

RK611/RK06

SUBSYSTEM VERIFICATION 2
MD-11-DZR6N-B

EP-DZR6N-B-DL-A
COPYRIGHT © 1976
FICHE 1 OF 2

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of 100 frames of technical data, arranged in 10 rows and 10 columns. Each frame contains a small, high-contrast image of a technical drawing or data table. The frames are separated by a grid of small squares. The data in the frames is too small to be legible, but it appears to be organized into a structured format, possibly a table or a series of small diagrams. The overall appearance is that of a dense, organized collection of technical information.

RK611/RK06

SUBSYSTEM VERIFICATION 2
MD-11-DZR6N-B

EP-DZR6N-B-DL-A
COPYRIGHT © 1976
FICHE 2 OF 2

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of frames, each containing technical data. The data is organized into columns and rows, with some frames containing diagrams or tables. The text is small and difficult to read, but it appears to be a detailed technical specification or verification data for the MD-11-DZR6N-B subsystem.

MARK: RK611/RK06

MARK: RK611/RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:42 PAGE 2

.REM 3

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZR6N-B-D
PRODUCT NAME: RK611/RK06 SUBSYSTEM VERIFICATION:PART 2
DATE: AUGUST 1976
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: DAVE HOFFMAN

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 BY DIGITAL EQUIPMENT CORPORATION

001

REFEN-B - RK61: RK06 SUBSYSTEM VERIFICATION : PART 2
SERENB.CMB

MACY11 27(732) 03-NOV-76 21:43 PAGE 4

96
97

8.2.3.5
8.2.3.6

CONTROL Z (1Z) FUNCTION
CONTROL C (1C) FUNCTION

TABLE OF CONTENTS (CONT'D)

8.2.4	SPECIAL PARAMETER SPECIFICATIONS
8.2.4.1	PT - DATA PATTERN SELECT WORD
8.2.4.2	CS - CONTROL SWITCH WORD
8.3	DATA PATTERNS
9.0	DESCRIPTION OF TESTS
9.1	TEST 1 - OFFSET-TO-FAILURE MEASUREMENT
9.2	NPR/MAIN MEMORY TESTS
9.2.1	TEST 2 - NPR/MEMORY WORD ADDRESSING TEST
9.2.2	TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST
9.2.3	TEST 4 - NPR/MEMORY DATA PATTERN TEST
9.3	TEST 5 - UNIBUS CONTENTION TEST
9.4	TEST 6 - MULTI-DRIVE INTERFERENCE TEST
10.0	ERROR REPORTING
10.1	COMMON ERRORS
10.2	ERROR HANDLING
10.3	ERROR PRINTOUT EXAMPLES
11.0	RK06 HEAD ALIGNMENT AID
11.1	HARDWARE REQUIREMENTS
11.2	OPERATIONAL MODES
11.2.1	MANUAL SELECT MODE
11.2.1.1	MANUAL SELECT ALIGNMENT
11.2.1.2	MANUAL SELECT VERIFY
11.2.1.3	MANUAL SELECT EXERCISE
11.2.2	AUTO SELECT MODE
11.2.2.1	AUTO SELECT ALIGNMENT
11.2.2.2	AUTO SELECT VERIFY
11.2.2.3	AUTO SELECT EXERCISE
11.3	ALIGNMENT AID ERROR MESSAGES
APPENDIX A	SAMPLE ADDRESS 200 DEFAULT RUN
APPENDIX B	SAMPLE ADDRESS 204 RUN
APPENDIX C	SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN
APPENDIX D	SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

10000
10001
10002
10003
10004
10005
10006
10007
10008
10009
10010
10011
10012
10013
10014
10015
10016
10017
10018
10019
10020
10021
10022
10023
10024
10025
10026
10027
10028
10029
10030
10031
10032
10033
10034
10035
10036
10037
10038
10039
10040
10041
10042
10043
10044
10045

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 2 OF THE RK611/RK06 SUBSYSTEM VERIFICATION TESTS, WHICH PROVIDE A FUNCTIONAL SHAKEDOWN OF THE ENTIRE SUBSYSTEM, INCLUDING THE UNIBUS INTERFACE AND ACCESS TO MAIN MEMORY. THE TESTING IN PART 2 EMPLOYS WORST-CASE SITUATIONS INVOLVING HEAD OFFSETTING, MEMORY ADDRESSING AND DATA TRANSFER, UNIBUS CYCLE CONTENTION, AND MULTIPLE DRIVE OPERATIONS. ADDITIONALLY, AN RK06 HEAD ALIGNMENT AID IS PROVIDED TO FACILITATE ON-LINE ALIGNMENT OF DRIVE HEADS.

WITHIN THE VARIOUS SUBSYSTEM TESTS, EMPHASIS IS GIVEN TO USEFUL SCOPE LOOPS AND OPERATOR SPECIFICATION OF TEST PARAMETER VALUES. AT THE BEGINNING OF TESTING, THE FOLLOWING OPTIONS MAY BE SPECIFIED BY THE USER:

- RK611 REGISTER ADDRESS
- RK06 VECTOR ADDRESS
- RK06 PRIORITY LEVEL
- DRIVE (S) TO BE TESTED
- TEST (S) TO BE RUN
- NUMBER OF TEST ITERATIONS
- INITIAL DISK ADDRESS ON TRANSFERS
- DATA PATTERNS USED
- STALL TIME BETWEEN OPERATIONS

IN ADDITION, STANDARD SWITCH REGISTER OPTIONS ARE PROVIDED, FOR A VARIETY OF PROGRAM LOOPING, RUNNING, AND REPORTING MODES.

ERRORS WILL BE REPORTED AS THEY OCCUR. THE REPORT WILL INCLUDE A TEST NUMBER AND DESCRIPTION, CURRENT AND PREVIOUS OPERATIONS, GOOD AND BAD TEST DATA, AND APPLICABLE DEVICE REGISTER CONTENTS.

2.0 HARDWARE REQUIREMENTS

2.1 REQUIREMENTS FOR SUBSYSTEM TESTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN PART 2 OF THE SUBSYSTEM VERIFICATION TESTS:

- PDP-11/04, (05, 10 MFG. ONLY), 20, 34, 35, 40, 45, 50, 70 OR PDQ
- 16 K MEMORY
- CONSOLE TELETYPE
- RK06 UNIBUS CONTROLLER (RK611)
- 1 TO 8 RK06 DRIVES
- 1 TO 8 RK06 DISK CARTRIDGES (FORMATTED IN 20 OR 22 SECTOR FORMAT)

2.2 REQUIREMENTS FOR HEAD ALIGNMENT AID

146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201

GO1

DZRB6N-8 - RK611 RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 7
DZRB6NB.CM3

202
203

ADDITIONAL HARDWARE IS REQUIRED BY THE RK06 HEAD ALIGNMENT AID:

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

RK06 FIELD TEST BOX (OR ALIGNMENT SECTION THEREOF)
RK06 ALIGNMENT CARTRIDGE
RK06 HEAD ALIGNMENT TOOL

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE SUBSYSTEM VERIFICATION PROGRAM, THE RK611 CONTROLLER DIAGNOSTIC (MAINDEC-11-DZR6A THRU DZR6F AND DZR6K) AND RK06 DRIVE DIAGNOSTIC (MAINDEC-11-DZR6H THRU DZR6J) SHOULD FIRST BE RUN, TO RESOLVE BASIC, SOLID HARDWARE FAULTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 SYSMAC

THIS PROGRAM USES PORTIONS OF THE SYSMAC DIAGNOSTIC SYSTEM MACRO PACKAGE.

4.2 XXDP

THIS PROGRAM MAY BE LOADED UNDER XXDP. SUBSYSTEM TESTS 1-6 MAY BE RUN IN CHAIN OR DUMP MODE, BUT THE ALIGNMENT AID MAY ONLY BE RUN IN DUMP MODE.

4.2.1 CHAIN MODE

THIS PROGRAM IS DESIGNED TO BE RUN IN THE DEFAULT MODE WHEN CHAINED UNDER XXDP. THUS, INPUT DIALOGUE IS BYPASSED, AND RK06 UNIBUS ADDRESS AND INTERRUPT VECTOR ARE DEFAULTED TO STANDARD VALUES. ALL DRIVES PRESENT AND READY WILL BE TESTED WITH THE EXCEPTION OF DRIVE 0, IF THE RK06 CONTAINS THE XXDP MEDIUM.

4.2.2 DUMP MODE

THE PROGRAM CAN BE RUN IN DUMP MODE, WITH OR WITHOUT DEFAULT PARAMETERS. DRIVE 0 MAY BE TESTED, BUT IF IT CONTAINS THE XXDP MEDIUM, THE OPERATOR MUST REPLACE THE XXDP PACK WITH A FORMATTED SCRATCH PACK PRIOR TO TESTING (OR AN ALIGNMENT CARTRIDGE IF ALIGNMENT IS TO BE DONE ON DRIVE 0). A MESSAGE WILL INFORM THE OPERATOR WHEN THIS IS NECESSARY.

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

4.3 ACT/APT

THE PROGRAM IS COMPATIBLE WITH ACT/APT CONVENTIONS. SUBSYSTEM TESTS 1-6 MAY BE RUN IN AUTOMATIC OR DUMP MODE, AND THE HEAD ALIGNMENT AID MAY ONLY BE RUN IN DUMP MODE.

4.3.1 AUTOMATIC MODE

THE PROGRAM PROVIDES FOR AUTOMATIC APT/ACT OPERATION. IN THIS MODE, PARAMETERS REVERT TO DEFAULT VALUES, AND ALL DRIVES WHICH ARE PRESENT AND READY ARE TESTED.

4.3.2 DUMP MODE

IN DUMP MODE, PARAMETERS MAY BE INPUT, OR DEFAULTED, AND SUBSYSTEM TESTS 1-6 OR HEAD ALIGNMENT AID MAY BE RUN.

4.3.3 APT ETABLE DEFINITIONS

THE FOLLOWING DEFINITIONS ARE VALID FOR SPECIFYING APT ENVIRONMENTAL TABLE (ETABLE) ENTRIES, VIA THE APT UTILITY PROGRAM "APPLU" :

1. SOFTWARE ENVIRONMENT

=1 IF APT SCRIPT MODE
=0 IF STANDALONE MODE

2. ENVIRONMENT MODE BYTE

BIT 7 = 1 ETABLE DOES SIZING
= 0 PROGRAM DOES SIZING
BIT 6 = 1 SPOOL MESSAGES TO APT IF SCRIPT MODE
= 0 DON'T SPOOL TO APT
BIT 5 = 1 SUPPRESS CONSOLE OUTPUT
= 0 ALLOW CONSOLE OUTPUT
BITS 4-0 NOT USED

3. SWITCH 1 (SOFTWARE SWITCH REGISTER)

IF ENVIRONMENT MODE BIT 7 (SIZING BIT) IS SET TO 1, THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD OF THE HARDWARE CONSOLE SWITCH REGISTER. REGARDLESS OF WHICH ONE IS USED, ALL BITS DEFINED IN SECTION 6.2 (SWITCH REGISTER OPTIONS) MAY BE USED WHEN RUNNING IN STANDALONE MODE. IN APT SCRIPT MODE, HOWEVER, BIT 14 (LOOP ON TEST) MUST ALWAYS BE SET TO 0.

