BICB	#BIT7,2\$::STRIP THE JUNK
10210	054046	122737	000003	054374	CMPB	#3,2\$::IS IT A CONTROL-C
10211	054054	001022			BNE	1\$::NO, PRINT <CHAR>?<15><12>
10212	054056	104401	054401		TYPE	,SCNTLC	::TYPE A CONTROL-C
10213	054062	005077	125056		CLR	@\$TKS	::CLEAR INTERRUPT ENABLE
10214	054066	016637	000002	177776	MOV	2(SP),PS	::REINSTATE PREVIOUS PSW
10215	054074	004737	011656		JSR	PC,OPRCMD	::PROCESS OPERATOR COMMAND
10216	054100	013737	001452	177776	MOV	RKPRI,PS	::LOAD PSW TO LOCK OUT TTY AND RK06
10217	054106	005777	125034		TST	@\$TKB	::CLEAR DONE FLAG
10218	054112	012777	000100	125024	MOV	#IE,@\$TKS	::ENABLE INTERRUPTS
10219	054120	000002			RTI		::RETURN

10220										
10221	054122	122737	000007	054374	1\$:	CMPB	#7,2\$:CHECK IF CONTROL G	
10222	054130	001403				BEQ	5\$:YES, GET NEW SWITCH REGISTER VALUE	
10223	054132	104401	054374			TYPE	.2\$:PRINT CHARACTER ?<15><12>	
10224	054136	000002			3\$:	RTI			:RETURN	
10225										
10226	054140	022737	000176	001140	5\$:	CMP	#SWREG,SWR		:CHECK IF SOFTWARE SWITCH REG	
10227	054146	001373				BNE	3\$:NO, RETURN	
10228	054150	122737	000001	001134		CMPB	#1,\$AUTOB		:CHECK IF IN AUTO ACCEPT MODE	
10229	054156	001767				BEQ	3\$:YES, RETURN	
10230	054160	104401	053752			TYPE	,\$CNTLG		:TYPE CONTROL G	
10231	054164	104401	053757			TYPE	,\$MSWR		:TYPE OLD SWITCH REGISTER	
10232	054170	013746	000176			MOV	SWREG,-(SP)			
10233	054174	004737	053224			JSR	PC,BINOCT			
10234	054200	104401	053770			TYPE	,\$MNEW			
10235	054204	005077	124734			CLR	@STKS		:GO INTO FLAG MODE	
10236	054210	005046			19\$:	CLR	-(SP)		::CLEAR COUNTER	
10237	054212	005046				CLR	-(SP)		::THE NEW SWR	
10238	054214	105777	124724		7\$:	TSTB	@STKS		::CHAR THERE?	
10239	054220	100375				BPL	7\$::IF NOT TRY AGAIN	
10240										
10241	054222	117746	124720			MOVB	@STKB,-(SP)		::PICK UP CHAR	
10242	054226	042716	177600			BIC	#^C177,(SP)		::MAKE IT 7-BIT ASCII	
10243										
10244	054232	021627	000025		9\$:	CMP	(SP),#25		::IS IT A CONTROL-U?	
10245	054236	001005				BNE	10\$::BRANCH IF NOT	
10246	054240	104401	053745			TYPE	,\$CNTLU		::YES, ECHO CONTROL-U (^U)	
10247	054244	062706	000006		20\$:	ADD	#6,SP		::IGNORE PREVIOUS INPUT	
10248	054250	000757				BR	19\$::LET'S TRY IT AGAIN	
10249										
10250										
10251	054252	021627	000015		10\$:	CMP	(SP),#15		::IS IT A <CR>?	
10252	054256	001016				BNE	16\$::BRANCH IF NO	
10253	054260	005766	000004			TST	4(SP)		::YES, IS IT THE FIRST CHAR?	
10254	054264	001403				BEQ	11\$::BRANCH IF YES	
10255	054266	016677	000002	124644		MOV	2(SP),@SWR		::SAVE NEW SWR	
10256	054274	062706	000006		11\$:	ADD	#6,SP		::CLEAR UP STACK	
10257	054300	104401	001165		14\$:	TYPE	,\$CRLF		::ECHO <CR> AND <LF>	
10258	054304	012777	000100	124632		MOV	#100,@STKS		::RE-ENABLE TTY KBD INTERRUPTS	
10259	054312	000002			15\$:	RTI			::RETURN	
10260	054314	004737	053102		16\$:	JSR	PC,\$TYPEC		::CHO CHAR	
10261	054320	021627	000060			CMP	(SP),#60		::CHAR < 0?	
10262	054324	002420				BLT	18\$::BRANCH IF YES	
10263	054326	021627	000067			CMP	(SP),#67		::CHAR > 7?	
10264	054332	003015				BGT	18\$::BRANCH IF YES	
10265	054334	042726	000060			BIC	#60,(SP)+		::STRIP-OFF ASCII	
10266	054340	005766	000002			TST	2(SP)		::IS THIS THE FIRST CHAR	
10267	054344	001403				BEQ	17\$::BRANCH IF YES	
10268	054346	006316				ASL	(SP)		::NO, SHIFT PRESENT	
10269	054350	006316				ASL	(SP)		::CHAR OVER TO MAKE	
10270	054352	006316				ASL	(SP)		::ROOM FOR NEW ONE.	
10271	054354	005266	000002		17\$:	INC	2(SP)		::KEEP COUNT OF CHAR	
10272	054360	056616	177776			BIS	-2(SP),(SP)		::SET IN NEW CHAR	
10273	054364	000713				BR	7\$::GET THE NEXT ONE	
10274	054366	104401	001164		18\$:	TYPE	,\$QUES		::TYPE ?<CR><LF>	
10275	054372	000724				BR	20\$::SIMULATE CONTROL-U	

```
10276  
10277 054374 000 2$: .BYTE 0 ;STORAGE FOR QUESTIONABLE INPUT  
10278 054375 077 005015 000 .ASCIZ '/?/<15><12>  
10279 054401 136 006503 000012 $CNTLC: .ASCIZ '/^C/<15><12> ;CONTROL-C  
10280  
10281 .SBTTL OCTAL TO BINARY CONVERSION ROUTINE  
10282  
10283 :*****  
10284 :  
10285 : THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED  
10286 : WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL  
10287 : IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS  
10288 : ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.  
10289 :  
10290 :*CALL  
10291 :* MOV <ADDRESS OF ASCII STRING>,-(SP)  
10292 :* JSR PC,OCTBIN  
10293 :* <ADDRESS OF ERROR RETURN>  
10294 :* RETURN  
10295 :*  
10296 :*****  
10297  
10298 054406 010046 OCTBIN: MOV R0,-(SP) ;SAVE R0  
10299 054410 010146 MOV R1,-(SP) ;SAVE R1  
10300 054412 010246 MOV R2,-(SP) ;SAVE R2  
10301 054414 016600 000010 MOV 10(SP),R0 ;GET ADDRESS OF ASCII STRING  
10302 054420 005001 CLR R1 ;CLEAR DATA WORDS  
10303 054422 005002 CLR R2  
10304 054424 112046 2$: MOV (R0)+,-(SP) ;PICK THIS CHARACTER  
10305 054426 001423 BEQ 3$ ;IF ZERO GET OUT  
10306 054430 121627 000054 CMPB (SP),#', ;CHECK IF COMMA  
10307 054434 001420 BEQ 3$ ;IF COMMA GET OUT  
10308 054436 122716 000060 CMPB #'0,(SP) ;MAKE SURE THIS CHARACTER IS  
10309 054442 003030 BGT 4$ ; AN OCTAL DIGIT  
10310 054444 122716 000067 CMPB #'7,(SP)  
10311 054450 002425 BLT 4$  
10312 054452 006301 ASL R1 ; *2  
10313 054454 006102 ROL R2  
10314 054456 006301 ASL R1 ; *4  
10315 054460 006102 ROL R2  
10316 054462 006301 ASL R1 ; *8  
10317 054464 006102 ROL R2  
10318 054466 042716 177770 BIC #'C7,(SP) ;STRIP THE ASCII JUNK  
10319 054472 062601 ADD (SP)+,R1 ;ADD THIS DIGIT  
10320 054474 000753 BR 2$ ;LOOP  
10321 054476 005726 3$: TST (SP)+ ;CLEAN PARTIAL FROM STACK  
10322 054500 010166 000010 MOV R1,10(SP) ;SAVE RESULT  
10323 054504 010237 054540 MOV R2,$HIOCT  
10324 054510 012602 MOV (SP)+,R2 ;RESTORE R2  
10325 054512 012601 MOV (SP)+,R1 ;RESTORE R1  
10326 054514 012600 MOV (SP)+,R0 ;RESTORE R0  
10327 054516 062716 000002 ADD #2,(SP) ;ADJUST RETURN  
10328 054522 000207 RTS PC ;RETURN  
10329  
10330 054524 005726 4$: TST (SP)+ ;CLEAN UP PARTIAL FROM STACK  
10331 054526 012602 MOV (SP)+,R2 ;RESTORE R2
```

10332 054530 012601
10333 054532 012600
10334 054534 013616
10335 054536 000207
10336 054540 000000
10337
10338
10339
10340
10341
10342
10343
10344
10345
10346
10347
10348
10349
10350
10351
10352
10353 054542 010046
10354 054544 010146
10355 054546 010246
10356 054550 016600 000010
10357 054554 005046
10358 054556 005002
10359 054560 122710 000055
10360 054564 001001
10361 054566 112002
10362 054570 112001
10363 054572 001427
10364 054574 120127 000054
10365 054600 001424
10366 054602 122701 000060
10367 054606 003034
10368 054610 122701 000071
10369 054614 002431
10370 054616 032716 170000
10371 054622 001026
10372 054624 006316
10373 054626 011646
10374 054630 006316
10375 054632 006316
10376 054634 062616
10377 054636 102420
10378 054640 162701 000060
10379 054644 060116
10380 054646 102414
10381 054650 000747
10382 054652 005702
10383 054654 001401
10384 054656 005416
10385 054660 012666 000010
10386 054664 012602
10387 054666 012601

```
MOV (SP)+,R1 :RESTORE R1
MOV (SP)+,R0 :RESTORE R0
MOV @ (SP)+,(SP) :PUT ADDRESS OF ERROR ROUTINE ON STACK
RTS PC :GO PROCESS ERROR
$HIOCT: .WORD 0 :HIGH ORDER BITS GO HERE
.SBTTL DECIMAL TO BINARY CONVERSION ROUTINE
```

: THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
: WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL,
: IT WILL GENERATE A BINARY WORD PLACING IT ON THE STACK.

```
*CALL  
: MOV <ADDRESS OF ASCII STRING>,-(SP)  
: JSR PC,DECBIN  
: <ADDRESS OF ERROR RETURN>  
: RETURN
```

```
DECBIN: MOV R0,-(SP) :SAVE R0
MOV R1,-(SP) :SAVE R1
MOV R2,-(SP) :SAVE R2
MOV 10(SP),R0 :GET ADDRESS OF ASCII STRING
CLR -(SP) :CLEAR DATA WORD
CLR R2 :SIGN SET POSITIVE
CMPB #'-(R0) :SEE IF A MINUS SIGN
BNE 2$ :BRANCH IF NO MINUS SIGN
MOVB (R0)+,R2 :SAVE FOR LATER USE
2$: MOVB (R0)+,R1 :PICKUP THIS CHARACTER
BEQ 3$ :GET OUT IF ZERO
CMPB R1,#' :CHECK IF COMMA
BEQ 3$ :GET OUT IF COMMA
CMPB #'0,R1 :MAKE SURE THIS CHARACTER IS
BGT 5$ : A DIGIT BETWEEN 0 & 9
CMPB #'9,R1
BLT 5$
BIT #170000,(SP) :DON'T LET NUMBER GET TO BIG
BNE 5$ :BRANCH IF NUMBER WOULD OVERFLOW
ASL (SP) : *2
MOV (SP),-(SP) :SAVE FOR LATER
ASL (SP) : *4
ASL (SP) : *8
ADD (SP)+,(SP) : *10
BVS 5$ :OVERFLOW ISN'T ALLOWED
SUB #'0,R1 :STRIP AWAY THE ASCII JUNK
ADD R1,(SP) :ADD IN THIS DIGIT
BVS 5$ :OVERFLOW ISN'T ALLOWED
BR 2$ :LOOP
3$: TST R2 :CHECK IF NUMBER IS NEGATIVE
BEQ 4$ :BRANCH IF NO
NEG (SP) :YES--NEGATE THE NUMBER
4$: MOV (SP)+,10(SP) :SAVE RESULT
MOV (SP)+,R2 :RESTORE R2
MOV (SP)+,R1 :RESTORE R1
```

```
10388 054670 012600          MOV    (SP)+,R0      ;RESTORE R0
10389 054672 062716 000002    ADD    #2,(SP)       ;ADJUST RETURN
10390 054676 000207          RTS     PC           ;RETURN
10391
10392 054700 005726          5$:   TST    (SP)+      ;CLEAN PARTIAL NUMBER FROM STACK
10393 054702 012602          MOV    (SP)+,R2     ;RESTORE R2
10394 054704 012601          MOV    (SP)+,R1     ;RESTORE R1
10395 054706 012600          MOV    (SP)+,R0     ;RESTORE R0
10396 054710 013616          MOV    @ (SP)+,(SP) ;PUT ADDRESS OF ERROR ON STACK
10397 054712 000207          RTS     PC           ;GO PROCESS ERROR
10398
10399          .SBTTL  ROUTINE TO SIZE MEMORY
10400
10401          ;*****
10402          ;*CALL:
10403          ;*   JSR    PC,$SIZE
10404          ;*   RETURN
10405          ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
10406 054714 010046          $SIZE: MOV    R0,-(SP)    ;;SAVE R0 ON THE STACK
10407 054716 010146          MOV    R1,-(SP)    ;;SAVE R1 ON THE STACK
10408 054720 013746 000114    MOV    @#114,-(SP) ;;SAVE MEMORY ERROR VECTOR PS & PC
10409 054724 013746 000116    MOV    @#116,-(SP)
10410 054730 012737 000116 000114    MOV    #116,@#114  ;;IGNORE PARITY ERRORS WHILE SIZING
10411 054736 012737 000002 000116    MOV    #RTI,@#116
10412 054744 013746 000004    MOV    @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
10413 054750 013746 000006    MOV    @#ERRVEC+2,-(SP)
10414 054754 010600          MOV    SP,R0        ;;SAVE THE STACK POINTER
10415          ;;SET THE ERRVEC PS TO THE PRESENT PS
10416 054756 104400          TRAP   ;PUSH OLD PSW AND PC ON STACK
10417 054760 012637 000006    MOV    (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
10418 054764 012737 055004 000004    MOV    #2,@#ERRVEC  ;;SET FOR TIMEOUT
10419 054772 012701 020000    MOV    #2000,R1     ;;FIRST ADDRESS
10420 054776 005711          1$:   TST    (R1)         ;;TEST THIS ADDRESS
10421 055000 005721          TST    (R1)+        ;;STEP TO NEXT ADDRESS
10422 055002 000775          BR     1$           ;;TRY ANOTHER
10423 055004 162701 000002          2$:   SUB    #2,R1        ;;DROP BACK
10424 055010 010006          MOV    R0,SP        ;;RESTORE THE STACK
10425 055012 012637 000006    MOV    (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
10426 055016 012637 000004    MOV    (SP)+,@#ERRVEC
10427 055022 012637 000116    MOV    (SP)+,@#116  ;;RESTORE MEMORY ERROR VECTOR
10428 055026 012637 000114    MOV    (SP)+,@#114
10429 055032 010137 055044    MOV    R1,$LSTAD   ;;LAST ADDRESS
10430 055036 012601          MOV    (SP)+,R1     ;;RESTORE R1
10431 055040 012600          MOV    (SP)+,R0     ;;RESTORE R0
10432 055042 000207          RTS     PC
10433 055044 000000          $LSTAD: .WORD 0     ;;CONTAINS THE LAST ADDRESS
10434
10435          ;*****
10436          .SBTTL  POWER DOWN AND UP ROUTINES
10437
10438          ;POWER DOWN ROUTINE
10439 055046 017737 124066 001404    $PWRDN: MOV    @SWR,SAVSWR ;SAVE SWITCH REGISTER
10440 055054 012737 055066 000024    MOV    # $PWRUP,PWRVEC ;SET UP VECTOR
10441 055062 000000          HALT
10442 055064 000776          BR     -2           ;HANG UP
10443
```

```
10444      :POWER UP ROUTINE
10445 055066 005037 055172      $PWRUP: CLR      $PWRCT      ;WAIT LOOP FOR THE TTY
10446 055072 012737 000144 055174      MOV      #100,$PWRCT+2
10447 055100 005237 055172      1$: INC      $PWRCT      ;WAIT FOR THE INCREMENT
10448 055104 001375                BNE      1$              ; OF WORD
10449 055106 005337 055174      DEC      $PWRCT+2
10450 055112 001372                BNE      1$
10451 055114 012737 055046 000024      MOV      #$PWRDN,PWRVEC ;SET POWER DOWN VECTOR
10452 055122 012737 000340 000026      MOV      #PR7,PWRVEC+2 ;PRIORITY 7
10453 055130 012737 000340 000036      MOV      #PR7,TRAPVEC+2 ;LOCK OUT ALL INTERRUPTS FOR TRAPS
10454 055136 012706 001100                MOV      #STACK,SP      ;INITIALIZE STACK
10455 055142 104401 055176                TYPE     $POWER         ;REPORT POWER FAIL
10456 055146 004737 044046                JSR      PC,PRITIM      ;PRINT TIME
10457 055152 112737 177777 001361      MOV      #-1,FLAG      ;SET FLAG FOR RESTART SEQUENCE
10458 055160 013777 001404 123752      MOV      SAVSWR,@SWR    ;RESTORE SWITCH REGISTER
10459 055166 000137 007200                JMP      PWSRT          ;RESTART PROGRAM
10460
10461 055172 000000 000000      SPWRCT: .WORD 0,0      ;WAIT COUNT FOR TTY
10462 055176 005015 047520 042527      $POWER: .ASCIZ <15><12>/POWER/<15><12>
10463 055204 006522 000012
10464
10465      .EVEN
10466      .SBTTL INTEGER MULTIPLY ROUTINE
10467
10468      ;*****
10469      ;*
10470      ;*CALL
10471      ;*      MOV      MULTIPLER,-(SP)
10472      ;*      MOV      MULTIPLICAND,-(SP)
10473      ;*      JSR      PC,$MULT
10474      ;*      RETURN
10475      ;*
10476      ;*      STACK  PRODUCT
10477      ;*      -----
10478      ;*
10479      ;*      TOP    LSB'S
10480      ;*      +2    MSB'S
10481      ;*
10482      ;*****
10483
10484      $MULT:
10485 055210 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
10486 055212 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
10487 055214 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
10488 055216 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
10489 055220 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
10490 055222 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
10491 055224 016605 000016      MOV      16(SP),R5     ;STORE MULTIPLICAND
10492 055230 016603 000020      MOV      20(SP),R3     ;STORE MULTIPLIER
10493 055234 005000                CLR      R0            ;CLEAR HIGH ORDER WORDS
10494 055236 005001                CLR      R1
10495 055240 005002                CLR      R2
10496 055242 005004                CLR      R4
10497 055244 012746 000041      1$: MOV      #41,-(SP)   ;MOVE 33 DEC TO COUNTER
10498 055250 006000                ROR     R0
10499 055252 006001                ROR     R1
```

```

10500 055254 006002          ROR      R2          :SHIFT TO ADD
10501 055256 006003          ROR      R3
10502 055260 103003          BCC      2$          :NO CARRY NO ADD
10503 055262 060501          ADD      R5,R1
10504 055264 005500          ADC      R0          :ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
10505 055266 060400          ADD      R4,R0      :PRODUCT
10506 055270 005316          2$: DEC      (SP)      :DECREMENT COUNTER
10507 055272 001366          BNE      1$
10508 055274 005726          TST      (SP)+
10509 055276 010266 000020  MOV      R2,20(SP)   :REMOVE COUNTER
10510 055302 010366 000016  MOV      R3,16(SP)   :GET LEAST SIGNIFICANT BITS
10511 055306 012605          MOV      (SP)+,R5    :GET MOST SIGNIFICANT BITS
10512 055310 012604          MOV      (SP)+,R4    :POP STACK INTO R5
10513 055312 012603          MOV      (SP)+,R3    :POP STACK INTO R4
10514 055314 012602          MOV      (SP)+,R2    :POP STACK INTO R3
10515 055316 012601          MOV      (SP)+,R1    :POP STACK INTO R2
10516 055320 012600          MOV      (SP)+,R0    :POP STACK INTO R1
10517 055322 000207          RTS      PC          :POP STACK INTO R0
10518
10519
10520
10521
10522
10523
10524
10525
10526
10527
10528
10529
10530
10531
10532
10533
10534
10535
10536
10537
10538
10539
10540
10541
10542
10543
10544
10545
10546
10547
10548
10549
10550
10551
10552
10553
10554
10555

```

.SBTTL RANDOM NUMBER GENERATOR ROUTINE

```

:*****
:*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
:*WITH A RANGE OF 0 TO 2(+33)-1.
:*CALL:
:*      JSR      PC,$RAND      ::CALL THE ROUTINE
:*      RETURN                    ::RETURN HERE THE RANDOM
:*                                ::NUMBER WILL BE IN
:*                                ::$HINUM,$LONUM

```

```

$RAND:
MOV      R0,-(SP)      ::PUSH R0 ON STACK
MOV      R1,-(SP)      ::PUSH R1 ON STACK
MOV      R2,-(SP)      ::PUSH R2 ON STACK
MOV      $LONUM,R0     ::SET R0 WITH LOW
MOV      $HINUM,R1     ::SET R1 WITH HIGH
MOV      #-7,R2        ::SET SHIFT COUNT
1$: ASL      R0          ::SHIFT R0 LEFT AND
ROL      R1            ::ROTATE CARRY INTO R1 AND
INC      R2            ::CHECK FOR DONE
BNE      1$           ::CONTINUE SHIFT LOOP
ADD      $LONUM,R0     ::ADD NUMBER TO MAKE X 129
ADC      R1            ::PROPOGATE CARRY
ADD      $HINUM,R1     ::ADD NUMBER TO MAKE X 129
ADD      #1057,R0      ::ADD LOW CONSTANT
ADC      R1            ::PROPOGATE CARRY
ADD      #47401,R1     ::ADD HIGH CONSTANT
MOV      R0,$LONUM     ::SAVE R0
MOV      R1,$HINUM     ::SAVE R1
MOV      (SP)+,R2      ::POP STACK INTO R2
MOV      (SP)+,R1      ::POP STACK INTO R1
MOV      (SP)+,R0      ::POP STACK INTO R0
RTS      PC            ::RETURN

```

```

$HINUM: .WORD 176543
$LONUM: .WORD 123456

```

10556
10557
10558
10559
10560
10561
10562
10563
10564
10565
10566
10567
10568
10569
10570
10571
10572
10573
10574 055426
10575 055426 010046
10576 055430 010146
10577 055432 010246
10578 055434 010346
10579 055436 010446
10580 055440 010546
10581 055442 016601 000016
10582 055446 005002
10583 055450 012737 000005 055574
10584 055456 012704 055652
10585 055462 012705 055654
10586 055466 000421
10587
10588 055470
10589 055470 010046
10590 055472 010146
10591 055474 010246
10592 055476 010346
10593 055500 010446
10594 055502 010546
10595 055504 016602 000016
10596 055510 012201
10597 055512 011202
10598 055514 012737 000012 055574
10599 055522 012704 055626
10600 055526 012705 055630
10601
10602 055532 012700 055676
10603 055536 005003
10604 055540 161401
10605 055542 005602
10606 055544 161502
10607 055546 002402
10608 055550 005203
10609 055552 000772
10610
10611 055554 062401