4. SWITCH 2 (USER SWITCH REGISTER)
NOT USED

J01

DZR6N-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 10
DZR6NB.CMB

315
316

5. MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES
NOT USED

317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372

6. INTERRUPT VECTOR 1
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 210

7. BUS PRIORITY 1
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 5

8. INTERRUPT VECTOR 2
NOT USED

9. BUS PRIORITY 2
NOT USED

10. BASE ADDRESS
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 177440

11. DEVICE MAP
USED WHEN ENVIRONMENT MODE BIT 7 = 1. EACH BIT SET TO 1 IN BITS
0-7 WILL SELECT THE CORRESPONDING DRIVE TO BE TESTED.
BITS 8-15 ARE NOT USED.

4.4 DUAL-ACCESS

THIS PROGRAM DOES NOT UTILIZE DUAL-ACCESS. FOR THE PURPOSES OF ALL TESTS (1-6), AND THE HEAD ALIGNMENT AID (SECTION 11), THE OPERATOR MUST GUARANTEE THAT THERE IS NO INTERFERENCE FROM THE UNUSED PORT. IF FAILURES ARE ENCOUNTERED IN TESTS 1-6 DUE TO INTERFERENCE FROM THE OTHER PORT, THE OPERATOR IS ADVISED TO SWITCH THAT PORT OFF-LINE AND RE-RUN THE TESTS.

4.5 MEMORY MANAGEMENT

THIS PROGRAM SUPPORTS MEMORY MANAGEMENT OPTIONS KT11C,D AND PDP11/70, FOR THE PURPOSES OF TESTS 2-4 ONLY. IN THESE TESTS (SEE SECTION 9.7) DIRECT ACCESS TO ALL OF PHYSICAL MEMORY ABOVE THE PROGRAM IS EXERCISED. FOR THE DURATION OF THESE 3 TESTS, MEMORY MANAGEMENT IS ENABLED. IN ALL OTHER TESTS, AND DURING THE USE OF THE HEAD ALIGNMENT AID, MEMORY MANAGEMENT IS DISABLED.

4.6 MEMORY PARITY CHECK

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, ALL TESTING IS DONE WITH PARITY CHECK ENABLED. PARITY ERRORS ARE REPORTED, AND TESTING IS ABORTED, UNLESS THE BYTE LABELED MEMABT WAS PREVIOUSLY LOADED WITH A NON-ZERO VALUE.

L01

DZR6N-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2 : MACY11 27(732) 03-NOV-75 21:43 PAGE 12
DZR6NB.CMB

373

4.7 BAD SECTORS

374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429

ACCORDING TO A CONTROL SWITCH OPTION (SEE SECTION 8.2.4.2) THE BAD SECTOR FILE ON EACH DRIVE TO BE TESTED IS TYPED AT THE CONSOLE PRIOR TO TESTING. ALL DATA ERRORS OCCURRING IN THE PROGRAM ARE MASKED OUT IF THEY OCCUR ON SECTORS DESIGNATED AS BAD, (BY EITHER FACTORY OR SOFTWARE).

4.8 EXECUTION TIME

EXECUTION TIME IS DEPENDENT UPON PARAMETERS INPUT BY THE OPERATOR (SUCH AS STALL TIME OR ITERATION COUNT), AND TO A GREAT DEGREE, UPON THE AMOUNT OF MEMORY. HOWEVER, THE "AVERAGE TIME" REQUIRED TO RUN A QUICK VERIFICATION (FIRST PASS) IS 2 MINUTES (FOR A 64K SYSTEM). A SUBSEQUENT PASS WITH PARAMETERS DEFAULTED REQUIRES 3 MINUTES PER DRIVE (ON A 64-K SYSTEM).

5.0 PROGRAM LOADING

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

6.0 STARTING PROCEDURE

6.1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS OF TESTS 1-6 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS, RK06 UNIBUS ADDR. AND INTERRUPT VECTOR FOR TESTS 1-6
224 HEAD ALIGNMENT AID START ADDRESS

NOTE

FOR HEAD ALIGNMENT AID OPERATING INFORMATION PLEASE SKIP TO SECTION 11. THE SECTIONS WHICH IMMEDIATELY FOLLOW APPLY ONLY TO THE SUBSYSTEM TESTS (1-6).

6.2 SWITCH REGISTER OPTIONS USED

THIS PROGRAM IS DESIGNED TO ALLOW THE USE OF THE HARDWARE SWITCH REGISTER IF PRESENT, OR THE SYSMAC-SUPPORTED SOFTWARE SWITCH REGISTER (IF HARDWARE SWR IS NOT PRESENT OR IS SET TO ALL ONES). IN EITHER CASE, THE FOLLOWING OPTIONS ARE IMPLEMENTED WHEN THE APPROPRIATE BITS

NO1

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB

MACY11 27(732) 03-NOV-76 21:43 PAGE 14

430

ARE SET TO 1:

BIT	OPTION
---	-----
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR REPORTS
12	REPORT DESCRIPTION ONLY, ON ERRORS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
09	LOOP ON ERROR
08	APPLY RANDOM STALL BETWEEN OPERATIONS
06	REPORT ONE ERROR PER TRANSFER IN TESTS 2-4
01	INHIBIT WRITES IN TEST 1
00	REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 2-4.

NOTE

FOR ADDITIONAL PROGRAM CONTROL OPTIONS, SEE DESCRIPTION OF CONTROL SWITCH WORD (CS), SECTION 8.2.4.2 .

7.0 OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 5.0)
2. LOAD A FORMATTED PACK ON EACH DRIVE TO BE TESTED.
3. BRING DRIVE (S) TO ONLINE STATE, WRITE ENABLED, AND LOCKED ON PORT.
4. LOAD THE DESIRED STARTING ADDRESS (SEE SECTION 6.1)
5. SET SWITCHES IF DESIRED (SEE SECTION 6.2)
6. PRESS START OR GIVE APPROPRIATE MONITOR START COMMAND.

8.0 PROGRAM ACTION

8.1 DESCRIPTION OF OPERATING PARAMETERS

AFTER THE PROGRAM IS STARTED, IT TYPES ITS IDENTIFICATION AS FOLLOWS: "DZR6N-B- RK11/RK06 SUBSYSTEM VERIFICATION:PART 2", FOLLOWED BY: "LAST PHYS MEM ADRS=XXXXXXXX". THEN, EITHER THE TESTS BEGIN EXECUTION WITH DEFAULT PARAMETERS, OR AN OPERATOR INTERACTIVE MODE IS ENTERED (SEE STARTING ADDRESSES, SECTION 6.1). IN THIS MODE OPERATING PARAMETERS MAY BE LISTED OR ALTERED BY VARIOUS MEANS. ALL PARAMETERS ARE TREATED AS OCTAL NUMBERS, AND THE PARAMETER LIST IS SHARED AMONG THE TESTS.

THE FOLLOWING IS THE LIST OF OPERATING PARAMETERS, (WHICH APPLY TO THE CURRENT SELECTION OF DRIVES AND TESTS). AFTER EACH, IS INDICATED THE

Vertical text on the left margin, possibly a page number or reference code, appearing as a series of characters.

487
488

VALID RANGE OF VALUES FOR THAT PARAMETER (IN OCTAL), AND ITS DEFAULT
VALUE. ALL PARAMETERS ARE ENTERED AS OCTAL NUMBERS, AND THE PARAMETER

MNEMONICS SHOWN ARE THOSE USED BY THE PROGRAM.

FC FIRST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, FC=0-631, DEFAULT=0

LC LAST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, LC=0-632, DEFAULT=632

FT FIRST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, FT=0-2, DEFAULT=0

LT LAST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, LT=0-2, DEFAULT=2

S0 FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. S0=0-23, DEFAULT=0.

S1 LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. S1=0-23, DEFAULT=23.

S2 FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE, S2=0-25, DEFAULT=0.

S3 LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE. S3=0-25, DEFAULT = 25.

PT DATA PATTERN SELECT WORD (SEE SECTION 8.2.4.1 FOR DETAILS) PT=0-177777, DEFAULT=0

CS CONTROL SWITCH WORD (SEE SECTION 8.2.4.2 FOR DETAILS) CS=0-000062, DEFAULT=0

ST NUMBER OF UNIT STALL TIMES WITH WHICH TO STALL (DELAY) BETWEEN RK06 COMMANDS

8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)

8.2.1 DRIVE SELECTION

AFTER AN ADDRESS 204 START, THE PROGRAM INDICATES INPUT MODE BY TYPING "PARAMETER INPUT MODE". THEN, THE RK611 REGISTER ADDRESS, RK06 VECTOR ADDRESS AND RK06 PRIORITY ARE OPENED FOR POSSIBLE MODIFICATION AS FOLLOWS:

RK06 BUS ADR = 177440 NEW=(TYPE NEW VALUE HERE)

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

E02

DZ6N-8 - RK611 RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 18
DZ6NB.CMB

545
546

RK06 VEC ADR = 210 NEW=(TYPE NEW VALUE HERE)
RK06 PRIORITY= 5 NEW=(TYPE NEW VALUE HERE)

547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602

NEXT THE PROGRAM TYPES THE NUMBER(S) OF THE CURRENT DRIVES UNDER TEST, FOLLOWED BY A (*) ON THE NEXT LINE TO REQUEST NEW DRIVE NUMBERS. FOR EXAMPLE:

PARAMETER INPUT MODE

DRIVE(S)=0,1,2,4,7
* (INPUT, IF ANY, GOES HERE)

THE OPERATOR TYPES THE NEW NUMBERS (SEPARATED BY COMMAS) PLUS <CR> OR SIMPLY <CR> TO LEAVE THEM UNCHANGED. IF HE ENTERS DRIVE NUMBERS, THESE DRIVES ARE THEN CHECKED FOR VALID STATUS BY THE PROGRAM, AND ALL THOSE TYPED WHICH ARE ACCEPTABLE WILL NOW BE LISTED AS BEFORE. THE OPERATOR MAY MANUALLY RECONFIGURE THE DRIVES AND TYPE IN NEW DRIVE NUMBERS, AND THESE WILL BE CHECKED AND TYPED, UNTIL THE OPERATOR FINALLY TYPES JUST <CR>.

ON INITIAL ENTRY FROM ADDRESS 204, THE DRIVES WHICH ARE LISTED ARE ALL THOSE ON THE SUBSYSTEM WHICH ARE ON-LINE, READY, WRITE-ENABLED, AND LOADED WITH A CARTRIDGE OTHER THAN AN ALIGNMENT PACK. (THIS IS THE DEFAULT DRIVE SELECTION FOR ADDRESS 200 START). ON ALL OTHER ENTRIES TO THE DRIVE SELECTION ROUTINE (VIA <↑C> AS DESCRIBED IN SECTIONS 8.2.2.5 AND 8.2.3.6) THE PROGRAM FIRST TYPES :

TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>
* (CHARACTER GOES HERE)

NEXT, THE DRIVES ARE LISTED AS SHOWN ABOVE. IF (A) IS TYPED, ALL AVAILABLE DRIVES WILL BE LISTED, AND IF NOT, THE PREVIOUS SELECTION OF DRIVES WILL BE LISTED.

8.2.2 TEST SELECTION

WHEN THE OPERATOR FINALLY TYPES JUST <CR> IN RESPONSE TO THE DRIVE REQUEST, THE PROGRAM ENTERS THE TEST INPUT ROUTINE. THE FOLLOWING INSTRUCTIVE LINES ARE TYPED :

L = LIST TESTS
C = CHANGE TESTS
I = INPUT PARAMETERS AND RUN TESTS

THEN, THE FOLLOWING REQUEST IS MADE :

ENTER L, C, OR I
* (CHARACTER GOES HERE)

THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.

8.2.2.1 LIST TESTS. (L)

G02

DZR6N-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 20
DZR6NB.CMB

603
604

IF (L) IS TYPED, THE PROGRAM TYPES THE CURRENT DESIRED TEST LIST, AS

605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660

FOLLOWS :

TEST	ITERATIONS
1	0
2	177777
3	400
4	25
ETC.	

THE ITERATION NUMBER IS THE NUMBER OF TIMES THE TEST WILL BE RUN ON THIS PASS, AND IT MUST BE BETWEEN 0 AND 177777. IF IT IS ZERO, THAT TEST WILL NOT BE RUN. ON THE FIRST TIME THROUGH, ALL TESTS ARE LISTED WITH DEFAULT ITERATIONS (YET TO BE DETERMINED).

WHEN THE LIST IS COMPLETED, THE PROGRAM RETURNS TO TYPE : "ENTER L,C, OR I" AGAIN.

8.2.2.2 CHANGE TESTS, (C)

IF (C) IS TYPED, THE PROGRAM THEN ASKS: "TO DEFAULT TESTS TYPE D<CR>, ELSE <CR>". IF JUST <CR> IS TYPED, THE PROGRAM SEQUENTIALLY OPENS EACH TEST FOR POSSIBLE ALTERATION OF THE ITERATION NUMBER. EACH TEST IS OPENED AND A NEW LINE IS TYPED, GIVING THE TEST NO. AND ITER. NO. IN OCTAL, FOLLOWED BY (*) ON THE SAME LINE. FOR EXAMPLE :

TEST	ITERATIONS
0	0 * (INPUT, IF ANY, GOES HERE)

THE OPERATOR THEN TYPES THE NEW VALUE PLUS <CR>, OR JUST <CR> TO LEAVE IT UNCHANGED. IF THE OPERATOR TYPES A NUMBER FOLLOWED BY (!) (EXCLAMATION POINT), THE NUMBER JUST ENTERED WILL BE LOADED INTO THE ITERATION TABLE FOR ALL REMAINING TESTS. AFTER ALL TESTS HAVE BEEN OPENED, THE PROGRAM RETURNS TO TYPE : "ENTER L,C, OR I" AGAIN.

8.2.2.3 INPUT PARAMETERS AND RUN TESTS, (I)

IF (I) IS TYPED, THE PROGRAM ENTERS THE PARAMETER LIST ALTERATION ROUTINE, DESCRIBED IN SECTION 8.2.3, IN WHICH OPERATING PARAMETERS MAY BE INPUT, AND TESTING BEGUN.

8.2.2.4 CONTROL Z (↑Z) FUNCTION

IF THE OPERATOR WISHES TO EXIT AT ANY TIME FROM L OR C MODE, AND RETURN TO SELECT A NEW MODE (L,C, OR I), CONTROL Z (↑Z) MAY BE TYPED.

661

9.2.2.5 CONTROL C (IC) FUNCTION

662
663
664
665 IF THE OPERATOR WISHES TO TERMINATE L,C, OR I MODE, AND RETURN TO
666 REQUEST NEW DRIVES AND TESTS, CONTROL 'C' (IC) MAY BE TYPED.
667
668
669

670
671
672 8.2.3 PARAMETER LIST ALTERATION
673

674 THE PROGRAM NEXT TYPES THE FOLLOWING INSTRUCTIVE LINES:
675

676 T = TYPE LIST
677 O = OPEN LIST
678 S = SET INDIVIDUAL PARAM.
679 R = RUN TESTS
680 ENTER T,O,S, OR R
681 * (CHARACTER GOES HERE)
682

683 THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.
684

685 8.2.3.1 TYPE LIST, (T)
686

687 IF (T) IS TYPED, THE PROGRAM TYPES THE CURRENT PARAMETER LIST, AS
688 FOLLOWS:
689

690 FC = XXXXXX
691 LC = XXXXXX
692 ETC.
693

694 WHEN THE LIST IS COMPLETED, THE PROGRAM RETURNS TO TYPE "ENTER T,O,S,
695 OR R" AGAIN.
696

697 8.2.3.2 OPEN LIST, (O)
698

699 IF (O) IS TYPED THE PROGRAM THEN ASKS: "TO DEFAULT ALL PARAMETERS
700 TYPE D<CR>, ELSE <CR>". IF JUST <CR> IS TYPED, THE PROGRAM
701 SEQUENTIALLY OPENS EACH OPERATING PARAMETER IN THE LIST FOR ALTERATION
702 (IN THE ORDER SHOWN IN SECTION 8.1). EACH PARAMETER IS OPENED AND A
703 NEW LINE IS TYPED, GIVING THE PARAMETER AND ITS CURRENT VALUE (IN
704 OCTAL), FOLLOWED BY (*) ON THE SAME LINE. FOR EXAMPLE:
705

706 IC=3 * (INPUT, IF ANY, GOES HERE)
707

708 THE OPERATOR THEN TYPES THE NEW VALUE PLUS A <CR> OR JUST <CR> TO
709 LEAVE THAT PARAMETER UNCHANGED. AFTER ALL PARAMETERS HAVE BEEN OPENED
710 FOR ALTERATION, THE PROGRAM RETURNS TO TYPE "ENTER T,O,S, OR R" AGAIN.
711

712 8.2.3.3 SET INDIVIDUAL PARAMETER, (S)
713
714
715
716
717

K02

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 24
DZR6NB.CMB

718
719

IF (S) IS TYPED, THE PROGRAM TYPES THE PROMPTER (>) ON THE NEXT LINE,
AND WAITS FOR THE OPERATOR TO SET THE VALUE OF ANY INDIVIDUAL

720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775

PARAMETER:

ENTER T,O,S, OR R

*S

> (ENTER PARAMETER AND VALUE HERE)

THE OPERATOR TYPES THE PARAMETER AND ITS NEW VALUE FOLLOWED BY <CR>, AND THE PROGRAM GIVES THE PROMPTER (>) AGAIN AS IN THE FOLLOWING EXAMPLE:

```
> FC = 600
> FS = 12
> IT = 1
> ETC.
```

TO TERMINATE INDIVIDUAL SETTING MODE, THE OPERATOR TYPES CONTROL Z (↑Z), AND THE PROGRAM RETURNS TO TYPE "ENTER T,O, S, OR R" AGAIN.

8.2.3.4 RUN TESTS, (R)

IF THE OPERATOR TYPES (R), EXECUTION OF TESTS BEGINS, USING ALL THE CURRENT PARAMETERS.

8.2.3.5 CONTROL Z (↑Z) FUNCTION

IF THE OPERATOR WISHES TO EXIT AT ANY TIME FROM T, O, S, OR R MODE (RUNNING TESTS), AND RETURN TO SELECT A NEW MODE (T,O,S, OR R), CONTROL Z (↑Z) MAY BE TYPED.

8.2.3.6 CONTROL C (↑C) FUNCTION

IF THE OPERATOR WISHES TO TERMINATE T, O, OR S MODE, OR STOP TEST EXECUTION (R MODE), CONTROL C (↑C) MAY BE TYPED, AND THE PROGRAM WILL RETURN TO REQUEST NEW DRIVE SELECTION (SEE SECTION 8.2.1). (ON AN ADDRESS 200 DEFAULT RUN, TYPING CONTROL C CAUSES THE PROGRAM TO HALT AFTER COMPLETING ANY DATA TRANSFER).

8.2.4 SPECIAL PARAMETER SPECIFICATIONS

8.2.4.1 PT-DATA PATTERN SELECT WORD

THE OPERATOR SPECIFIES A WORD OF SIXTEEN BITS (6 OCT. DIGITS), TO SELECT UP TO ALL SIXTEEN OF THE PATTERNS, LISTED IN SECTION 9.5.1.

776
777

THE NO. OF ANY BIT SET IN PT CORRESPONDS TO THE NO. OF A PATTERN
CHOSEN. IF THE HIGH BIT OF PT (BIT 15) IS SET, THE USER WILL BE ASKED

778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833

TO TYPE UP TO SIXTEEN DATA WORDS WHICH WILL BE LOADED INTO THE BUFFER FOR PATTERN 15. THE PROGRAM TYPES:

"SELECT USER DEFINED PATTERN 15"

NEXT, EACH OF UP TO 16 WORDS ARE REQUESTED BY THE PROGRAM IN THE SAME MANNER IN WHICH THE OPERATING PARAMETERS ARE OPENED FOR ALTERATION (SEE SECT. 8.2.3.2):

WORD 0 = (OLD VALUE) * (NEW VALUE GOES HERE)
WORD 1 = (OLD VALUE) * (NEW VALUE GOES HERE)
ETC.

FOR EACH WORD, THE USER TYPES THE NEW VALUE PLUS <CR>, OR JUST <CR>, (TO LEAVE IT UNCHANGED). TO INPUT LESS THAN 16 WORDS FOR PATTERN 15, THE OPERATOR MAY TYPE (!) (EXCLAMATION POINT) AND THE PROGRAM WILL FILL UP THE REST OF THE PATTERN 15 BUFFER WITH THE LAST WORD SPECIFIED. NEXT, THE PROGRAM WILL CONTINUE, IN THE PROPER PARAMETER INPUT MODE (O OR S).

8.2.4.2 CS- CONTROL SWITCH WORD

IN ADDITION TO THE OPTIONS PROVIDED BY THE SWITCH REGISTER (HARDWARE OR SOFTWARE), A CONTROL SWITCH WORD (CS) MAY BE SPECIFIED AMONG THE OPERATING PARAMETERS TO PROVIDE THE FOLLOWING OPTIONS:

BIT	OPTION
---	-----
05	DROP DRIVE IF 20 (DEC) ERRORS EXCEEDED
04	TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS)
01	INHIBIT OFFSET REPORTS IN TEST 1

NOTE

OTHER BITS UNUSED

8.3 DATA PATTERNS

THIS SECTION DESCRIBES THE DATA PATTERNS AVAILABLE FOR USE IN THE TESTS. EACH DATA PATTERN IS COMPRISED OF 16 WORDS. PATTERNS ARE REPEATED AS NECESSARY TO OBTAIN THE DESIRED WORD COUNT ON THE WRITE DATA COMMANDS. THE MAXIMUM ALLOWABLE WORD COUNT IS DETERMINED BY THE AMOUNT OF AVAILABLE BUFFER SPACE AT THE END OF THE PROGRAM, AND COULD BE AS LARGE AS 65,536 (DEC) WORDS. DATA SPIRALING AND TRACK SWITCHING

834

IS THIS POSSIBLE DURING THE DATA TESTS.

E03

0000
0001
0002
0003
0004
0005
0006
0007
0008
0009
0010
0011
0012
0013
0014
0015
0016
0017
0018
0019
0020
0021
0022
0023
0024
0025
0026
0027
0028
0029
0030
0031
0032
0033
0034
0035

8 SHIFTED 1 IN FIELD OF 0'S	9 SHIFTED 0 IN FIELD OF 1'S	10 ALTERNATING 0-1	11 ALTERNATING 1-0
000001	177776	052525	125252
000002	177775	052525	125252
000004	177773	052525	125252
000010	177767	052525	125252
000020	177757	052525	125252
000040	177737	052525	125252
000100	177677	052525	125252
000200	177577	052525	125252
000400	177377	052525	125252
001000	176777	052525	125252
002000	175777	052525	125252
004000	173777	052525	125252
010000	167777	052525	125252
020000	157777	052525	125252
040000	137777	052525	125252
100000	077777	052525	125252

12 SHIFTING 0'S AND 1'S	13 COMPOSITE ROTATING	14 PSEUDO- RANDOM	15 USER- DEFINED
000001	072307		
000003	135143		
000007	156461		
000017	167230		
000037	073514		
000077	035646		
000177	016723		
000377	107351		
000777	143564		
001777	061672		
003777	030735		
007777	114356		
017777	046167		
037777	123073		
077777	151453		
177777	164616		

936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991

9.0 DESCRIPTION OF TESTS

9.1 TEST 1 - OFFSET-TO-FAILURE MEASUREMENT

IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC. THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH 167230(OCT) (TO ESTABLISH A KNOWN PATTERN), BUT THEY ARE NOT TESTED. THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE OFFSETS, UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER: SECTORS FS AND LS ON CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF INCREASING TRACKS, AS FOLLOWS:

OFFSET-TO-FAILURE MEASUREMENTS:

TRACK	CYLN	SECT	-OFST (UIN)	-OFST (UIN)
X	XXX	XX	XXXX	XXXX
X	XXX	XX	XXXX	XXXX
X	XXX	XX	XXXX	XXXX

ETC.