.SBTTL SINGLE/DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINES

```

*****
*
* THESE ROUTINE WIL CONVERT A 16-BIT OR 32-BIT BINARY NUMBER
* TO AN UNSIGNED DECIMAL (ASCII) NUMBER.
*
*CALL MOV #PNTR,-(SP) ;POINTER TO LOW WORD OF BINARY NUMBER
* JSR PC,$DB2D
* RETURN ;DECIMAL NUMBER IN $DECVL
*
*CALL MOV NUM,-(SP)
* JSR PC,$SB2D
* RETURN ;DECIMAL NUMBER IN $DECVL
*****

```

```

$SB2D:
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK
MOV R5,-(SP) ;:PUSH R5 ON STACK
MOV 16(SP),R1 ;:STORE NUMBER IN R1
CLR R2 ;:CLEAR MOST SIGNIFICANT BITS
MOV #5,$DECVL ;:SET UP FOR 5 CONVERSIONS
MOV #5,$TNPW,R4 ;:ADDRESS FOR 5 POWER
MOV #5,$TNPW+2,R5
BR $B2D ;:START CONVERSION

```

```

$DB2D:
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK
MOV R5,-(SP) ;:PUSH R5 ON STACK
MOV 16(SP),R2 ;:PICKUP DATA POINTER
MOV (R2)+,R1 ;:PICKUP BINARY NUMBER
MOV (R2),R2
MOV #10,$DECVL ;:SET UP TO DO 10 CONVERSIONS
MOV #10,$TNPW,R4 ;:ADDRESS OF TEN POWER
MOV #10,$TNPW+2,R5

```

```

$B2D: MOV #5,$DECVL,R0 ;:GET ADDRESS OF "$DECVL" STRING
$2DEC: CLR R3 ;:CLEAR PARTIAL
2$: SUB (R4),R1 ;:SUBTRACT TEN POWER
SBC R2
SUB (R5),R2
BLT 3$ ;:BRANCH IF TEN POWER TOO LARGE
INC R3 ;:ADD 1 TO PARTIAL
BR 2$ ;:LOOP
3$: ADD (R4)+,R1 ;:RESTORE SUBTRACTED VALUE

```

```

10612 055556 005502          ADC      R2
10613 055560 062402          ADD      (R4)+,R2
10614 055562 022525          CMP      (R5)+,(R5)+      :MOVE TO NEXT TEN POWER
10615 055564 052703 000060  BIS      #'0,R3           :CHANGE IT TO ASCII
10616 055570 110320          MOV      R3,(R0)+         :SAVE ITA
10617 055572 005327          DEC      (PC)+           :DONE?
10618 055574 000000          SDECNT: .WORD 0
10619 055576 001357          BNE      $2DEC           :BRANCH IF NO
10620 055600 105020          CLRB    (R0)+           :TERMINATOR
10621 055602 012605          MOV      (SP)+,R5        :POP STACK INTO R5
10622 055604 012604          MOV      (SP)+,R4        :POP STACK INTO R4
10623 055606 012603          MOV      (SP)+,R3        :POP STACK INTO R3
10624 055610 012602          MOV      (SP)+,R2        :POP STACK INTO R2
10625 055612 012601          MOV      (SP)+,R1        :POP STACK INTO R1
10626 055614 012600          MOV      (SP)+,R0        :POP STACK INTO R0
10627 055616 012616          MOV      (SP)+,(SP)      :SETUP STACK FOR RETURN
10628 055620 104401 055676  TYPE     $DECVL
10629 055624 000207          RTS      PC              :RETURN
10630
10631 055626 145000          $TNPWR: 145000           :1.0E09
10632 055630 035632          35632
10633 055632 160400          160400           :1.0E08
10634 055634 002765          2765
10635 055636 113200          113200           :1.0E07
10636 055640 000230          230
10637 055642 041100          041100           :1.0E06
10638 055644 000017          17
10639 055646 103240          103240           :1.0E05
10640 055650 000001          1
10641 055652 023420          $TNPW: 23420           :1.0E04
10642 055654 000000          0
10643 055656 001750          1750             :1.0E03
10644 055660 000000          0
10645 055662 000144          144              :1.0E02
10646 055664 000000          0
10647 055666 000012          12               :1.0E01
10648 055670 000000          0
10649 055672 000001          1                :1.0E00
10650 055674 000000          0
10651 055676 000014          $DECVL: .BLKB 12.      :RESERVE STORAGE FOR ASCII STRING
10652          .SBTTL APT COMMUNICATIONS ROUTINE
10653
10654          :*****
10655 055712 112737 000001 056156 $ATY1: MOV      #1,$FFLG      ;;TO REPORT FATAL ERROR
10656 055720 112737 000001 056154 $ATY3: MOV      #1,$MFLG      ;;TO TYPE A MESSAGE
10657 055726 000403          BR      $ATYC
10658 055730 112737 000001 056156 $ATY4: MOV      #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
10659 055736          $ATYC:
10660 055736 010046          MOV      R0,-(SP)        ;;PUSH R0 ON STACK
10661 055740 010146          MOV      R1,-(SP)        ;;PUSH R1 ON STACK
10662 055742 105737 056154          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
10663 055746 001450          BEQ     5$              ;;IF NOT: BR
10664 055750 122737 000001 001210          CMPB    #APTENV,$ENV     ;;OPERATING UNDER APT?
10665 055756 001031          BNE     3$              ;;IF NOT: BR
10666 055760 132737 000100 001211          BITB    #APTPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
10667 055766 001425          BEQ     3$              ;;IF NOT: BR

```


10668	055770	017600	000004			MOV	@4(SP),R0	::GET MESSAGE ADDR.
10669	055774	062766	000002	000004		ADD	#2,4(SP)	::BUMP RETURN ADDR.
10670	056002	005737	001170		1\$:	TST	\$MSGTYPE	::SEE IF DONE W/ LAST XMISSION?
10671	056006	001375				BNE	1\$::IF NOT: WAIT
10672	056010	010037	001204			MOV	R0,\$MSGAD	::PUT ADDR IN MAILBOX
10673	056014	105720			2\$:	TSTB	(R0)+	::FIND END OF MESSAGE
10674	056016	001376				BNE	2\$	
10675	056020	163700	001204			SUB	\$MSGAD,R0	::SUB START OF MESSAGE
10676	056024	006200				ASR	R0	::GET MESSAGE LGTH IN WORDS
10677	056026	010037	001206			MOV	R0,\$MSGGLT	::PUT LENGTH IN MAILBOX
10678	056032	012737	000004	001170		MOV	#4,\$MSGTYPE	::TELL APT TO TAKE MSG.
10679	056040	000413				BR	5\$	
10680	056042	017637	000004	056066	3\$:	MOV	@4(SP),4\$::PUT MSG ADDR IN JSR LINKAGE
10681	056050	062766	000002	000004		ADD	#2,4(SP)	::BUMP RETURN ADDRESS
10682	056056	013746	177776			MOV	177776,-(SP)	::PUSH 177776 ON STACK
10683	056062	004737	052670			JSR	PC,\$TYPE	::CALL TYPE MACRO
10684	056066	000000			4\$:	.WORD	0	
10685	056070				5\$:			
10686	056070	105737	056156		10\$:	TSTB	\$FFLG	::SHOULD REPORT FATAL ERROR?
10687	056074	001416				BEQ	12\$::IF NOT: BR
10688	056076	005737	001210			TST	\$ENV	::RUNNING UNDER APT?
10689	056102	001413				BEQ	12\$::IF NOT: BR
10690	056104	005737	001170		11\$:	TST	\$MSGTYPE	::FINISHED LAST MESSAGE?
10691	056110	001375				BNE	11\$::IF NOT: WAIT
10692	056112	017637	000004	001172		MOV	@4(SP),\$FATAL	::GET ERROR #
10693	056120	062766	000002	000004		ADD	#2,4(SP)	::BUMP RETURN ADDR.
10694	056126	005237	001170			INC	\$MSGTYPE	::TELL APT TO TAKE ERROR
10695	056132	105037	056156		12\$:	CLRB	\$FFLG	::CLEAR FATAL FLAG
10696	056136	105037	056155			CLRB	\$LFLG	::CLEAR LOG FLAG
10697	056142	105037	056154			CLRB	\$MFLG	::CLEAR MESSAGE FLAG
10698	056146	012601				MOV	(SP)+,R1	::POP STACK INTO R1
10699	056150	012600				MOV	(SP)+,R0	::POP STACK INTO R0
10700	056152	000207				RTS	PC	::RETURN
10701	056154	000				\$MFLG:	.BYTE 0	::MESSG. FLAG
10702	056155	000				\$LFLG:	.BYTE 0	::LOG FLAG
10703	056156	000				\$FFLG:	.BYTE 0	::FATAL FLAG
10704		056160				.EVEN		
10705		000200				APTSIZE=200		
10706		000001				APTENV=001		
10707		000100				APTSPool=100		
10708		000040				APTCSUP=040		
10709						.SBTTL TRAP DECODER		
10710								
10711						::*****		
10712						::*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION		
10713						::*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS		
10714						::*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL		
10715						::*GO TO THAT ROUTINE.		
10716								
10717	056160	010046				\$TRAP: MOV	R0,-(SP)	::SAVE R0
10718	056162	016600	000002			MOV	2(SP),R0	::GET TRAP ADDRESS
10719	056166	005740				TST	-(R0)	::BACKUP BY 2
10720	056170	111000				MOVB	(R0),R0	::GET RIGHT BYTE OF TRAP
10721	056172	006300				ASL	R0	::POSITION FOR INDEXING
10722	056174	016000	056214			MOV	\$TRPAD(R0),R0	::INDEX TO TABLE
10723	056200	000200				RTS	R0	::GO TO ROUTINE

```
10724
10725
10726      ::THIS IS USE TO HANDLE THE "GETPRI" MACRO
10727
10728 056202 011646      STRAP2: MOV      (SP),-(SP)      ::MOVE THE PC DOWN
10729 056204 016666 000004 000002      MOV      4(SP),2(SP)      ::MOVE THE PSW DOWN
10730 056212 000002      RTI      ::RESTORE THE PSW
10731
10732      .SBTTL TRAP TABLE
10733
10734      :*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
10735      :*BY THE "TRAP" INSTRUCTION.
10736
10737      :      ROUTINE
10738      :      -----
10739 056214 056202      $TRPAD: .WORD  $STRAP2
10740 056216 052670      $TYPE  ::CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
10741
10742
10743 056220 053330      $RDCHR ::CALL=RDCHR      TRAP+2(104402) TTY TYPEIN CHARACTER ROUTINE
10744 056222 053460      $RDLIN ::CALL=RDLIN      TRAP+3(104403) TTY TYPEIN STRING ROUTINE
```

```
10745 .SBTTL ASCII MESSAGES
10746
10747 056224 005015 047524 052040 XXDPMG: .ASCII <CR><LF>/TO TEST DRIVE 0,REMOVE XXDP MEDIA,/
10748 056232 051505 020124 051104
10749 056240 053111 020105 026060
10750 056246 042522 047515 042526
10751 056254 054040 042130 020120
10752 056262 042515 044504 026101
10753 056270 005015 046103 040505 .ASCII <CR><LF>/CLEAR LOC 40,AND HIT CONT./
10754 056276 020122 047514 020103
10755 056304 030064 040454 042116
10756 056312 044040 052111 041440
10757 056320 047117 027124
10758 056324 005015 043111 042040 .ASCIIZ <CR><LF>/IF DRIVE 0 ISN'T TO BE TESTED,JUST HIT CONT. ./<CR><LF>
10759 056332 044522 042526 030040
10760 056340 044440 047123 052047
10761 056346 052040 020117 042502
10762 056354 052040 051505 042524
10763 056362 026104 052512 052123
10764 056370 044040 052111 041440
10765 056376 047117 027124 027040
10766 056404 005015 000
10767 056407 111 046114 043505 ILLCOM: .ASCII /ILLEGAL OPERATOR COMMAND /
10768 056414 046101 047440 042520
10769 056422 040522 047524 020122
10770 056430 047503 046515 047101
10771 056436 020104
10772 056440 020040 006477 000012 ILLCMD: .ASCIIZ / ?/<15><12>
10773 056446 051104 053111 020105 OPR000: .ASCII /DRIVE NUMBER /
10774 056454 052516 041115 051105
10775 056462 040
10776 056463 060 000040 OPR001: .ASCIIZ /0 /
10777 056466 046111 042514 040507 OPR002: .ASCIIZ /ILLEGAL?/<15><12>
10778 056474 037514 005015 000
10779 056501 125 042116 051105 OPR003: .ASCIIZ /UNDER TEST/
10780 056506 052040 051505 000124
10781 056514 046101 042522 042101 OPR004: .ASCIIZ /ALREADY ASSIGNED?/<15><12>
10782 056522 020131 051501 044523
10783 056530 047107 042105 006477
10784 056536 000012
10785 056540 047516 020124 051501 OPR005: .ASCIIZ /NOT ASSIGNED?/<15><12>
10786 056546 044523 047107 042105
10787 056554 006477 000012
10788 056560 040510 020123 042502 OPR006: .ASCIIZ /HAS BEEN DROPPED FROM TEST SEQUENCE/<15><12>
10789 056566 047105 042040 047522
10790 056574 050120 042105 043040
10791 056602 047522 020115 042524
10792 056610 052123 051440 050505
10793 056616 042525 041516 006505
10794 056624 000012
10795 056626 046120 040505 042523 OPR007: .ASCIIZ /PLEASE ENTER COMMAND/<15><12> /*/
10796 056634 042440 052116 051105
10797 056642 041440 046517 040515
10798 056650 042116 005015 000052
10799 056656 050117 051105 052101 OPR008: .ASCIIZ /OPERATOR INITIATED DROP DRIVE DURING WRITE PACK/
10800 056664 051117 044440 044516
```

10801	056672	044524	052101	042105	
10802	056700	042040	047522	020120	
10803	056706	051104	053111	020105	
10804	056714	052504	044522	043516	
10805	056722	053440	044522	042524	
10806	056730	050040	041501	000113	
10807	056736	005015	051104	053111	OPR010: .ASCII <15><12>/DRIVE NUMBER /
10808	056744	020105	052516	041115	
10809	056752	051105	040		
10810	056755	060	000040		OPR011: .ASCIZ /0 /
10811	056760	051104	053111	020105	OPR012: .ASCIZ /DRIVE WRITE LOCKED/<15><12>
10812	056766	051127	052111	020105	
10813	056774	047514	045503	042105	
10814	057002	005015	000		
10815	057005	101	046114	042040	OPR013: .ASCIZ /ALL DRIVES CURRENTLY UNDER TEST/<15><12>
10816	057012	044522	042526	020123	
10817	057020	052503	051122	047105	
10818	057026	046124	020131	047125	
10819	057034	042504	020122	042524	
10820	057042	052123	005015	000	
10821	057047	116	020117	051104	OPR014: .ASCIZ /NO DRIVE IN USE/<15><12>
10822	057054	053111	020105	047111	
10823	057062	052440	042523	005015	
10824	057070	000			
10825	057071	101	046114	043511	OPR015: .ASCIZ /ALIGNMENT PACK IN DRIVE/<15><12>
10826	057076	046516	047105	020124	
10827	057104	040520	045503	044440	
10828	057112	020116	051104	053111	
10829	057120	006505	000012		
10830	057124	025052	051104	053111	DRVSTT: .ASCIZ /**DRIVE STATISTICS**/<15><12>
10831	057132	020105	052123	052101	
10832	057140	051511	044524	051503	
10833	057146	025052	005015	000	
10834	057153	015	042012	044522	STT000: .ASCII <15><12>/DRIVE SERIAL NO. /
10835	057160	042526	051440	051105	
10836	057166	040511	020114	047516	
10837	057174	020056			
10838	057176	000003			STT100: .BLKB 3
10839	057201	000			.BYTE 0
10840	057202	051117	042504	051522	STT001: .ASCIZ /ORDERS PERFORMED /
10841	057210	050040	051105	047506	
10842	057216	046522	042105	000040	
10843	057224	005015	047527	042122	STT002: .ASCIZ <15><12>/WORDS WRITTEN*65K /
10844	057232	020123	051127	052111	
10845	057240	042524	025116	032466	
10846	057246	020113	000		
10847	057251	015	053412	051117	STT003: .ASCIZ <15><12>/WORDS READ*65K /
10848	057256	051504	051040	040505	
10849	057264	025104	032466	020113	
10850	057272	000			
10851	057273	015	051412	043117	STT004: .ASCIZ <15><12>/SOFT DATA ERRORS/<15><12>/ ECC /
10852	057300	020124	040504	040524	
10853	057306	042440	051122	051117	
10854	057314	006523	020012	042440	
10855	057322	041503	000040		
10856	057326	005015	020040	042522	STT005: .ASCIZ <15><12>/ REREAD /

10857	057334	042522	042101	000040	
10858	057342	005015	020040	043117	STT006: .ASCIZ <15><12>/ OFFSET /
10859	057350	051506	052105	000040	
10860	057356	005015	040510	042122	STT007: .ASCIZ <15><12>/HARD DATA ERRORS /
10861	057364	042040	052101	020101	
10862	057372	051105	047522	051522	
10863	057400	000040			
10864	057402	005015	042523	045505	STT008: .ASCIZ <15><12>/SEEK INCOMPLETES /
10865	057410	044440	041516	046517	
10866	057416	046120	052105	051505	
10867	057424	000040			
10868	057426	005015	050117	051105	STT009: .ASCIZ <15><12>'OPERATION INCOMPLETES/MISPOSITIONS ''
10869	057434	052101	047511	020116	
10870	057442	047111	047503	050115	
10871	057450	042514	042524	027523	
10872	057456	044515	050123	051517	
10873	057464	052111	047511	051516	
10874	057472	000040			
10875	057474	005015	052117	042510	STT010: .ASCIZ <15><12>/OTHER ERRORS /
10876	057502	020122	051105	047522	
10877	057510	051522	000040		
10878	057514	005015	040520	045503	STT011: .ASCII <15><12>/PACK SERIAL NUMBER /
10879	057522	051440	051105	040511	
10880	057530	020114	052516	041115	
10881	057536	051105	040		
10882	057541	000013			STT012: .BLKB 11.
10883	057554	005015	000		STT013: .ASCIZ <15><12>
10884	057557	103	052520	044440	SYS000: .ASCIZ /CPU ID=/'
10885	057564	036504	000		
10886	057567	117	042526	046122	SYS001: .ASCIZ /OVERLAY LOADER=/'
10887	057574	054501	046040	040517	
10888	057602	042504	036522	000	
10889	057607	115	054101	052040	SYS002: .ASCIZ /MAX TRANSFER SIZE (MAX: /
10890	057614	040522	051516	042506	
10891	057622	020122	044523	042532	
10892	057630	024040	046440	054101	
10893	057636	020072	000		
10894	057641	040	036451	000	SYS003: .ASCIZ /)=/'
10895	057645	116	027117	047440	SYS004: .ASCIZ /NO. OF SOFTWARE COMPARES=/'
10896	057652	020106	047523	052106	
10897	057660	040527	042522	041440	
10898	057666	046517	040520	042522	
10899	057674	036523	000		
10900	057677	122	030113	026466	SYS005: .ASCIZ /RK06-RK07 BUS ADD=/'
10901	057704	045522	033460	041040	
10902	057712	051525	040440	042104	
10903	057720	000075			
10904	057722	045522	033060	051055	SYS006: .ASCIZ /RK06-RK07 VEC ADD=/'
10905	057730	030113	020067	042526	
10906	057736	020103	042101	036504	
10907	057744	000			
10908	057745	116	047117	054105	SYS007: .ASCIZ /NONEXISTENT MEMORY/<15><12>
10909	057752	051511	042524	052116	
10910	057760	046440	046505	051117	
10911	057766	006531	000012		
10912	057772	042522	042101	020131	SYS008: .ASCIZ /READY TO BEGIN PERFORMANCE TESTING/<15><12><12>

10913	060000	047524	041040	043505	
10914	060006	047111	050040	051105	
10915	060014	047506	046522	047101	
10916	060022	042503	052040	051505	
10917	060030	044524	043516	005015	
10918	060036	000012			
10919	060040	052123	052101	051511	SYS009: .ASCIZ /STATISTIC INTERVAL=/ /
10920	060046	044524	020103	047111	
10921	060054	042524	053122	046101	
10922	060062	000075			
10923	060064	045522	033060	051055	SYS010: .ASCIZ /RK06-RK07 PRIOR=/ /
10924	060072	030113	020067	051120	
10925	060100	047511	036522	000	
10926	060105	120	051101	042515	PAR000: .ASCII /PARAMETERS FOR DRIVE / /
10927	060112	042524	051522	043040	
10928	060120	051117	042040	044522	
10929	060126	042526	040		
10930	060131	060	005015	000012	PAR001: .ASCIZ /0/<15><12><12> /
10931	060136	054503	044514	042116	PAR002: .ASCIZ /CYLINDER MAX,MIN =/ /
10932	060144	051105	046440	054101	
10933	060152	046454	047111	036440	
10934	060160	000			
10935	060161	124	040522	045503	PAR003: .ASCIZ /TRACK MAX,MIN =/ /
10936	060166	046440	054101	046454	
10937	060174	047111	036440	000	
10938	060201	123	041505	047524	PAR004: .ASCIZ /SECTOR MAX,MIN =/ /
10939	060206	020122	040515	026130	
10940	060214	044515	020116	000075	
10941	060222	042522	042101	053457	PAR005: .ASCIZ 'READ/WRITE RATIO ='
10942	060230	044522	042524	051040	
10943	060236	052101	047511	036440	
10944	060244	000			
10945	060245	127	044522	042524	PAR006: .ASCIZ /WRITE CHECK AFTER WRITE =/ /
10946	060252	041440	042510	045503	
10947	060260	040440	052106	051105	
10948	060266	053440	044522	042524	
10949	060274	036440	000		
10950	060277	103	051117	042522	PAR007: .ASCIZ /CORRECTABLE READ ERROR THRESHOLD =/ /
10951	060304	052103	041101	042514	
10952	060312	051040	040505	020104	
10953	060320	051105	047522	020122	
10954	060326	044124	042522	044123	
10955	060334	046117	020104	000075	
10956	060342	047125	047503	051122	PAR008: .ASCIZ /UNCORRECTABLE READ ERROR THRESHOLD =/ /
10957	060350	041505	040524	046102	
10958	060356	020105	042522	042101	
10959	060364	042440	051122	051117	
10960	060372	052040	051110	051505	
10961	060400	047510	042114	036440	
10962	060406	000			
10963	060407	123	042505	020113	PAR009: .ASCIZ /SEEK ERROR THRESHOLD =/ /
10964	060414	051105	047522	020122	
10965	060422	044124	042522	044123	
10966	060430	046117	020104	000075	
10967	060436	047111	044510	044502	PAR010: .ASCIZ /INHIBIT ERROR CORRECTION AND RETRY =/ /
10968	060444	020124	051105	047522	