NOTE

IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS TO 12(OCT), FOR THIS TEST.

9.2 NPR / MAIN MEMORY TESTS

THIS GROUP OF TESTS EXERCISES ALL MEMORY ABOVE THE PROGRAM, VIA READ/WRITE NPR TRANSFERS WITH THE RK06. THE TESTS ARE DESIGNED TO DIAGNOSE MEMORY TRANSFER FAILURES DURING HEAVY NPR ACTIVITY, IN SYSTEMS PROVIDED WITH MEMORY MANAGEMENT (KT 11 C, D OR PDP 11/70). ALTHOUGH MEMORY MANAGEMENT IS NOT REQUIRED.

NOTE

THERE IS NO PROGRAM RELOCATION (EXCEPT FOR THE XXDP LOADER, IF PRESENT).

G03

DZR6N-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB

MACY11 27(732) 03-NOV-76 21:43 PAGE 33

992
993

HENCE, MEMORY IN WHICH THE PROGRAM
RESIDES IS NOT TESTED. ALSO, THE UNIBUS

.
:
.

994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049

I/O ADDRESSES ARE NOT TESTED.

THERE ARE 3 MEMORY TESTS PROVIDED. TESTS 2 AND 3 ARE TESTS OF MEMORY ADDRESSING CAPABILITY DURING DISK NPR'S, AND TEST 4 IS AN NPR/MEMORY DATA PATTERN TEST. DURING TESTING, MEMORY MANAGEMENT WILL BE ENABLED IF INSTALLED, AND ALL OPERATIONS WILL BE PERFORMED IN KERNAL MODE, WITH ADDRESS RELOCATION BEING DONE THROUGH MANIPULATION OF KERNAL I PAGE ADDRESS REGISTERS.

9.2.1 TEST 2 - NPR/MEMORY WORD ADDRESSING TEST

STARTING AT THE FIRST ADDRESS OF THE NEXT 1 K MEMORY BLOCK BEYOND THE END OF THE READ/WRITE DATA BUFFER (RWBUF), WRITE UNIQUE NUMBERS (STARTING WITH 1) INTO EACH OF UP TO 64K WORDS (DEPENDING ON THE AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES WITHIN THE 64K BLOCK. NEXT WRITE THE 64K WORDS (MAX) ONTO DISK AT FC, FS, FT (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW). THEN ZERO THE ENTIRE 64K BLOCK IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING VIRTUAL (IF MEM. MANAGEMENT) AND PHYSICAL ADDRESSES, AS WELL AS THE GOOD AND BAD DATA, FOR UP TO THE FIRST 10 (DECIMAL) FAILING LOCATIONS.

IF MEMORY MANAGEMENT IS PROVIDED AND THERE IS ADDITONAL MEMORY TO TEST, REPEAT THE ABOVE ADDRESSING TEST FOR EACH OF THE REMAINING 64K PHYSICAL MEMORY BLOCKS.

9.2.2 TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST

IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06 NPR'S IS TESTED.

THE TEST BEGINS BY WRITING A SINGLE NUMBER INTO ALL LOCATIONS OF EACH MEMORY BLOCK OF UP TO 64K WORDS. THIS NUMBER UNIQUELY DEFINES EACH BLOCK. THEN WRITE EACH BLOCK ONTO DISK AT ADDRESS FC, FS, FT (SCALED, IF NECESSARY TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD SECTOR FILE). ZERO THE BLOCK IN MEMORY, AND THEN READ THE DATA BACK INTO THE PROPER PHYSICAL ADDRESSES, DO THIS FOR ALL 64K BLOCKS, AND THEN COMPARE THE CONTENTS OF ALL THE MEMORY TO EXPECTED VALUES. TYPE FAILURES AS DESCRIBED ABOVE FOR THE FIRST 10(DEC) FAILING LOCATIONS IN EACH 64K MEMORY BLOCK.

9.2.3 TEST 4 - NPR/MEMORY DATA PATTERN TEST

IN THIS TEST, ALL PHYSICAL MEMORY LOCATIONS ABOVE THE PROGRAM (NOT INCLUDING UPPER UNIBUS DEVICE ADDRESSES) ARE EXERCISED WITH UP TO 15

1050
1051

DATA PATTERNS, AS CHOSEN FROM THE FIRST 14 PATTERNS DESCRIBED IN
SECTION 8.3, PLUS PATTERN 15 (USER DEFINED PATTERN). THE DATA

1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107

PATTERNS DESIRED FOR THIS TEST ARE CHOSEN SEPARATELY, HOWEVER, AND THEY DEFAULT TO PATTERNS 8, 9, 10, AND 11.

MEMORY IS TESTED IN BLOCKS OF 64K, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE PROGRAM. EACH BLOCK OF UP TO 64K IS LOADED WITH DATA, WRITTEN ONTO DISK AT FC, FS, FT, LOADED WITH ZEROS, AND READ BACK AND COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN, INCLUDING AN OPTIONAL PATTERN CHOSEN BY THE USER.

9.3 TEST 5 - UNIBUS CONTENTION TEST

THE PURPOSE OF THIS TEST IS TO EXERCISE THE NPR CONTROL LOGIC WITHIN THE CONTROLLER IN SUCH A MANNER THAT A CONTENTION FOR MEMORY CYCLES ARISES BETWEEN THE PROCESSOR AND THE RK06.

FIRST, A WRITE DATA IS BEGUN ON CYLINDER FC, (SCALED TO AVOID PACK OVERFLOW) AT SECTOR 0, WITH THE BUS ADDRESS INCREMENT INHIBIT BIT SET IN THE CONTROLLER. USING A WORD COUNT OF 12000(OCT) OR 13000(OCT) (DEPENDING ON THE FORMAT), THE ENTIRE TRACK IS WRITTEN WITH THE FIRST WORD OF USER DEFINED PATTERN 15 (IF SELECTED) OR THE FIRST WORD OF PATTERN 13 (BY DEFAULT). THEN, AN INSTRUCTION LOOP IS ENTERED IN WHICH A NON-EXISTENT UNIBUS DEVICE ADDRESS IS REFERENCED ONCE PER PASS. THIS WILL RESULT IN A PROCESSOR SEIZURE OF THE UNIBUS, FOR 5-20 MICRO-SEC (DEPENDING ON THE PROCESSOR), FOLLOWED BY A NON-EXISTENT MEMORY PROCESSOR TRAP, ONCE PER PASS. THUS, FOR EACH PASS THROUGH THE INSTRUCTION LOOP, THE CONTROLLER IS FORCED TO LOSE FROM ONE TO FOUR NPR BUS CYCLES. THE EXECUTION TIME OF THE LOOP DETERMINES THE REPETITION RATE OF THE NON-EXISTENT MEMORY REFERENCES, AND THIS IS CHOSEN TO ALLOW ENOUGH NPR'S SO THAT THERE SHOULD BE NO DATA LATE ERRORS IN A FAULT-FREE CONTROLLER. THE DATA IS THEN WRITE CHECKED FOR VERIFICATION, AND THE ABOVE TEST IS REPEATED USING A READ DATA COMMAND. NO ERRORS ARE EXPECTED.

THE ABOVE OPERATIONS ARE REPEATED 15 MORE TIMES, USING THE NEXT WORD OF THE PROPER PATTERN (13 OR 15) EACH TIME, TO WRITE AND READ THE ENTIRE TRACK.

9.4 TEST 6 - MULTI-DRIVE INTERFERENCE TEST

THIS TEST PERFORMS MULTI-DRIVE POSITIONING OPERATIONS, WHILE A LARGE DATA TRANSFER IS IN PROGRESS ON THE DRIVE UNDER TEST, FOR THE PURPOSE OF DETECTING PROBLEMS OF CONCURRENT DRIVE OPERATION. THE TEST IS RUN ONLY IF THERE IS MORE THAN ONE DRIVE ON THE SUBSYSTEM.

THE TEST PROCEEDS AS FOLLOWS: IT IS FIRST DETERMINED WHICH DRIVE(S) (BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH. NEXT, A SEEK IS DONE TO CYLINDER FC (SCALED, IF NECESSARY) ON THE DRIVE UNDER TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH TO A DIFFERENT

K03

DZR6N-B - RK611: RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 37
DZR6NB.CMB

1108
1109

PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE FROM 10-67
MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN, A WRITE DATA

1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165

IS BEGUN ON THE DRIVE UNDER TEST, AT THE CURRENT CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616 (DEC) WORDS IF 20 SECTOR FORMAT, OR 17,152 (DEC) WORDS IF 22 SECTOR FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA. THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN THAT TIME, ALL THE OTHER DRIVES SHOULD HAVE COMPLETED POSITIONING. NEXT, A WRITE CHECK IS DONE TO VERIFY THE DATA WRITTEN ON THE DRIVE UNDER TEST, AND A CHECK IS MADE TO INSURE THAT ALL OTHER DRIVES WHICH PERFORMED SEEKS WERE ABLE TO COMPLETE THEM SUCCESSFULLY, BY CHECKING ATTENTION BITS AND COMPARING DRIVE CYLINDER ADDRESSES TO EXPECTED VALUES.

10.0 ERROR REPORTING

10.1 COMMON ERRORS

THE FOLLOWING IS A LIST OF COMMON ERROR MESSAGES WHICH ACCOMPANY ERROR TYPEOUTS FROM THE SUBSYSTEM VERIFICATION PROGRAM. THE ERRORS ARE SELF-EXPLANATORY.