10969	060452	020122	047503	051122	
10970	060460	041505	044524	047117	
10971	060466	040440	042116	051040	
10972	060474	052105	054522	036440	
10973	060502	000			
10974	060503	117	042520	040522	PAR011: .ASCIZ /OPERATION COUNT THRESHOLD*65K =/
10975	060510	044524	047117	041440	
10976	060516	052517	052116	052040	
10977	060524	051110	051505	047510	
10978	060532	042114	033052	045465	
10979	060540	036440	000		
10980	060543	127	051117	051504	PAR012: .ASCIZ /WORDS TRANSFERRED THRESHOLD*1,049K =/
10981	060550	052040	040522	051516	
10982	060556	042506	051122	042105	
10983	060564	052040	051110	051505	
10984	060572	047510	042114	030452	
10985	060600	030054	034464	020113	
10986	060606	000075			
10987	060610	044506	051522	000124	PAR013: .ASCIZ /FIRST/
10988	060616	042523	047503	042116	PAR014: .ASCIZ /SECOND/
10989	060624	000			
10990	060625	124	044510	042122	PAR015: .ASCIZ /THIRD/
10991	060632	000			
10992	060633	106	052517	052122	PAR016: .ASCIZ /FOURTH/
10993	060640	000110			
10994	060642	044506	052106	000110	PAR017: .ASCIZ /FIFTH/
10995	060650	042440	041530	052514	PAR018: .ASCIZ / EXCLUDED PACK AREA =/
10996	060656	042504	020104	040520	
10997	060664	045503	040440	042522	
10998	060672	020101	000075		
10999	060676	044103	047101	042507	PAR019: .ASCIZ /CHANGE/
11000	060704	000			
11001	060705	125	040516	046102	PAR020: .ASCII /UNABLE TO GENERATE NEW COMMAND DUE TO/<15><12>
11002	060712	020105	047524	043440	
11003	060720	047105	051105	052101	
11004	060726	020105	042516	020127	
11005	060734	047503	046515	047101	
11006	060742	020104	052504	020105	
11007	060750	047524	005015		
11008	060754	054105	046103	042125	.ASCIZ /EXCLUDED PACK AREAS AND MAX TRANSFER SIZE CONFLICT/<15><12>
11009	060762	042105	050040	041501	
11010	060770	020113	051101	040505	
11011	060776	020123	047101	020104	
11012	061004	040515	020130	051124	
11013	061012	047101	043123	051105	
11014	061020	051440	055111	020105	
11015	061026	047503	043116	044514	
11016	061034	052103	005015	000	
11017	061041	123	046501	046120	PAR021: .ASCIZ /SAMPLED COMPARES =/
11018	061046	042105	041440	046517	
11019	061054	040520	042522	020123	
11020	061062	000075			
11021	061064	042524	052123	047111	PAR022: .ASCIZ /TESTING RK06? Y OR N /
11022	061072	020107	045522	033060	
11023	061100	020077	020131	051117	
11024	061106	047040	020040	000	

CZR6PD0 RK611/06 PERF EXEC
CZR6PD.P11 10-FEB-82 08:40

MACY11 30(1046) 23-FEB-82 08:29 PAGE 249
ASCII MESSAGES

SEQ 0246

```
11025 061113 040 037440 037477 QUESMK: .ASCIZ / ?????? /
11026 061120 037477 020077 000040
11027 061126 020040 000
11028 061131 052 025052 040506 BLANKS: .ASCIZ / /
11029 061136 040524 020114 051105 ERR000: .ASCIZ /***FATAL ERROR***/<15><12><12>
11030 061144 047522 025122 025052
11031 061152 005015 000012
11032 061156 047503 052116 047522 ERR001: .ASCIZ /CONTROLLER CLEAR DID NOT CLEAR ERROR/
11033 061164 046114 051105 041440
11034 061172 042514 051101 042040
11035 061200 042111 047040 052117
11036 061206 041440 042514 051101
11037 061214 042440 051122 051117
11038 061222 000
11039 061223 116 020117 052101 ERR002: .ASCIZ /NO ATTENTION IN RKASOF/
11040 061230 042524 052116 047511
11041 061236 020116 047111 051040
11042 061244 040513 047523 000106
11043 061252 047125 054105 042520 ERR003: .ASCIZ /UNEXPECTED DATA TYPE ERROR/
11044 061260 052103 042105 042040
11045 061266 052101 020101 054524
11046 061274 042520 042440 051122
11047 061302 051117 000
11048 061305 104 044522 042526 ERR004: .ASCIZ /DRIVE CLEAR DID NOT RESET ATTENTION/
11049 061312 041440 042514 051101
11050 061320 042040 042111 047040
11051 061326 052117 051040 051505
11052 061334 052105 040440 052124
11053 061342 047105 044524 047117
11054 061350 000
11055 061351 111 046114 043505 ERR005: .ASCIZ /ILLEGAL DRIVER COMMAND/
11056 061356 046101 042040 044522
11057 061364 042526 020122 047503
11058 061372 046515 047101 000104
11059 061400 052515 052114 050111 ERR006: .ASCIZ /MULTIPLE DRIVE SELECT/
11060 061406 042514 042040 044522
11061 061414 042526 051440 046105
11062 061422 041505 000124
11063 061426 042523 020124 053523 ERR007: .ASCIZ /SET SWITCH 14 TO CYCLE ON ERROR/<15><12>
11064 061434 052111 044103 030440
11065 061442 020064 047524 041440
11066 061450 041531 042514 047440
11067 061456 020116 051105 047522
11068 061464 006522 000012
11069 061470 047503 052116 030122 ERR008: .ASCIZ /CONTROLLER ERROR THRESHOLD EXCEEDED/
11070 061476 046114 051105 042440
11071 061504 051122 051117 052040
11072 061512 051110 051505 047510
11073 061520 042114 042440 041530
11074 061526 042505 042504 000104
11075 061534 040504 040524 046040 ERR009: .ASCIZ /DATA LATE THRESHOLD EXCEEDED/
11076 061542 052101 020105 044124
11077 061550 042522 044123 046117
11078 061556 020104 054105 042503
11079 061564 042105 042105 000
11080 061571 101 042104 042522 ERR012: .ASCII /ADDRESS GOOD BAD SECTOR DATA/<15><12>
```


11081	061576	051523	043440	047517
11082	061604	020104	020040	041040
11083	061612	042101	020040	020040
11084	061620	051440	041505	047524
11085	061626	020122	042040	052101
11086	061634	006501	012	
11087	061637	040	020040	020040
11088	061644	020040	042040	052101
11089	061652	020101	020040	042040
11090	061660	052101	020101	020040
11091	061666	050040	051517	020040
11092	061674	020040	050040	052101
11093	061702	021440	005015	000
11094	061707	103	047117	051124
11095	061714	046117	042514	020122
11096	061722	051105	047522	020122
11097	061730	047516	020124	046106
11098	061736	043501	042507	006504
11099	061744	000012		
11100	061746	047527	042122	041440
11101	061754	052517	052116	047040
11102	061762	052117	042440	052521
11103	061770	046101	030040	005015
11104	061776	000		
11105	061777	102	051525	040440
11106	062004	042104	044440	041516
11107	062012	051117	042522	052103
11108	062020	005015	000	
11109	062023	103	046131	052054
11110	062030	045522	051454	041505
11111	062036	044440	041516	051117
11112	062044	042522	052103	005015
11113	062052	000		
11114	062053	103	047117	051124
11115	062060	046117	042514	020122
11116	062066	047503	046515	047101
11117	062074	020104	044524	042515
11118	062102	047440	052125	000
11119	062107	120	047522	020107
11120	062114	051105	047522	000122
11121	062122	046111	020114	052506
11122	062130	041516	020124	047503
11123	062136	042504	000	
11124	062141	106		
11125	062142	051105	047522	020122
11126	062150	044127	046111	020105
11127	062156	040527	052111	047111
11128	062164	020107	047524	051040
11129	062172	050105	051117	020124
11130	062200	051105	047522	000122
11131	062206	106		
11132	062207	116	047117	042455
11133	062214	044530	052123	042040
11134	062222	053122	000	
11135	062225	106		
11136	062226	042523	041522	047117

.ASCIZ / DATA DATA POS PAT #/<15><12>

ERR013: .ASCIZ /CONTROLLER ERROR NOT FLAGGED/<15><12>

ERR014: .ASCIZ /WORD COUNT NOT EQUAL 0/<15><12>

ERR015: .ASCIZ /BUS ADD INCORRECT/<15><12>

ERR016: .ASCIZ /CYL,TRK,SEC INCORRECT/<15><12>

ERR017: .ASCIZ /CONTROLLER COMMAND TIME OUT/

ERR020: .ASCIZ /PROG ERROR/

ERR021: .ASCIZ /ILL FUNCT CODE/

ERR022: .BYTE 106
.ASCIZ /ERROR WHILE WAITING TO REPORT ERROR/

ERR023: .BYTE 106
.ASCIZ /NON-EXIST DRV/

ERR024: .BYTE 106
.ASCIZ /SERCON PAR/

11137	062234	050040	051101	000	
11138	062241	106			ERR025: .BYTE 106
11139	062242	047125	052111	043040	.ASCIZ /UNIT FIELD ERR/
11140	062250	042511	042114	042440	
11141	062256	051122	000		
11142	062261	015	044412	052116	ERR026: .ASCIZ <15><12>/INT FROM DRV NOT UNDER TEST/<15><12>
11143	062266	043040	047522	020115	
11144	062274	051104	020126	047516	
11145	062302	020124	047125	042504	
11146	062310	020122	042524	052123	
11147	062316	005015	000		
11148	062321	110	051101	020104	ERR027: .ASCIZ /HARD DRV ERR/<15><12>
11149	062326	051104	020126	051105	
11150	062334	006522	000012		
11151	062340	051104	020126	052123	ERR028: .ASCIZ /DRV STATUS CHANGE DID NOT CLR/<15><12>
11152	062346	052101	051525	041440	
11153	062354	040510	043516	020105	
11154	062362	044504	020104	047516	
11155	062370	020124	046103	006522	
11156	062376	000012			
11157	062400	106			ERR029: .BYTE 106
11158	062401	104	044522	042526	.ASCIZ /DRIVE BECAME NOT AVAIL/
11159	062406	041040	041505	046501	
11160	062414	020105	047516	020124	
11161	062422	053101	044501	000114	
11162	062430	047125	054105	042520	ERR030: .ASCIZ /UNEXPECT ATTN/<15><12>
11163	062436	052103	040440	052124	
11164	062444	006516	000012		
11165	062450	046111	020114	051504	ERR031: .ASCIZ /ILL DSK ADD/
11166	062456	020113	042101	000104	
11167	062464	051127	020124	047514	ERR032: .ASCIZ /WRT LOCK ERR/
11168	062472	045503	042440	051122	
11169	062500	000			
11170	062501	101	020103	047514	ERR033: .ASCIZ /AC LOW/
11171	062506	000127			
11172	062510	050123	042505	020104	ERR034: .ASCIZ /SPEED LOSS/
11173	062516	047514	051523	000	
11174	062523	104	053122	052040	ERR035: .ASCIZ /DRV TYPE ERR/
11175	062530	050131	020105	051105	
11176	062536	000122			
11177	062540	047506	046522	052101	ERR036: .ASCIZ /FORMAT ERR/
11178	062546	042440	051122	000	
11179	062553	116	047117	042455	ERR037: .ASCIZ /NON-EXIST DRV FUNCT/
11180	062560	044530	052123	042040	
11181	062566	053122	043040	047125	
11182	062574	052103	000		
11183	062577	106			ERR038: .BYTE 106
11184	062600	042523	045505	044440	.ASCIZ /SEEK INCOMP/
11185	062606	041516	046517	000120	
11186	062614	106			ERR039: .BYTE 106
11187	062615	104	053122	047440	.ASCIZ /DRV OFF TRK/
11188	062622	043106	052040	045522	
11189	062630	000			
11190	062631	106			ERR040: .BYTE 106
11191	062632	044515	050123	051517	.ASCIZ /MISPOS/
11192	062640	000			

11193	062641	126			ERR042: .BYTE 126
11194	062642	050117	051105	052101	.ASCIZ /OPERAT INCOMP/
11195	062650	044440	041516	046517	
11196	062656	000120			
11197	062660	127			ERR043: .BYTE 127
11198	062661	110	051104	053040	.ASCIZ /HDR VRC ERR/
11199	062666	041522	042440	051122	
11200	062674	000			
11201	062675	116	047117	042455	ERR044: .ASCIZ /NON-EXIST MEM/
11202	062702	044530	052123	046440	
11203	062710	046505	000		
11204	062713	125	044516	052502	ERR045: .ASCIZ /UNIBUS PAR/
11205	062720	020123	040520	000122	
11206	062726	117			ERR046: .BYTE 117
11207	062727	104	053122	052040	.ASCIZ /DRV TIMING ERR/
11208	062734	046511	047111	020107	
11209	062742	051105	000122		
11210	062746	040504	040524	046040	ERR047: .ASCIZ /DATA LATE/<15><12>
11211	062754	052101	006505	000012	
11212	062762	117			ERR048: .BYTE 117
11213	062763	104	052101	020101	.ASCIZ /DATA CHK/
11214	062770	044103	000113		
11215	062774	107			ERR050: .BYTE 107
11216	062775	127	044522	042524	.ASCIZ /WRITE CHK ERR/
11217	063002	041440	045510	042440	
11218	063010	051122	000		
11219	063013	103	046131	040440	ERR051: .ASCIZ /CYL ADDR OVRFLW/
11220	063020	042104	020122	053117	
11221	063026	043122	053514	000	
11222	063033	106			ERR052: .BYTE 106
11223	063034	046503	042116	052040	.ASCIZ /CMND TIME OUT DURING POSITIONING OPERATION/<15><12>
11224	063042	046511	020105	052517	
11225	063050	020124	052504	044522	
11226	063056	043516	050040	051517	
11227	063064	052111	047511	044516	
11228	063072	043516	047440	042520	
11229	063100	040522	044524	047117	
11230	063106	005015	000		
11231	063111	106			ERR053: .BYTE 106
11232	063112	051104	020126	047516	.ASCIZ /DRV NOT RDY AFTER START SPINDLE/
11233	063120	020124	042122	020131	
11234	063126	043101	042524	020122	
11235	063134	052123	051101	020124	
11236	063142	050123	047111	046104	
11237	063150	000105			
11238	063152	106			ERR054: .BYTE 106
11239	063153	126	046117	053040	.ASCIZ /VOL VALID DID NOT SET AFTER PACK ACK/
11240	063160	046101	042111	042040	
11241	063166	042111	047040	052117	
11242	063174	051440	052105	040440	
11243	063202	052106	051105	050040	
11244	063210	041501	020113	041501	
11245	063216	000113			
11246	063220	051104	020126	047125	ERR056: .ASCIZ /DRV UNSAFE/
11247	063226	040523	042506	000	
11248	063233	104	053122	042440	ERR057: .ASCIZ /DRV ERR NOT INDICATED BY FAULT/<15><12>

11249	063240	051122	047040	052117	
11250	063246	044440	042116	041511	
11251	063254	052101	042105	041040	
11252	063262	020131	040506	046125	
11253	063270	006524	000012		
11254	063274	051104	020126	052101	ERR058: .ASCIZ /DRV ATTN BUT NO FAULT OR DRV STATUS CHANGE/<15><12>
11255	063302	047124	041040	052125	
11256	063310	047040	020117	040506	
11257	063316	046125	020124	051117	
11258	063324	042040	053122	051440	
11259	063332	040524	052524	020123	
11260	063340	044103	047101	042507	
11261	063346	005015	000		
11262	063351	103	047115	020104	ERR059: .ASCIZ /CMND TIME OUT DRV SEIZED BY OTHER PORT/<15><12>
11263	063356	044524	042515	047440	
11264	063364	052125	042040	053122	
11265	063372	051440	044505	042532	
11266	063400	020104	054502	047440	
11267	063406	044124	051105	050040	
11268	063414	051117	006524	000012	
11269	063422	041505	020103	044515	ERR060: .ASCIZ /ECC MISCORRECTION/<15><12>
11270	063430	041523	051117	042522	
11271	063436	052103	047511	006516	
11272	063444	000012			
11273	063446	040504	040524	041440	ERR062: .ASCIZ /DATA COMPARE ERROR/<15><12>
11274	063454	046517	040520	042522	
11275	063462	042440	051122	051117	
11276	063470	005015	000		
11277	063473	126			ERR064: .BYTE 126
11278	063474	044515	050123	051517	.ASCIZ /MISPOS/
11279	063502	000			
11280	063503	106			ERR065: .BYTE 106
11281	063504	042510	042101	051440	.ASCIZ /HEAD SELECT ERROR/
11282	063512	046105	041505	020124	
11283	063520	051105	047522	000122	
11284	063526	041505	020103	047514	ERR066: .ASCIZ /ECC LOGIC ERROR/
11285	063534	044507	020103	051105	
11286	063542	047522	000122		
11287	063546	045522	041505	052120	ERR067: .ASCIZ /RKECPT RKECPS/<15><12>
11288	063554	020040	045522	041505	
11289	063562	051520	005015	000	
11290	063567	127	051117	020104	ERR070: .ASCIZ /WORD COUNT INVALID/
11291	063574	047503	047125	020124	
11292	063602	047111	040526	044514	
11293	063610	000104			
11294	063612	052502	020123	042101	ERR071: .ASCIZ /BUS ADDRESS INVALID/
11295	063620	051104	051505	020123	
11296	063626	047111	040526	044514	
11297	063634	000104			
11298	063636	054503	044514	042116	ERR072: .ASCIZ /CYLINDER INVALID/
11299	063644	051105	044440	053116	
11300	063652	046101	042111	000	
11301	063657	124	040522	045503	ERR073: .ASCIZ "TRACK/SECTOR INVALID"
11302	063664	051457	041505	047524	
11303	063672	020122	047111	040526	
11304	063700	044514	000104		

11305	063704	106		
11306	063705	105	051122	051117
11307	063712	044440	020116	051104
11308	063720	020126	044504	020104
11309	063726	047516	020124	042522
11310	063734	047514	042101	044040
11311	063742	040505	051504	000
11312	063747	103	051125	042522
11313	063754	052116	051440	050125
11314	063762	046120	042511	006504
11315	063770	012		
11316	063771	103	047115	020104
11317	063776	020040	041440	046131
11318	064004	020040	020040	052040
11319	064012	045522	020040	020040
11320	064020	051440	041505	020040
11321	064026	020040	047440	043106
11322	064034	042523	020124	041040
11323	064042	051525	040440	020104
11324	064050	053440	042122	041440
11325	064056	000124		
11326	064060	051120	053105	047511
11327	064066	051525	006440	012
11328	064073	103	047115	020104
11329	064100	020040	041440	046131
11330	064106	020040	020040	052040
11331	064114	045522	020040	020040
11332	064122	051440	041505	020040
11333	064130	020040	053440	042122
11334	064136	041440	000124	
11335	064142	020040	040520	000124
11336	064150	043117	051506	052105
11337	064156	036440	000	
11338	064161	110	030504	020040
11339	064166	020040	044040	031104
11340	064174	020040	020040	044040
11341	064202	031504	005015	000
11342	064207	104	052101	020101
11343	064214	040514	042524	053440
11344	064222	042510	020116	047125
11345	064230	047514	042101	047111
11346	064236	020107	042510	042101
11347	064244	051105	000	
11348	064247	103	047117	051124
11349	064254	046117	042514	020122
11350	064262	051105	047522	020122
11351	064270	052504	044522	043516
11352	064276	042040	044522	042526
11353	064304	051440	051105	044526
11354	064312	044503	043516	000
11355	064317	105	051122	051117
11356	064324	042040	051125	047111
11357	064332	020107	051105	047522
11358	064340	020122	042522	047503
11359	064346	042526	054522	005015
11360	064354	000		

ERR074: .BYTE 106
.ASCIZ /ERROR IN DRV DID NOT RELOAD HEADS/

ERR100: .ASCII /CURRENT SUPPLIED/<15><12>

.ASCIZ /CMND CYL TRK SEC OFFSET BUS AD WRD CT/

ERR101: .ASCII /PREVIOUS /<15><12>

.ASCIZ /CMND CYL TRK SEC WRD CT/

ERR102: .ASCIZ / PAT/

ERR108: .ASCIZ /OFFSET =/

ERR111: .ASCIZ /HD1 HD2 HD3/<15><12>

ERR200: .ASCIZ /DATA LATE WHEN UNLOADING HEADER/

ERR201: .ASCIZ /CONTROLLER ERROR DURING DRIVE SERVICING/

ERR202: .ASCIZ /ERROR DURING ERROR RECOVERY/<15><12>

11361	064355	105	051122	051117	ERR203: .ASCIZ	/ERROR RECOVERY SUCESSFUL/<15><12>
11362	064362	051040	041505	053117		
11363	064370	051105	020131	052523		
11364	064376	042503	051523	052506		
11365	064404	006514	000012			
11366	064410	051105	047522	020122	ERR204: .ASCIZ	/ERROR RECOVERY UNSUCCESSFUL/<15><12>
11367	064416	042522	047503	042526		
11368	064424	054522	052440	051516		
11369	064432	041525	051505	043123		
11370	064440	046125	005015	000		
11371	064445	105	051122	051117	ERR205: .ASCIZ	/ERROR THRESHOLD EXCEEDED/<15><12>
11372	064452	052040	051110	051505		
11373	064460	047510	042114	042440		
11374	064466	041530	042505	042504		
11375	064474	006504	000012			
11376	064500	052521	051505	044524	ERR206: .ASCIZ	/QUESTIONABLE DRIVE STATUS/<15><12>
11377	064506	047117	041101	042514		
11378	064514	042040	044522	042526		
11379	064522	051440	040524	052524		
11380	064530	006523	000012			
11381	064534	047503	052116	047522	ERR207: .ASCIZ	/CONTROLLER TIMED OUT WITH GO SET/<15><12>
11382	064542	046114	051105	052040		
11383	064550	046511	042105	047440		
11384	064556	052125	053440	052111		
11385	064564	020110	047507	051440		
11386	064572	052105	005015	000		
11387	064577	105	051122	051117	ERR208: .ASCIZ	/ERROR WHILE DIAGNOSING/<15><12>
11388	064604	053440	044510	042514		
11389	064612	042040	040511	047107		
11390	064620	051517	047111	006507		
11391	064626	000012				
11392	064630	127			ERR209: .BYTE	127
11393	064631	110	040505	042504	.ASCIZ	/HEADER ERROR/<15><12>
11394	064636	020122	051105	047522		
11395	064644	006522	000012			
11396	064650	025052	043052	051111	ERR210: .ASCIZ	/***FIRST ERROR INFO***/<15><12>
11397	064656	052123	042440	051122		
11398	064664	051117	044440	043116		
11399	064672	025117	025052	005015		
11400	064700	000				
11401	064701	122	050113	051517	ERR211: .ASCIZ	/RKPOS /
11402	064706	000040				
11403	064710	045522	040520	020124	ERR212: .ASCIZ	/RKPAT /
11404	064716	000				
11405	064717	101	042104	042522	ERR213: .ASCIZ	/ADDRESS OF BEGINNING OF SECTOR IN ERROR /
11406	064724	051523	047440	020106		
11407	064732	042502	044507	047116		
11408	064740	047111	020107	043117		
11409	064746	051440	041505	047524		
11410	064754	020122	047111	042440		
11411	064762	051122	051117	000040		
11412	064770	005015	040504	040524	ERR214: .ASCIZ	<15><12>/DATA READ/<15><12>
11413	064776	051040	040505	006504		
11414	065004	000012				
11415	065006	045522	051503	020061	ERR300: .ASCIZ	/RKCS1 RKCS2 RKWCR RKBA RKDA RKDC RKASOF RKER/<15><12>
11416	065014	020040	045522	051503		