1. UNIBUS PARITY ERROR
2. NON-EXISTANT MEMORY ERROR
3. NON-EXISTANT DRIVE ERROR
4. UNIT FIELD ERROR
5. SUBSYSTEM TIMEOUT
6. SERCON PARITY ERROR
7. DRIVE DETECTED PARITY ERROR
8. AC LOW
9. SPEED LOSS
10. ILLEGAL FUNCTION ERROR
11. PROGRAMMING ERROR
12. NON-EXECUTABLE FUNCTION ERROR
13. DRIVE TYPE ERROR
14. FORMAT ERROR
15. WRITE LOCK ERROR
16. DRIVE UNSAFE ERROR
17. SEEK INCOMPLETE ERROR
18. CYLINDER OVERFLOW ERROR
19. ILLEGAL CYLINDER ADDRESS ERROR
20. DRIVE OFF TRACK
21. DRIVE TIMING ERROR
22. DATA LATE ERROR
23. CONTROLLER TIMEOUT ERROR
24. OPERATION INCOMPLETE ERROR
25. HEADER VRC ERROR
26. DATA CHECK ERROR
27. WRITE CHECK ERROR
28. DATA MISCOMPARE
29. NO DRIVE RESPONSE - UFE AND NXD

M03

1156
1167

- 30. DRIVE ERROR WILL NOT CLEAR
- 31. DRIVE STATUS CHANGE WILL NOT CLEAR

1168
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1177
 1178
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218
 1219
 1220
 1221
 1222
 1223

- 32. ATTENTION BUT NO STATUS CHANGE OR FAULT
- 33. ATTENTION BUT DRIVE NOT AVAILABLE
- 34. ERROR WHILE GATHERING DRIVE STATUS
- 35. MULTIPLE DRIVE SELECT
- 36. HEADER COMPARE ERROR
- 37. ERROR IN RECALIBRATE FOR RECOVERY
- 38. CLEAR CONTROLLER DID NOT CLEAR ERROR
- 39. NO ATTENTION IN ATTENTION SUMMARY REGISTER
- 40. UNSOLICITED ATTENTION
- 41. UNEXPECTED DATA TYPE ERROR
- 42. ATTENTION DID NOT RESET WITH CLEAR
- 43. SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
- 44. DATA LATE WHEN UNLOADING HEADER
- 45. CONTROLLER ERROR WHEN DRIVER SERVICING
- 46. RETRY UNSUCCESSFUL
- 47. BAD SECTOR ERROR ON SECTOR NOT LISTED BAD

10.2 ERROR HANDLING

ERRORS REPORTED BY THE PROGRAM CONSIST OF COMMON FAILURES RESULTING FROM ATTEMPTED SUBSYSTEM FUNCTIONS, AS WELL AS CERTAIN ERRORS UNIQUE TO PARTICULAR TESTS. EACH ERROR PRINTOUT CONSISTS OF AN ERROR DESCRIPTION AND TEST NUMBER, FOLLOWED BY HEADER LINES, COLUMN HEADINGS, AND COLUMNS OF REGISTER CONTENTS IN OCTAL. AS MUCH MEANINGFUL REGISTER DATA AS POSSIBLE (FOR EXAMPLE, RK611 REGISTERS) ARE REPORTED IN A GIVEN ERROR. IN THE CASE OF A DATA MISCOMPARE, FOR INSTANCE, THE NUMBER GOOD DATA WORD, BAD DATA WORD, PHYSICAL MEMORY ADDRESS, VIRTUAL ADDRESS, AND MEMORY MANAGEMENT REGISTER CONTENTS (IF PRESENT) ARE REPORTED. OTHER ERROR REPORTS MAY CONSIST OF A SINGLE DESCRIPTIVE LINE.

10.3 ERROR PRINTOUT EXAMPLES

EXAMPLE 1:

** WRITE CHECK ERROR
 TEST 2

PREVIOUS COMMAND:

DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
000000	000121	000016	000001	000000	175000
HI BA	LO BA				
000000	061566				

CURRENT COMMAND:

ERR PC	DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
041154	000000	000131	000016	000001	000006	175000
HI BA	LO BA					
000000	061566					

804

03-NOV-76 21:43 PAGE 41
RAC1: RAC6 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 41

11111111

PAGE ADDRESS OF ERROR(S):
CYLNDR TRACK SECTOR

000016 000001 000006

EXAMPLE 2:

** DATA MISCOMPARE
TEST 2

PREVIOUS COMMAND:

DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
000000	000131	000016	000001	000000	175000
HI BA	LO BA				
000000	074000				

CURRENT COMMAND:

ERR PC	DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
033156	000000	000121	000016	000001	000000	175000
HI BA	LO BA					
000000	074000					

PACK ADDRESS OF ERROR(S):

CYLNR	TRACK	SECTOR
000016	000001	000000

WD #	GOOD	BAD	HI PHY	LO PHY	VRT AD	KIPAR6
000400	000401	125252	000000	075000	155000	000600
000401	000402	125252	000000	075002	155002	000600
000402	000403	125252	000000	075004	155004	000600
000403	000404	125252	000000	075006	155006	000600
000404	000405	125252	000000	075010	155010	000600
000405	000406	125252	000000	075012	155012	000600
000406	000407	125252	000000	075014	155014	000600
000407	000410	125252	000000	075016	155016	000600
000410	000411	125252	000000	075020	155020	000600
000411	000412	125252	000000	075022	155022	000600

11.0 RK06 HEAD ALIGNMENT AID

THIS PROGRAM IS PROVIDED AS A SOFTWARE AID TO HEAD ALIGNMENT OF THE RK06 DRIVE, TO BE USED IN CONJUNCTION WITH THE RK06 FIELD TEST BOX, OR OTHER SUITABLE INDICATOR OF HEAD ALIGNMENT. THE PROGRAM OPERATES IN TWO MODES. THE FIRST IS MANUAL SELECT MODE, WHICH SELECTS A SPECIFIC DRIVE AND HEAD TO BE ALIGNED BY TTY INPUT AT THE CONSOLE. THE SECOND MODE IS AUTO-SELECT MODE, WHICH ALLOWS REMOTE DRIVE AND HEAD SELECTION THROUGH OPERATION OF DRIVE DUAL-PORT SELECT SWITCHES. IN EITHER MODE THE OPERATOR HAS THE OPTION TO EITHER PERFORM HEAD ALIGNMENT, VERIFY ALIGNMENT, OR REQUEST UP TO FIVE MINUTES OF RANDOM SEEK EXERCISES TO BE PERFORMED ON THE SPECIFIED DRIVE (S).

1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281

004

1292
1293

11.1 HARDWARE REQUIREMENTS

1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339

TO PERFORM HEAD ALIGNMENT, THE ALIGNMENT INDICATING DEVICE MUST BE CONNECTED TO THE DRIVE VIA THE ALIGNMENT CABLE WHICH ATTACHES TO THE RK06 READ/WRITE MODULE. THE ALIGNMENT INFORMATION FROM THE DRIVE MUST BE DISPLAYED ON A METER OR OTHER INDICATOR. EACH DRIVE TO BE ALIGNED MUST BE LOADED WITH AN RK06 ALIGNMENT CARTRIDGE, AND MUST BE WRITE-PROTECTED. UP TO EIGHT DRIVES PER CONTROLLER MAY BE ALIGNED, AND EACH DRIVE MAY BE OPERATED IN SINGLE-PORT MODE ONLY. FOR THE PURPOSE OF THE HEAD ALIGNMENT AND ALL SWITCH REGISTER BITS (HARDWARE OR SOFTWARE) ARE IGNORED.

11.2 OPERATIONAL MODES

THE PROGRAM COMMUNICATES WITH THE OPERATOR DOING THE ALIGNMENT THROUGH CONSOLE DIALOGUE. WHEN THE PROGRAM IS STARTED AT ADDRESS 224(OCTAL), IT TYPES IDENTIFICATION:

" *** RK06 HEAD ALIGNMENT AID ***
TYPE H(CR) FOR INFORMATION- OTHERWISE, TYPE (CR) "

NEXT, THE PROGRAM REQUESTS THE DESIRED MODE OF OPERATION:

" MANUAL OR AUTO MODE (M OR A) ? "

THE OPERATOR MAKES THE SELECTION, AND ENTERS ONE OF THE FOLLOWING MODES.

11.2.1 MANUAL SELECT MODE

IN THIS MODE, THE DRIVE(S) TO BE ALIGNED AND THE HEAD(S) TO BE SELECTED ARE SPECIFIED BY TTY INPUT. THIS MODE IS USEFUL FOR SYSTEMS WITH FEW DRIVES, WHICH ARE IN CLOSE PROXIMITY TO THE CONSOLE, AS IN TYPICAL FIELD INSTALLATIONS.

THE PROGRAM FIRST ECHOS THE MODE:

"* MANUAL SELECT MODE *"
AND TYPES "ENTER DRIVE NO. (0-7):".

THE OPERATOR TYPES THE DRIVE NUMBER, THE PROGRAM TYPES THE DRIVE SERIAL NUMBER, AND THEN ASKS "ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?". THE OPERATOR RESPONDS, AND THE PROGRAM ENTERS THE APPROPRIATE SUB-MODE.

11.2.1.1 MANUAL SELECT ALIGNMENT

ALIGNMENT IN MANUAL MODE PROCEEDS AS FOLLOWS. THE PROGRAM TYPES:

F04

DZ6N-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 45
DZ6N8.CMB

:340
1341

"* MANUAL SELECT ALIGNMENT *," FOLLOWED BY
" ENTER HEAD NO. (0-2):"

1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397

THE OPERATOR TYPES THE DESIRED HEAD NUMBER, AND THE PROGRAM UNLOADS THE HEADS ON THE DRIVE, IN ANTICIPATION OF OPERATOR MOUNTING OF THE HEAD ALIGNMENT TOOL. THE PROGRAM TYPES "TYPE<R> WHEN READY:". AFTER THE TOOL HAS BEEN MOUNTED THE OPERATOR TYPES <R>, AND THE PROGRAM LOADS HEADS AND RESPONDS WITH:

"HEADS POSITIONED AT CYLINDER 365(OCT)"
AND "HEAD X SELECTED"

THE POSITIONING TO CYLINDER 365 IS DONE IN SINGLE INCREMENT SEEKS, TO AVOID EXCESSIVE MOMENTUM ON THE POSITIONER, WHILE THE ALIGNMENT TOOL IS INSTALLED. THE OPERATOR MAY NOW PROCEED TO ALIGN THIS HEAD, AND THE PROGRAM LOOPS BACK TO ASK FOR ANOTHER HEAD NUMBER ON THIS DRIVE. IF THE OPERATOR TYPES CONTROL Z (↑Z) THE PROGRAM WILL LOOP BACK TO ASK FOR "ALIGN OR EXERCISE ?" (STILL IN THE SAME MODE). IF CONTROL C (↑C) IS TYPED, THE PROGRAM LOOPS FURTHER BACK TO ASK FOR A NEW DRIVE NUMBER (STILL IN THE SAME MODE). IF CONTROL R (↑R) IS TYPED, THE PROGRAM EXITS FROM THE CURRENT MODE, AND DOES A COMPLETE RESTART.

11.2.1.2 MANUAL SELECT VERIFY

VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

"* MANUAL SELECT VERIFY *" , FOLLOWED BY
"ENTER HEAD NO. (0-2) :"

THE VERIFY MODE IS IDENTICAL TO THE ALIGNMENT MODE, EXCEPT THAT UNLOADING AND LOADING OF HEADS IS NOT DONE BETWEEN HEAD SELECTIONS. THEREFORE, IN THIS MODE, THE OPERATOR MAY SELECT HEADS FOR QUICK ALIGNMENT VERIFICATION (USING THE ALIGNMENT METER), WITHOUT THE DELAYS OF UNLOADING AND LOADING HEADS. IN THIS MODE, THE OPERATOR WILL NOT BE ASKED TO "TYPE <R> WHEN READY".

11.2.1.3 MANUAL SELECT EXERCISE

WHEN THE EXERCISE SUB-MODE OF MANUAL SELECT MODE IS ENTERED, THE PROGRAM UNLOADS HEADS ON THE SELECTED DRIVE, AND TYPES "TYPE <R> WHEN READY". THE OPERATOR MUST THEN REMOVE THE ALIGNMENT TOOL PRIOR TO THE RANDOM SEEKS WHICH FOLLOW, AND ALLOW EXERCISES TO PROCEED BY TYPING <R>. THE PROGRAM THEN LOADS HEADS, AND RESPONDS WITH:

"* SEEK EXERCISES IN PROGRESS ON DRIVE X *".

AT THIS POINT, 7500 RANDOM SEEKS ARE BEGUN ON THE DRIVE, WHICH LASTS FOR ABOUT FIVE MINUTES. UPON COMPLETION OF THESE EXERCISES, THE PROGRAM RETURNS TO ASK FOR "ALIGN OR EXERCISE ?" (STILL IN MANUAL MODE) ON THIS DRIVE. WHILE IN EXERCISE SUB-MODE, THE CHARACTERS (↑Z), (↑C), AND (↑R) PERFORM THE SAME FUNCTIONS AS DESCRIBED IN SECTION 11.2.1.1.

1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453

11.2.2 AUTO SELECT MODE

IN THIS MODE, ALL SELECTION OF DRIVES AND HEADS FOR ALIGNMENT IS DONE BY SEQUENTIAL OPERATION OF DRIVE PORT SELECT SWITCHES. THIS ALLOWS COMMUNICATION WITH THE PROGRAM TO BE REMOTE FROM THE CONSOLE. THUS AUTO SELECT MODE IS WELL SUITED TO THE MANUFACTURING TEST ENVIRONMENT, OR TO A FIELD INSTALLATION, WHICH HAS A NUMBER OF REMOTE DRIVES. THE PROGRAM FIRST ECHOS THE MODE:

" * AUTO SELECT MODE * ", AND ASKS
"ALIGN OR EXERCISE (A OR E) ?".

THE OPERATOR RESPONDS, AND THE PROGRAM ENTERS THE APPROPRIATE SUB-MODE.

11.2.2.1 AUTO SELECT ALIGNMENT

ALIGNMENT IN AUTO SELECT MODE PROCEEDS AS FOLLOWS. THE PROGRAM TYPES:

" * AUTO SELECT ALIGNMENT * ".

THE PROGRAM CONSTANTLY SCANS ALL DRIVES 0-7 TO DETERMINE THE EXISTENCE OF EACH. WHENEVER A DRIVE IS DESELECTED BY THE OPERATION OF ITS PORT SELECT SWITCH, THE PROGRAM RECEIVES A NON-EXISTENT DRIVE INDICATION WHEN IT SELECTS THAT DRIVE. THEN, WHEN THE PROGRAM DETERMINES THAT A DRIVE WHICH WAS PREVIOUSLY OFF-LINE (DESELECTED) HAS JUST BECOME ON-LINE (SELECTED), IT PREPARES FOR ALIGNMENT ON THAT DRIVE. IN SUMMARY, THE OPERATOR SELECTS A DRIVE FOR ALIGNMENT BY DEPRESSING AND THEN RELEASING THE PORT SELECT SWITCH (FOR THE PORT IN USE) ON THAT DRIVE.

THE PROGRAM RECOGNIZES THE DRIVE SELECTED, AND CHECKS IT TO BE SURE THAT IT IS WRITE-LOCKED. THE PROGRAM THEN TYPES:

"DRIVE X SELECTED"

AND THEN IT UNLOADS THE HEADS ON THAT DRIVE, SO THAT THE OPERATOR CAN MOUNT THE HEAD ALIGNMENT TOOL. WHEN IT IS MOUNTED THE OPERATOR MUST DEPRESS AND RELEASE THE PORT SWITCH AGAIN, AND THE PROGRAM LOADS HEADS, SEEKS IN SINGLE INCREMENTS OUT TO THE ALIGNMENT CYLINDER (365 OCT.), AND TYPES:

"HEADS POSITIONED AT CYLINDER 365(OCT.)
HEAD 0 SELECTED".

AFTER THE OPERATOR ALIGNS HEAD 0, HE MAY PROCEED TO HEAD 1, BY DEPRESSING AND RELEASING THE PORT SELECT SWITCH. WHEN HE DOES, THE HEADS WILL BE UNLOADED AGAIN, AND THE OPERATOR MAY MOVE THE ALIGNMENT TOOL TO HEAD 1, AND UPON DEPRESSING AND RELEASING THE PORT SWITCH AGAIN, THE HEADS WILL BE LOADED, THE PROGRAM WILL SEEK TO CYLINDER 365(OCT), AND TYPE:

1454
1455

"HEADS POSITIONED AT CYLINDER 365(OCT)

1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

HEAD 1 SELECTED".

THIS PROCESS MAY BE CONTINUED WITH HEADS BEING SELECTED IN THE ORDER 0, 1, 2, 0, 1, 2, 0, ETC. UNTIL THE OPERATOR HAS COMPLETED THE ALIGNMENT.

WHEN THE OPERATOR COMPLETES ALIGNMENT ON THIS DRIVE, HE SHOULD REMOVE THE ALIGNMENT TOOL, AND SWITCH THE DRIVE ON-LINE. IF HE THEN WISHES TO SELECT ANOTHER DRIVE, HE MUST SWITCH THE DESIRED DRIVE OFF-LINE AND THEN ON-LINE AGAIN TO INITIATE SELECTION. WHEN ALL DESIRED DRIVES HAVE BEEN ALIGNED THE OPERATOR TYPES ↑Z OR ↑R, AS DESCRIBED IN SECTION 11.2.1.1. BEFORE LEAVING ALIGNMENT MODE AND PROCEEDING TO DO SEEK EXERCISES, THE OPERATOR MUST BE SURE TO REMOVE THE ALIGNMENT TOOL(S) FROM ALL DRIVE(S).

NOTE

THE ↑C FUNCTION DOES NOT APPLY IN AUTO MODE.

11.2.2.2 AUTO SELECT VERIFY

VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

"* AUTO SELECT VERIFY *

THE AUTO SELECT VERIFY MODE IS IDENTICAL TO THE AUTO ALIGNMENT MODE, EXCEPT THAT UNLOADING AND LOADING OF HEADS IS NOT DONE BETWEEN HEAD SELECTIONS. THEREFORE, IN THIS MODE, THE OPERATOR MAY SELECT HEADS FOR QUICK ALIGNMENT VERIFICATION (USING THE ALIGNMENT METER), WITHOUT THE DELAY OF UNLOADING AND LOADING HEADS.

11.2.2.3 AUTO SELECT EXERCISE

IN AUTO SELECT EXERCISE SUB MODE, RANDOM SEEK EXERCISES ARE PERFORMED UPON ALL DRIVES WHICH ARE ON-LINE. DRIVES ARE EXERCISED SEQUENTIALLY, STARTING WITH DRIVE 0. THE OPERATOR SPECIFIES EITHER LONG EXERCISES (7500 SEEKS- 5 MINUTES) OR SHORT EXERCISES (1500 SEEKS- 1 MINUTE) TO BE PERFORMED UPON EACH DRIVE.

AUTO SELECT EXERCISE MODE IDENTIFIES ITSELF AS FOLLOWS:

" * AUTO SELECT EXERCISES * "
FOLLOWED BY "SHORT OR LONG (S OR L) ?".

THE OPERATOR TYPES HIS RESPONSE , AND THE SELECTED SEEK EXERCISES ARE

K04

DZRSN-B - RK611-RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 50
DZRSNB.CME

1512
1513

PERFORMED ON EACH DRIVE. UPON COMPLETION OF THESE EXERCISES, THE
PROGRAM RETURNS TO ASK FOR "ALIGN OR EXERCISE?" (STILL IN AUTO MODE).

1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567

WHILE IN EXERCISE SUB-MODE THE CHARACTERS ↑Z AND ↑R PERFORM THE SAME FUNCTIONS AS DESCRIBED IN SECTION 11.2.1.1.

NOTE

BEFORE REQUESTING AUTO-EXERCISE MODE, THE OPERATOR MUST REMOVE THE ALIGNMENT TOOL (WHILE STILL IN AUTO-ALIGNMENT MODE) WHILE THE HEADS ARE UNLOADED.

11.3 ALIGNMENT AID ERROR MESSAGES

1. "DRIVE X NOT READY! PLEASE START IF DESIRED, THEN PRESS 'CONT' ON CPU WHEN READY." - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS FOUND NOT READY.

2. "DRIVE X NOT WRITE-LOCKED! PLEASE SET WRITE LOCK SWITCH. PRESS 'CONT' ON CPU WHEN READY". - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS FOUND NOT TO BE WRITE-PROTECTED.

3. "PLEASE LOAD ALIGNMENT CARTRIDGE ON DRIVE X! PRESS 'CONT' ON CPU WHEN READY." - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS NOT LOADED WITH AN ALIGNMENT CARTRIDGE.

4. "MULTIPLE DRIVES AUTO-SELECTED! PRESS 'CONT' TO RESTART." - THIS INDICATES THAT IN AUTO ALIGNMENT MODE MORE THAN ONE OF THE DRIVES WERE SELECTED FOR ALIGNMENT SIMULTANEOUSLY.

5. "CANNOT READ BAD SECTOR FILE ON DRIVE X! PRESS 'CONT' ON CPU TO RESTART". - THIS INDICATES THAT A READ ERROR WAS ENCOUNTERED WHILE THE PROGRAM WAS ATTEMPTING TO IDENTIFY THE CARTRIDGE ON DRIVE X AS AN ALIGNMENT CARTRIDGE.

6. "? X" - THIS IS TYPED BY THE PROGRAM IN RESPONSE TO THE INPUT OF AN INVALID PARAMETER, SHOWN HERE AS X. THE OPERATOR SHOULD NOW ENTER THE PROPER VALUE.

APPENDIX A

SAMPLE ADDRESS 200 DEFAULT RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART 2, ON A SINGLE DRIVE (DRIVE 0), FOR THE FIRST PASS OF THE PROGRAM (QUICK-VERIFY PASS). IN THIS MODE, ALL MESSAGES APPEARING HERE WERE OUTPUTS--NO OPERATOR INPUTS ARE PROVIDED ON AN ADDRESS 200 RUN.

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2

LAST PHYS MEM ADR = 377776

- DRIVE 1 NON-EXISTENT
- DRIVE 2 NON-EXISTENT
- DRIVE 3 NON-EXISTENT
- DRIVE 4 NON-EXISTENT
- DRIVE 5 NON-EXISTENT
- DRIVE 6 NON-EXISTENT
- DRIVE 7 NON-EXISTENT

DRIVE(S) = 0

TESTING DRIVE 0
DRIVE SER. NO. 8
CART. SER. NO. 334

OFFSET-TO-FAILURE MEASUREMENTS:

TRACK	CYLN	SECT	+OFST (UIN)	-OFST (UIN)
0	0	0	NONE	NONE
00	0	12	NONE	1175
00	631	0	NONE	NONE
00	631	12	NONE	NONE
11	0	0	1050	NONE
11	0	12	NONE	1200
11	631	0	NONE	NONE
11	631	12	NONE	NONE
22	0	0	NONE	NONE
22	0	12	NONE	NONE
22	631	0	NONE	NONE
22	631	12	NONE	NONE

END PASS # 1

1568
1569
1570
1571
1572
1573
1574
1575
1575
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622

APPENDIX B

SAMPLE ADDRESS 204 RUN

1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART 2, ON A SINGLE DRIVE (DRIVE 0), WITH PARTICULAR TESTS SELECTED TO BE RUN FOR SPECIFIED NUMBERS OF TIMES, WITH SPECIFIED PARAMETERS, AND A SPECIFIED NO. OF PROGRAM PASSES. SPECIFICALLY, IT WAS DESIRED THAT TEST 1 BE RUN ONCE, TEST 4 ONCE, AND TEST 6 ONCE. THESE TESTS WERE SPECIFIED TO RUN FOR 1 PROGRAM PASS, USING PARAMETERS SPECIFIED BY THE OPERATOR. IN THIS CASE, SECTOR ADDRESS LIMITS (S2 AND S3) AND DATA PATTERN (PT) WERE SPECIFIED AS NON-DEFAULT VALUES.