11417 065022 020062 020040 045522
11418 065030 041527 020122 020040
11419 065036 045522 040502 020040
11420 065044 020040 045522 040504
11421 065052 020040 020040 045522
11422 065060 041504 020040 020040
11423 065066 045522 051501 043117
11424 065074 020040 045522 051105
11425 065102 005015 000
11426 065105 122 042113 020123
11427 065112 020040 051040 046513
11428 065120 030522 020040 051040
11429 065126 046513 031122 020040
11430 065134 051040 046513 031522
11431 065142 020040 051040 050113
11432 065150 051517 020040 051040
11433 065156 050113 052101 005015
11434 065164 000
11435 065165 122 042113 006523
11436 065172 000012
11437 065174 005015 042515 051523
11438 065202 020040 020040 020040
11439 065210 040440 020040 020040
11440 065216 020040 041040 005015
11441 065224 030040 020060 020040
11442 065232 020040 000
11443 065235 015 020012 030460
11444 065242 020040 020040 000040
11445 065250 005015 030440 020060
11446 065256 020040 020040 000
11447 065263 015 020012 030461
11448 065270 020040 020040 000040
11449 065276 047125 054105 042520
11450 065304 052103 042105 046440
11451 065312 046505 051117 020131
11452 065320 040520 044522 054524
11453 065326 052040 040522 000120
11454 065334 044524 042515 072
11455 065341 040 020040 020072
11456 065346 035040 020040 005015
11457 065354 000
11458 065356

ERR301: .ASCIZ /RKDS RKMR1 RKMR2 RKMR3 RKPOS RKPAT/<15><12>

ERR302: .ASCIZ /RKDS/<15><12>

ERR303: .ASCIZ <15><12>/MESS A B/<15><12>/ 00 /

ERR304: .ASCIZ <15><12>/ 01 /

ERR305: .ASCIZ <15><12>/ 10 /

ERR306: .ASCIZ <15><12>/ 11 /

ERR400: .ASCIZ /UNEXPECTED MEMORY PARITY TRAP/

TIM000: .ASCII /TIME:/

TIM001: .ASCIZ / : : /<15><12>

.EVEN

```
11459          .SBTTL  *** DUMP MEMORY ***
11460
11461          .EVEN
11462 065356 000040          .BLKW  40          :RESERVE SPACE FOR STACK
11463 065456 000000          ..STCK: .WORD  0          :STORED STACK POINTER
11464
11465 065460 012737 000340 177776 ..DUMP: MOV  #PR7,PS          :LOCK OUT ALL INTERRUPTS
11466 065466 010637 065456          MOV  SP,..STCK          :STORE STACK POINTER
11467 065472 012706 065456          MOV  #,..STCK,SP          :LOAD STACK POINTER
11468 065476 010546          MOV  R5,-(SP)          :PUSH R5 ON STACK
11469 065500 010446          MOV  R4,-(SP)          :PUSH R4 ON STACK
11470 065502 010346          MOV  R3,-(SP)          :PUSH R3 ON STACK
11471 065504 010246          MOV  R2,-(SP)          :PUSH R2 ON STACK
11472 065506 010146          MOV  R1,-(SP)          :PUSH R1 ON STACK
11473 065510 010046          MOV  R0,-(SP)          :PUSH R0 ON STACK
11474 065512 013705 000224          MOV  ..LOW,R5          :STORE FIRST ADDRESS
11475 065516 004737 066050          JSR  PC,..CR          :TYPE <CR><LF>
11476 065522 004037 066130          JSR  R0,..ASCII          :TYPE R0 =
11477 065526 066236          ..R0
11478 065530 011646          MOV  (SP),-(SP)
11479 065532 004737 066154          JSR  PC,..OCT
11480 065536 004737 066050          JSR  PC,..CR
11481 065542 004037 066130          JSR  R0,..ASCII          :TYPE R1 =
11482 065546 066251          ..R1
11483 065550 016646 000002          MOV  2(SP),-(SP)
11484 065554 004737 066154          JSR  PC,..OCT
11485 065560 004737 066050          JSR  PC,..CR
11486 065564 004037 066130          JSR  R0,..ASCII          :TYPE R2 =
11487 065570 066264          ..R2
11488 065572 016646 000004          MOV  4(SP),-(SP)
11489 065576 004737 066154          JSR  PC,..OCT
11490 065602 004737 066050          JSR  PC,..CR
11491 065606 004037 066130          JSR  R0,..ASCII          :TYPE R3 =
11492 065612 066277          ..R3
11493 065614 016646 000006          MOV  6(SP),-(SP)
11494 065620 004737 066154          JSR  PC,..OCT
11495 065624 004737 066050          JSR  PC,..CR
11496 065630 004037 066130          JSR  R0,..ASCII          :TYPE R4 =
11497 065634 066312          ..R4
11498 065636 016646 000010          MOV  10(SP),-(SP)
11499 065642 004737 066154          JSR  PC,..OCT
11500 065646 004737 066050          JSR  PC,..CR
11501 065652 004037 066130          JSR  R0,..ASCII          :TYPE R5 =
11502 065656 066325          ..R5
11503 065660 016646 000012          MOV  12(SP),-(SP)
11504 065664 004737 066154          JSR  PC,..OCT
11505 065670 004737 066050          JSR  PC,..CR
11506 065674 004037 066130          JSR  R0,..ASCII          :TYPE SP =
11507 065700 066340          ..SP
11508 065702 013746 065456          MOV  ..STCK,-(SP)
11509 065706 004737 066154          JSR  PC,..OCT
11510 065712 004737 066050          JSR  PC,..CR
11511 065716 004737 066050          JSR  PC,..CR
11512 065722 004037 066130          JSR  R0,..ASCII          :TYPE HEADER
11513 065726 066353          ..ADD
11514 065730 004737 066050          JSR  PC,..CR
```



```

11515 065734 004737 066050          JSR    PC,..CR
11516
11517 065740 010546          10$:  MOV    R5,-(SP)          ;TYPE ADDRESS
11518 065742 004737 066154          JSR    PC,..OCT
11519 065746 012703 000004          MOV    #4,R3              ;TYPE 4 LOCATIONS PER LINE
11520 065752 004037 066130          12$:  JSR    R0,..ASCII
11521 065756 066376          ..BLNK
11522 065760 012546          MOV    (R5)+,-(SP)        ;TYPE CONTENTS
11523 065762 004737 066154          JSR    PC,..OCT
11524 065766 005303          DEC    R3                  ;CHECK IF START OF NEXT LINE
11525 065770 001370          BNE   12$
11526 065772 004737 066050          JSR    PC,..CR
11527 065776 005777 113136          TST   @SWR
11528 066002 100001          BPL   15$
11529 066004 000000          HALT
11530 066006 023705 000226          15$:  CMP    ..HIGH,R5          ;CHECK IF FINISHED
11531 066012 103352          BHIS  10$
11532 066014 004737 066050          JSR    PC,..CR
11533 066020 004737 066050          JSR    PC,..CR
11534 066024 012600          MOV    (SP)+,R0           ;;POP STACK INTO R0
11535 066026 012601          MOV    (SP)+,R1           ;;POP STACK INTO R1
11536 066030 012602          MOV    (SP)+,R2           ;;POP STACK INTO R2
11537 066032 012603          MOV    (SP)+,R3           ;;POP STACK INTO R3
11538 066034 012604          MOV    (SP)+,R4           ;;POP STACK INTO R4
11539 066036 012605          MOV    (SP)+,R5           ;;POP STACK INTO R5
11540 066040 013706 065456          MOV    ..STCK,SP          ;RESTORE STACK POINTER
11541 066044 000000          HALT
11542 066046 000776          BR    -2
11543
11544 066050 105777 113074          ..CR: TSTB   @STPS          ;WAIT FOR READY
11545 066054 100375          BPL   ..CR
11546 066056 112777 000015 113066          5$:  MOVB   #15,@STPB          ;LOAD <CR>
11547 066064 105777 113060          TSTB   @STPS          ;WAIT FOR READY
11548 066070 100375          BPL   5$
11549 066072 112777 000012 113052          MOVB   #12,@STPB          ;LOAD <LF>
11550 066100 113701 001155          MOVB   $FILLS,R1         ;LOAD FILL COUNT
11551 066104 001410          BEQ   20$                ;RETURN IF NO FILLS
11552 066106 105777 113036          10$:  TSTB   @STPS          ;WAIT FOR READY
11553 066112 100375          BPL   10$
11554 066114 113777 001154 113030          MOVB   $NULL,@STPB       ;LOAD FILL CHARACTER
11555 066122 005301          DEC    R1                  ;CHECK IF FILL COMPLETE
11556 066124 001370          BNE   10$
11557 066126 000207          20$:  RTS    PC              ;RETURN
11558
11559 066130 012001          ..ASCII: MOV    (R0)+,R1     ;STORE ADDRESS OF ASCII
11560
11561 066132 105711          1$:  TSTB   (R1)              ;CHECK IF FINISHED
11562 066134 001406          BEQ   20$
11563 066136 105777 113006          2$:  TSTB   @STPS          ;WAIT FOR READY
11564 066142 100375          BPL   2$
11565 066144 112177 113002          MOVB   (R1)+,@STPB       ;TYPE CHARACTER
11566 066150 000770          BR    1$
11567
11568 066152 000200          20$:  RTS    R0              ;RETURN
11569
11570 066154 012701 000006          ..OCT: MOV    #6,R1         ;LOAD DIGIT COUNT INTO R1

```

11571	066160	005002				CLR	R2	:CLEAR R2 FOR OUTPUT
11572	066162	000407				BR	10\$:GET LEADING DIGIT
11573								
11574	066164	005002	5\$:			CLR	R2	:GET OCTAL DIGIT
11575	066166	006366	000002			ASL	2(SP)	
11576	066172	006102				ROL	R2	
11577	066174	006366	000002			ASL	2(SP)	
11578	066200	006102				ROL	R2	
11579	066202	006366	000002	10\$:		ASL	2(SP)	
11580	066206	006102				ROL	R2	
11581	066210	052702	000060			BIS	#60,R2	:MAKE IT ASCII
11582	066214	105777	112730	15\$:		TSTB	@STPS	:WAIT FOR READY
11583	066220	100375				BPL	15\$	
11584	066222	110277	112724			MOVB	R2,@STPB	:TYPE OCTAL DIGIT
11585	066226	005301				DEC	R1	:CHECK IF WHOLE WORD PRINTED
11586	066230	001355				BNE	5\$	
11587	066232	012616				MOV	(SP)+,(SP)	:ADJUST RETURN
11588	066234	000207				RTS	PC	:RETURN
11589								
11590	066236	030122	036440	020040	..R0:	.ASCIIZ	/R0 =	/
11591	066244	020040	020040	000				
11592	066251	122	020061	020075	..R1:	.ASCIIZ	/R1 =	/
11593	066256	020040	020040	000040				
11594	066264	031122	036440	020040	..R2:	.ASCIIZ	/R2 =	/
11595	066272	020040	020040	000				
11596	066277	122	020063	020075	..R3:	.ASCIIZ	/R3 =	/
11597	066304	020040	020040	000040				
11598	066312	032122	036440	020040	..R4:	.ASCIIZ	/R4 =	/
11599	066320	020040	020040	000				
11600	066325	122	020065	020075	..R5:	.ASCIIZ	/R5 =	/
11601	066332	020040	020040	000040				
11602	066340	050123	036440	020040	..SP:	.ASCIIZ	/SP =	/
11603	066346	020040	020040	000				
11604	066353	101	042104	042522	..ADD:	.ASCIIZ	/ADDRESS	CONTENTS/
11605	066360	051523	020040	041440				
11606	066366	047117	042524	052116				
11607	066374	000123						
11608	066376	020040	020040	000	..BLNK:	.ASCIIZ	/	/
11609		066404			.EVEN			
11610		066404			HDBUFF=.			
11611		066610			BUFADD=.	+132.		
11612	066404	044124	020105	045522	HELP:	.ASCII	/THE RK06-RK07-K MUST BE PROPERLY FORMATED 22 SECTORS PER TRACK/<15><12>	
11613	066412	033060	051055	030113				
11614	066420	026467	020113	052515				
11615	066426	052123	041040	020105				
11616	066434	051120	050117	051105				
11617	066442	054514	043040	051117				
11618	066450	040515	042524	020104				
11619	066456	031062	051440	041505				
11620	066464	047524	051522	050040				
11621	066472	051105	052040	040522				
11622	066500	045503	005015					
11623	066504	047524	041040	043505	.ASCII	/TO BEGIN PERFORMANCE TESTING TYPE CONTROL-C <^C>/<15><12>		
11624	066512	047111	050040	051105				
11625	066520	047506	046522	047101				
11626	066526	042503	052040	051505				

11627	066534	044524	043516	052040
11628	066542	050131	020105	047503
11629	066550	052116	047522	026514
11630	066556	020103	057074	037103
11631	066564	005015		
11632	066566	042514	040507	020114
11633	066574	047503	046515	047101
11634	066602	051504	040440	042522
11635	066610	006472	005012	
11636	066614	047124	026440	020040
11637	066622	042524	052123	042040
11638	066630	044522	042526	047040
11639	066636	005015		
11640	066640	047127	026440	020040
11641	066646	051127	052111	020105
11642	066654	040520	045503	053054
11643	066662	051105	054506	050040
11644	066670	041501	026113	040440
11645	066676	042116	052040	051505
11646	066704	020124	051104	053111
11647	066712	020105	006516	012
11648	066717	120	020116	020055
11649	066724	041440	040510	043516
11650	066732	020105	040520	040522
11651	066740	042515	042524	020122
11652	066746	020046	042524	052123
11653	066754	042040	044522	042526
11654	066762	047040	005015	
11655	066766	047123	026440	020040
11656	066774	042507	020124	052123
11657	067002	052101	051511	044524
11658	067010	051503	047440	020116
11659	067016	051104	053111	020105
11660	067024	006516	012	
11661	067027	104	020116	020055
11662	067034	042040	047522	020120
11663	067042	051104	053111	020105
11664	067050	020116	047101	020104
11665	067056	042507	020124	052123
11666	067064	052101	051511	044524
11667	067072	051503	005015	012
11668	067077	116	041440	047101
11669	067104	041040	020105	026460
11670	067112	020067	051117	040440
11671	067120	005015	012	
11672	067123	104	052101	020101
11673	067130	040520	052124	051105
11674	067136	051516	046440	051525
11675	067144	020124	042502	053440
11676	067152	044522	052124	047105
11677	067160	041040	020131	044124
11678	067166	020105	054105	051105
11679	067174	044503	042523	006522
11680	067202	012		
11681	067203	102	043105	051117
11682	067210	020105	042520	043122

.ASCII /LEGAL COMMANDS ARE:/<15><12><12>

.ASCII /TN - TEST DRIVE N/<15><12>

.ASCII /WN - WRITE PACK,VERFY PACK, AND TEST DRIVE N/<15><12>

.ASCII /PN - CHANGE PARAMETER & TEST DRIVE N/<15><12>

.ASCII /SN - GET STATISTICS ON DRIVE N/<15><12>

.ASCII /DN - DROP DRIVE N AND GET STATISTICS/<15><12><12>

.ASCII /N CAN BE 0-7 OR A/<15><12><12>

.ASCII /DATA PATTERNS MUST BE WRITTEN BY THE EXERCISER/<15><12>

.ASCII /BEFORE PERFORMANCE TESTING/<15><12><12>

11683	067216	051117	040515	041516
11684	067224	020105	042524	052123
11685	067232	047111	006507	005012
11686	067240	025052	053452	051101
11687	067246	044516	043516	025052
11688	067254	020052	043111	050040
11689	067262	047522	042503	051523
11690	067270	051117	044040	046101
11691	067276	020124	052504	044522
11692	067304	043516	042040	052101
11693	067312	020101	051124	047101
11694	067320	043123	051105	005015
11695	067326	020040	020040	052040
11696	067334	020117	051104	053111
11697	067342	026105	041040	042101
11698	067350	042440	041503	046440
11699	067356	054501	041040	020105
11700	067364	051127	052111	042524
11701	067372	020116	047117	050040
11702	067400	041501	027113	005015
11703	067406	020040	020040	042040
11704	067414	044522	042526	020123
11705	067422	052515	052123	041040
11706	067430	020105	051104	050117
11707	067436	042520	020104	054502
11708	067444	050040	047522	051107
11709	067452	046501	047440	020122
11710	067460	044527	044124	042040
11711	067466	047522	020120	047503
11712	067474	046515	047101	006504
11713	067502	005012	000	
11714		000001		

.ASCII /***WARNING*** IF PROCESSOR HALT DURING DATA TRANSFER/<15><12>

.ASCII / TO DRIVE, BAD ECC MAY BE WRITTEN ON PACK./<15><12>

.ASCIZ / DRIVES MUST BE DROPPED BY PROGRAM OR WITH DROP COMMAND/<15><12><12>

.END

ERR032	062464	7933	11167#			
ERR033	062501	7901	11170#			
ERR034	062510	7904	11172#			
ERR035	062523	7915	11174#			
ERR036	062540	7920	11177#			
ERR037	062553	7925	11179#			
ERR038	062577	7101	11183#			
ERR039	062614	7109	11186#			
ERR040	062631	7286	11190#			
ERR042	062641	8176	11193#			
ERR043	062660	8120	11197#			
ERR044	062675	7087	11201#			
ERR045	062713	7093	11204#			
ERR046	062726	7211	11206#			
ERR047	062746	7328	11210#			
ERR048	062762	7379	11212#			
ERR050	062774	7458	11215#			
ERR051	063013	7120	11219#			
ERR052	063033	8064	8068	8437	8449	11222#
ERR053	063111	7939	11231#			
ERR054	063152	7945	11238#			
ERR056	063220	7910	11246#			
ERR057	063233	7950	11248#			
ERR058	063274	7960	8461	11254#		
ERR059	063351	6989	8446	11262#		
ERR060	063422	7411	11269#			
ERR062	063446	7499	11273#			
ERR064	063473	8157	11277#			
ERR065	063503	7301	11280#			
ERR066	063526	7393	7840	11284#		
ERR067	063546	6819	7842	11287#		
ERR070	063567	7129	8355	11290#		
ERR071	063612	7144	8347	11294#		
ERR072	063636	7152	11298#			
ERR073	063657	7159	11301#			
ERR074	063704	7004	11305#			
ERR100	063747	6662	11312#			
ERR101	064060	6630	11326#			
ERR102	064142	6633	6667	11335#		
ERR108	064150	7658	11336#			
ERR111	064161	6828	11338#			
ERR200	064207	6373	11342#			
ERR201	064247	6381	6385	6428	11348#	
ERR202	064317	6731	6798	7502	7974	11355#
ERR203	064355	7680	11361#			
ERR204	064410	6928	6938	8421	11366#	
ERR205	064445	6888	6918	7672	11371#	
ERR206	064500	6737	6807	11376#		
ERR207	064534	6395	11381#			
ERR208	064577	8434	11387#			
ERR209	064630	8182	8441	11392#		
ERR210	064650	8467	11396#			
ERR211	064701	7412	11401#			
ERR212	064710	7416	11403#			
ERR213	064717	6539	11405#			
ERR214	064770	6545	11412#			

SPWRCT	055172	10445*	10446*	10447*	10449*	10461#													
SPWRDN	055046	3316	10439#	10451															
SPWRUP	055066	10440	10445#																
SQUES	001164	1697#	3485	3514	3553	3580	3606	3639	3663	3693	4557	4598	4639	4668					
		4692	4716	4740	4762	4783	4808	4841	4864	4884	5143	10036	10161	10178					
		10274																	
SRAND	055324	5226	5285	10531#															
SRDCHR	053330	10096#	10743																
SRDECT=	***** U	10745																	
SRDLIN	053460	10126#	10744																
SRDOCT=	***** U	10745																	
SRDSZ =	000031	10119#																	
SR2A =	***** U	10745																	
SSAVRE=	***** U	10745																	
SSB2D	055426	4111	4114	4117	4120	4123	4126	4129	10574#										
SSETUP=	000014	3292#	3313	3314	3316	3318	3343	3344	10085	10184									
SSIZE	054714	3349	10406#																
SSTUP =	177777	3292#																	
SSWR =	162000	979#	992	1011	1012	1013	1014	1015	1696										
SSWREG	001212	1717#	3337																
STESTN	001174	1708#																	
STKB	001146	1689#	10009	10016	10036	10083	10100	10106	10201	10207	10217	10241							
STKINT	054002	3745	10199#																
STKS	001144	1688#	5216*	5368*	10007	10014	10036	10083	10098	10104	10202*	10213*	10218*	10235*					
		10238	10258*																
STKSRV	054032	10199	10207#																
STN =	000001	979#	992																
STNPW	055652	10584	10585	10641#															
STNPWR	055626	10599	10600	10631#															
STPB	001152	1691#	10025*	10036	11546*	11549*	11554*	11565*	11584*										
STPFLG	001157	1695#	9957	10036															
STPS	001150	1690#	10023	10036	11544	11547	11552	11563	11582										
STRAP	056160	3314	10717#																
STRAP2	056202	10728#	10739																
STRP =	000004	10732#	10741#	10743	10744#	10745#													
STRPAD	056214	10722	10739#																
STSTM	001004	1655#																	
STSTM	001102	1668#																	
STTYIN	053714	10128	10129	10141	10159	10173	10177#												
STYPBN=	***** U	10741																	
STYPDS=	***** U	10741																	
STYPE	052670	9957#	10683	10732	10740														
STYPEC	053102	9987	9994	10001	10006#	10260													
STYPEX	053222	10029	10031	10034#															
STYPOC=	***** U	10741																	
SUNIT	001202	1711#																	
SUNITM	001010	1657#																	
SUSWR	001214	1718#																	
SVECT1	001240	1743#																	
SVECT2	001242	1744#																	
SXOFF =	000023	10011	10036																
SXON =	000021	10018	10036	10111															
S.S.SM=	***** U	1857	3365	3727	8676	8892	9135	9146	9184	9508	9520	9570	9590	9605					
		9632	9921	9933															
S2DEC	055536	10603#	10619																
.	= 067505	1619#	1623#	1629#	1634#	1636#	1642	1643#	1645#	1647#	1665#	1700	3311	3346#					

.EQUAT	1#	979#	1023			
.HEADE	1#	979#	982			
.KT11	1#					
.SETUP	1#	979#	3292			
.SWRHI	1#	979#	1007			
.SWRLO	979#	1015#	1017	1019	1021	1022
.SACT1	1#					
.SAPT8	1#	1701#				
.SAPTH	1#	979#	1637			
.SAPTY	1#	979#	10652			
.SASTA	1#					
.SCATC	1#	979#	1617			
.SCMTA	1#	979#	1659			
.SDB2D	1#					
.SDB2O	1#					
.SDIV	1#					
.SEOP	1#					
.SERRO	1#					
.SERRT	1#					
.SMULT	1#					
.SPOWE	1#					
.SRAND	1#					
.SRDDE	1#					
.SRDOC	1#					
.SREAD	1#	979#	10080			
.SR2AZ	1#					
.\$SAVE	1#					
.\$SB2D	1#					
.\$SB2O	1#					
.\$SCOP	1#					
.\$SIZE	1#	979#	10398			
.\$SUPR	1#					
.\$STRAP	1#	979#	10709			
.\$STYPB	1#					
.\$STYPD	1#					
.\$STYPE	1#	979#	9940			
.\$STYPO	1#					
.\$40CA	1#					
.1170	1#					

. ABS. 067505 000

ERRORS DETECTED: 0

CZR6PD,CZR6PD/SOL/CRF/NL:TOC/EQ:QNEWSW=SYSMAC.SML,DRIV11.P11,DUMP.P11,CZR6PD.P11
RUN-TIME: 27 34 2 SECONDS
RUN-TIME RATIO: 118/64=1.8
CORE USED: 59K (118 PAGES)