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION:PART 2

LAST PHYS MEM ADR=377776

PARAMETER INPUT MODE

RK06 BUS ADR = 177440 NEW =
RK06 VEC ADR = 210 NEW =
RK06 PRIORITY = 5 NEW =

DRIVE(S) = 0

*

L = LIST TESTS
C = CHANGE TESTS
I = INPUT PARAMETERS AND RUN TESTS

ENTER L,C, OR I

* C

TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>

*

TEST	ITERATIONS
1	* 1
2	* 0
3	* 0
4	* 0\0\1

1679
1663

617

2 * 0
100 * 1

1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736

ENTER L,C, OR I

* L

TEST ITERATIONS

1
2
3
4
5
6

0
0
0
0
0
0

ENTER L,C, OR I

* I

T = TYPE PARAMETER LIST
O = OPEN PARAMETER LIST
S = SET INDIVIDUAL PARAM
R = RUN TESTS

ENTER T,O,S, OR R

* T

FC=0
LC=632
FT=0
LT=2
SQ=0
S1=23
S2=0
S3=25
PT=0
CS=0
ST=0

ENTER T,O,S, OR R

* S

> S2=4
> SV=20
SV=20?
> S3=20
> PT=100000
TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR>

* M

MODIFY USER-DEFINED PATTERN 15:

WORD 00 = 072307 * 123456!

> !Z

ENTER T,O,S, OR R

* T

FC=0
LC=632
FT=0
LT=2

DOS

D2R6N-8 - R4611 RK06 SUBSYSTEM VERIFICATION : PART 2
D2R6NB.CMB

MACY11 27(732) 03-NOV-76 21:43 PAGE 56

1737
1738

SC=0
SI=23

1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793

S2=4
S3=20
PT=100000
CS=0
ST=0

USER-DEFINED PATTERN 15:

WORD 00 = 123456
WORD 01 = 123456
WORD 02 = 123456
WORD 03 = 123456
WORD 04 = 123456
WORD 05 = 123456
WORD 06 = 123456
WORD 07 = 123456
WORD 10 = 123456
WORD 11 = 123456
WORD 12 = 123456
WORD 13 = 123456
WORD 14+Z
= 123456

ENTER T,O,S, OR R
* R

ENTER NO. OF PASSES (1-77777) :
* 1

TESTING DRIVE 0
DRIVE SER. NO. 8
CART. SER. NO. 334

OFFSET-TO-FAILURE MEASUREMENTS:

TRACK	CYLN	SECT	+OFST (UIN)	-OFST (UIN)
0	0	4	NONE	NONE
0	0	20	NONE	NONE
0	631	4	NONE	NONE
0	631	20	NONE	NONE
1	0	4	NONE	NONE
1	0	20	NONE	NONE
1	631	4	NONE	NONE
1	631	20	NONE	NONE
2	0	4	NONE	NONE
2	0	20	NONE	NONE
2	631	4	NONE	NONE
2	631	20	NONE	NONE

END PASS # 1
TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>
*

1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849

APPENDIX C

SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING THE HEAD ALIGNMENT AID, PROVIDED IN SUBSYSTEM VERIFICATION PART 2 (ADDRESS 224 START). MANUAL MODE WAS REQUESTED, AND THE OPERATOR SELECTED DRIVE 0 FOR ALIGNMENT. IN THIS CASE, THE OPERATOR SELECTED HEADS 1, 0, AND 2, IN THAT ORDER, AND THEN REQUESTED RANDOM SEEK EXERCISES TO BE RUN ON THE DRIVE (AFTER REMOVAL OF THE HEAD ALIGNMENT TOOL). THE EXERCISES WOULD HAVE NORMALLY RUN FOR 5 MINUTES BUT THEY WERE PREMATURELY TERMINATED BY THE OPERATOR, BY TYPING (↑Z), IN THIS PARTICULAR RUN. FINALLY, VERIFY MODE WAS REQUESTED, AND HEAD 2 WAS SELECTED BY THE OPERATOR.

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2

*** RK06 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A)?
M

* MANUAL SELECT MODE *

ENTER DRIVE NO. (0-7):
0
DRIVE SER. NO. 8

ALIGN, VERIFY, OR EXERCISE (A, V, OR E)?
A

* MANUAL SELECT ALIGNMENT *

ENTER HEAD NO. (0-2):
1
TYPE <R> WHEN READY:
R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED

ENTER HEAD NO. (0-2):

G05

DZR6N-B - RK611:RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 59
DZR6NB.CMB

1850
1851

0
TYPE (R) WHEN READY:

1852
1853
1354
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892

R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED

ENTER HEAD NO. (0-2):

2
TYPE <R> WHEN READY:

R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED

ENTER HEAD NO. (0-2):

↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?

E
TYPE <R> WHEN READY:

R

*RANDOM SEEK EXERCISES IN PROGRESS ON DRIVE 0 *

↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?

V

* MANUAL SELECT VERIFY *

ENTER HEAD NO. (0-2) :

2
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED

ENTER HEAD NO. (0-2) :

↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?

APPENDIX D

SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING THE HEAD ALIGNMENT AID, PROVIDED IN SUBSYSTEM VERIFICATION PART 2 (ADDRESS 224 START). AUTO MODE WAS REQUESTED, AND THE OPERATOR SELECTED DRIVE 0 FOR ALIGNMENT. IN THIS CASE, HEADS 0,1,2, AND 0 (AGAIN) WERE SELECTED FOR ALIGNMENT (IN AUTO MODE, THE HEADS ARE ALWAYS SELECTED IN THE ORDER 0,1,2,0,1,2, ETC.). BETWEEN HEAD SELECTIONS, THE HEADS WERE UNLOADED TO ALLOW REMOVAL OR INSTALLATION OF THE HEAD ALIGNMENT TOOL. NOTE THAT ALL DRIVE AND HEAD SELECTION (IN AUTO MODE) IS DONE BY THE OPERATOR, AT THE DRIVES, BY OPERATION OF PORT SELECT SWITCHES. AFTER ALIGNING HEADS, THE OPERATOR REQUESTED RANDOM SEEK EXERCISES TO BE RUN (ON ALL DRIVES, IN THIS CASE DRIVE 0). SHORT EXERCISES WERE REQUESTED, WHICH RUN FOR 1 MINUTE PER DRIVE. FINALLY, AUTO SELECT VERIFY WAS REQUESTED, AND HEADS 0,1, AND 2 WERE SELECTED SEQUENTIALLY.

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2

*** RK06 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A)?
A

* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
A

* AUTO SELECT ALIGNMENT *

DRIVE 0 SELECTED

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED

1893
1894
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948

J05

DZR6N-B - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 62
DZR6NB.CMB

1949
1950

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED

1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006

↑Z

* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
E

* AUTO SELECT EXERCISES *

SHORT OR LONG (S OR L) ?
S
EXERCISING DRIVE 0

* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
V

* AUTO SELECT VERIFY *

DRIVE 0 SELECTED

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED

↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
E

000001

PART=1

*** IF "PART" IS DEFINED = 0, PART 1 WILL BE ASSEMBLED. ***
*** IF "PART" IS DEFINED NOT = 0, PART 2 WILL BE ASSEMBLED. ***

.NLIST MC,MD,CND
.LIST ME
.ENABL ABS,AMA
.MCALL .HEADER,.SWRHI,.SWRLO,EQUAT,SETUP,\$CATCH,SWRSU
.MCALL .SCMTAG,\$TYPE,\$STYPOCT,\$POWER,\$TRAP,\$DB20,\$DB2D
.MCALL .SREAD,\$STPDEC,\$RAND,\$SIZE,\$ERROR,\$SAVE
.MCALL .SRAND,\$SUPRS,\$EOP,\$SCOPE,\$KT11
.MCALL .SACT11,\$APTHDR,\$APTTYPE
\$SWR= 167000

167000

.SBTTL STARTING ADDRESSES
*
*

2007
2008
2009
2010
2011
2012 000001
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049 001100
2050
2051
2052
2053
2054 000011
2055 000012
2056 000015
2057 000200
2058 177776
2059
2060 177774
2061 177772
2062 177570

```

;*      200      DEFAULT PARAMETERS FOR TESTS 1-6
;*      204      SELECT PARAMETERS FOR TESTS 1-6
;*      224      HEAD ALIGNMENT AID
;*****
$TN=1
.TITLE DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY DAVE HOFFMAN
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
;*
.SBTTL OPERATIONAL SWITCH SETTINGS
;*
SWITCH          USE
-----          -----
;*      15      HALT ON ERROR
;*      14      LOOP ON TEST
;*      13      INHIBIT ERROR TYPEOUTS
;*      12      REPORT DESCRIPTION ONLY, ON ERRORS
;*      11      INHIBIT ITERATIONS
;*      10      BELL ON ERROR
;*      9       LOOP ON ERROR
;*      8       APPLY RANDOM STALL BETWEEN OPERATIONS
;*      6       REPORT 1 ERROR PER TRANSFER IN TESTS 2-4
;*      1       INHIBIT WRITES IN TEST 1
;*      0       REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 2-4
.SBTTL CONTROL SWITCH SETTINGS (PARAMETER CS)
;*
SWITCH          USE
-----          -----
;*      05      DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
;*      04      TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
;*      01      INHIBIT OFFSET REPORTS IN TEST 1
.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
;*MISCELLANEOUS DEFINITIONS
HT= 11      ;;CODE FOR HORIZONTAL TAB
LF= 12      ;;CODE FOR LINE FEED
CR= 15      ;;CODE FOR CARRIAGE RETURN
CRLF= 200   ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776  ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;;STACK LIMIT REGISTER
PIRQ= 177772  ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570  ;;HARDWARE SWITCH REGISTER
```

M05

```

2063      177570      DDISP= 177570      ;;HARDWARE DISPLAY REGISTER
2064
2065      ;;*GENERAL PURPOSE REGISTER DEFINITIONS
2066      000000      R0= %0      ;;GENERAL REGISTER
2067      000001      R1= %1      ;;GENERAL REGISTER
2068      000002      R2= %2      ;;GENERAL REGISTER
2069      000003      R3= %3      ;;GENERAL REGISTER
2070      000004      R4= %4      ;;GENERAL REGISTER
2071      000005      R5= %5      ;;GENERAL REGISTER
2072      000006      R6= %6      ;;GENERAL REGISTER
2073      000007      R7= %7      ;;GENERAL REGISTER
2074      .EQUIV R6,SP      ;;STACK POINTER
2075      .EQUIV R7,PC      ;;PROGRAM COUNTER
2076
2077      ;;*PRIORITY LEVEL DEFINITIONS
2078      000000      PR0= 0      ;;PRIORITY LEVEL 0
2079      000040      PR1= 40      ;;PRIORITY LEVEL 1
2080      000100      PR2= 100      ;;PRIORITY LEVEL 2
2081      000140      PR3= 140      ;;PRIORITY LEVEL 3
2082      000200      PR4= 200      ;;PRIORITY LEVEL 4
2083      000240      PR5= 240      ;;PRIORITY LEVEL 5
2084      000300      PR6= 300      ;;PRIORITY LEVEL 6
2085      000340      PR7= 340      ;;PRIORITY LEVEL 7
2086
2087      ;;*"SWITCH REGISTER" SWITCH DEFINITIONS
2088      100000      SW15= 100000
2089      040000      SW14= 40000
2090      020000      SW13= 20000
2091      010000      SW12= 10000
2092      004000      SW11= 4000
2093      002000      SW10= 2000
2094      001000      SW09= 1000
2095      000400      SW08= 400
2096      000200      SW07= 200
2097      000100      SW06= 100
2098      000040      SW05= 40
2099      000020      SW04= 20
2100      000010      SW03= 10
2101      000004      SW02= 4
2102      000002      SW01= 2
2103      000001      SW00= 1
2104      .EQUIV SW09,SW9
2105      .EQUIV SW08,SW8
2106      .EQUIV SW07,SW7
2107      .EQUIV SW06,SW6
2108      .EQUIV SW05,SW5
2109      .EQUIV SW04,SW4
2110      .EQUIV SW03,SW3
2111      .EQUIV SW02,SW2
2112      .EQUIV SW01,SW1
2113      .EQUIV SW00,SW0
2114
2115      ;;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
2116      100000      BIT15= 100000
2117      040000      BIT14= 40000
2118      020000      BIT13= 20000
  
```

2119	010000	BIT12=	10000
2120	004000	BIT11=	4000
2121	002000	BIT10=	2000
2122	001000	BIT09=	1000
2123	000400	BIT08=	400
2124	000200	BIT07=	200
2125	000100	BIT06=	100
2126	000040	BIT05=	40
2127	000020	BIT04=	20
2128	000010	BIT03=	10
2129	000004	BIT02=	4
2130	000002	BIT01=	2
2131	000001	BIT00=	1
2132		.EQUIV	BIT09, BIT9
2133		.EQUIV	BIT08, BIT8
2134		.EQUIV	BIT07, BIT7
2135		.EQUIV	BIT06, BIT6
2136		.EQUIV	BIT05, BIT5
2137		.EQUIV	BIT04, BIT4
2138		.EQUIV	BIT03, BIT3
2139		.EQUIV	BIT02, BIT2
2140		.EQUIV	BIT01, BIT1
2141		.EQUIV	BIT00, BIT0
2142			
2143		.*BASIC "CPU" TRAP VECTOR ADDRESSES	
2144	000004	ERRVEC=	4 ; TIME OUT AND OTHER ERRORS
2145	000010	RESVEC=	10 ; RESERVED AND ILLEGAL INSTRUCTIONS
2146	000014	TBITVEC=	14 ; "T" BIT
2147	000014	TRTVEC=	14 ; TRACE TRAP
2148	000014	BPTVEC=	14 ; BREAKPOINT TRAP (BPT)
2149	000020	IOTVEC=	20 ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
2150	000024	PWRVEC=	24 ; POWER FAIL
2151	000030	EMTVEC=	30 ; EMULATOR TRAP (EMT) **ERROR**
2152	000034	TRAPVEC=	34 ; "TRAP" TRAP
2153	000060	TKVEC=	60 ; TTY KEYBOARD VECTOR
2154	000064	TPVEC=	64 ; TTY PRINTER VECTOR
2155	000240	PIRQVEC=	240 ; PROGRAM INTERRUPT REQUEST VECTOR
2156		.SBTTL	MEMORY MANAGEMENT DEFINITIONS
2157			
2158		.*KT11 VECTOR ADDRESS	
2159			
2160	000250	MMVEC=	250
2161			
2162		.*KT11 STATUS REGISTER ADDRESSES	
2163			
2164	177572	SRO=	177572
2165	177574	SR1=	177574
2166	177576	SR2=	177576
2167	172516	SR3=	172516
2168			
2169		.*KERNEL "I" PAGE DESCRIPTOR REGISTERS	
2170			
2171	172300	KIPDR0=	172300
2172	172302	KIPDR1=	172302
2173	172304	KIPDR2=	172304
2174	172306	KIPDR3=	172306

22000
22001
22002
22003
22004
22005
22006
22007
22008
22009
22010
22011
22012
22013
22014
22015
22016
22017
22018
22019
22020
22021
22022
22023
22024
22025
22026
22027
22028
22029
22030

172310
172312
172314
172316

172340
172342
172344
172346
172350
172352
172354
172356

170200
170202
172100
177740
177742
177744

000000

000174
000176

000200
000204
000224
000230
000232

000234
000246
000052
140000
000234
001000

001000

013450
013504
023602

KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316

;*KERNEL "I" PAGE ADDRESS REGISTERS

KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356

MAPL00=170200
MAPH00=170202
MEMCSR=172100
LOERAD=177740
HIERAD=177742
MEMSYS=177744

;; MEMORY CSR REG START ADRS
;; 11/70 MEM LO ERROR ADRS REG
;; 11/70 MEM HI ERROR ADRS REG
;; 11/70 MEM SYSTEM REG

.SBTTL TRAP CATCHER

.=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A "+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

DISPREG: .WORD 0 ;; SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;; SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP @#DFSTRT ;; JUMP TO STARTING ADDRESS OF PROGRAM

.=204
JMP @#PSTART
.=224
JMP @#ASTART

..LOW: .WORD 700
..HIGH: .WORD 1100
.SBTTL ACT11 HOOKS

;HOOKS REQUIRED BY ACT11

SSVPC=.; ;SAVE PC
.=46
SENDAD ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
.=52
.WORD 140000 ;;2)SET LOC.52 TO 140000
.=SSVPC ;; RESTORE PC
.=1000

.SBTTL APT PARAMETER BLOCK

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

.SX=.; ;SAVE CURRENT LOCATION

```

2231
2232 000024 000024
2233 000024 000200
2234 000044 000044
2235 000044 001000
2236 000044 001000
2237
2238
2239
2240 001000
2241 001000 000000
2242 001002 001320
2243 001004 001130
2244 001006 003410
2245 001010 003410
2246 001012 000030
2247 120210
2248 177440
2249 000377

```

```

      =24      ;; SET POWER FAIL TO POINT TO START OF PROGRAM
      200      ;; FOR APT START UP
      =44      ;; POINT TO APT INDIRECT ADDRESS PNTR.
      $APTHDR ;; POINT TO APT HEADER BLOCK
      =.SX     ;; RESET LOCATION COUNTER
;*****
; SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PP01! DIAGNOSTIC
; INTERFACE SPEC.

$APTHD:
$SHIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBAADR: .WORD $MAIL  ;; ADDRESS OF APT MAILBOX (BITS 0-15)
$STSTM:  .WORD 1130   ;; RUN TIM OF LONGEST TEST
$PASTM:  .WORD 3410   ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM:  .WORD 3410   ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
          .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)
AVECT1=120210      ;; RKVEC=210, RKPRI=5
ABASE=177440      ;; RKBAS ADRS
ADEVM=000377      ;; SET DEVICES 0-7 IN MAP

```

2250
2251
2252
2253
2254
2255
2256 001100
2257 001100 000000
2258 001100 000000
2259 001102 000
2260 001103 000
2261 001104 000000
2262 001106 000000
2263 001110 000000
2264 001112 000000
2265 001114 000
2266 001115 001
2267 001116 000000
2268 001120 000000
2269 001122 000000
2270 001124 000000
2271 001126 000000
2272 001130 000000
2273 001132 000000
2274 001134 000
2275 001135 000
2276 001136 000000
2277 001140 177570
2278 001142 177570
2279 001144 177560
2280 001146 177562
2281 001150 177564
2282 001152 177566
2283 001154 000
2284 001155 002
2285 001156 012
2286 001157 000
2287 001160 000000
2288
2289 001162 000000
2290 001164 000000
2291 001166 000000
2292 001170 000000
2293 001172 000000
2294 001174 000000
2295 001176 000000
2296 001200 000000
2297 001202 000000
2298 001204 000000
2299 001206 000000
2300 001210 000000
2301 001212 000000
2302 001214 000000
2303 001216 000000
2304 001220 000000
2305 001222 000000

.SBTTL COMMON TAGS

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

SCMTAG: .=1100

.WORD 0
\$TSTNM: .BYTE 00
\$ERFLG: .BYTE 00
\$ICNT: .WORD 00
\$LPADR: .WORD 00
\$LPERR: .WORD 00
\$ERTTL: .WORD 00
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 00
\$BDADR: .WORD 00
\$GDADR: .WORD 00
\$BDDAT: .WORD 00

\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0

\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566

\$.NULL: .BYTE 0
\$.FILLS: .BYTE 2
\$.FILLC: .BYTE 12
\$.STPFLG: .BYTE 0
\$.REGAD: .WORD 0

\$.REG0: .WORD 0
\$.REG1: .WORD 0
\$.REG2: .WORD 0
\$.REG3: .WORD 0
\$.REG4: .WORD 0
\$.REG5: .WORD 0
\$.REG6: .WORD 0
\$.REG7: .WORD 0
\$.REG10: .WORD 0
\$.REG11: .WORD 0
\$.REG12: .WORD 0
\$.REG13: .WORD 0
\$.REG14: .WORD 0
\$.REG15: .WORD 0
\$.REG16: .WORD 0
\$.REG17: .WORD 0
\$.REG20: .WORD 0

::START OF COMMON TAGS

::CONTAINS THE TEST NUMBER
::CONTAINS ERROR FLAG
::CONTAINS SUBTEST ITERATION COUNT
::CONTAINS SCOPE LOOP ADDRESS
::CONTAINS SCOPE RETURN FOR ERRORS
::CONTAINS TOTAL ERRORS DETECTED
::CONTAINS ITEM CONTROL BYTE
::CONTAINS MAX. ERRORS PER TEST
::CONTAINS PC OF LAST ERROR INSTRUCTION
::CONTAINS ADDRESS OF 'GOOD' DATA
::CONTAINS ADDRESS OF 'BAD' DATA
::CONTAINS 'GOOD' DATA
::CONTAINS 'BAD' DATA
::RESERVED--NOT TO BE USED

::AUTOMATIC MODE INDICATOR
::INTERRUPT MODE INDICATOR

::ADDRESS OF SWITCH REGISTER
::ADDRESS OF DISPLAY REGISTER
::TTY KBD STATUS
::TTY KBD BUFFER
::TTY PRINTER STATUS REG. ADDRESS
::TTY PRINTER BUFFER REG. ADDRESS
::CONTAINS NULL CHARACTER FOR FILLS
::CONTAINS # OF FILLER CHARACTERS REQUIRED
::INSERT FILL CHARS. AFTER A "LINE FEED"
::"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
::CONTAINS THE ADDRESS FROM
::WHICH (\$.REG0) WAS OBTAINED
::CONTAINS ((\$REGAD)+0)
::CONTAINS ((\$REGAD)+2)
::CONTAINS ((\$REGAD)+4)
::CONTAINS ((\$REGAD)+6)
::CONTAINS ((\$REGAD)+10)
::CONTAINS ((\$REGAD)+12)
::CONTAINS ((\$REGAD)+14)
::CONTAINS ((\$REGAD)+16)
::CONTAINS ((\$REGAD)+20)
::CONTAINS ((\$REGAD)+22)
::CONTAINS ((\$REGAD)+24)
::CONTAINS ((\$REGAD)+26)
::CONTAINS ((\$REGAD)+30)
::CONTAINS ((\$REGAD)+32)
::CONTAINS ((\$REGAD)+34)
::CONTAINS ((\$REGAD)+36)
::CONTAINS ((\$REGAD)+40)

2306 001224 000000
2307 001226 000000
2308 001230 000000
2309 001232 000000
2310 001234 000000
2311 001236 000000
2312 001240 000000
2313 001242 000000
2314 001244 000000
2315 001246 000000
2316 001250 000000
2317 001252 000000
2318 001254 000000
2319 001256 000000
2320 001260 000000
2321 001262 000000
2322 001264 000000
2323 001266 000000
2324 001270 000000
2325 001272 000000
2326 001274 000000
2327 001276 000000
2328 001300 000000
2329 001302 000000
2330 001304 000000
2331 001306 000000
2332 001310 177607 000377
2333 001314 077
2334 001315 015
2335 001316 000012
2336
2337
2338
2339
2340
2341 001320
2342 001320 000000
2343 001322 000000
2344 001324 000000
2345 001326 000000
2346 001330 000000
2347 001332 000000
2348 001334 000000
2349 001336 000000
2350 001340
2351 001340 000
2352 001341 000
2353 001342 000000
2354 001344 000000
2355 001346 000000
2356
2357
2358
2359
2360
2361

\$REG21: .WORD 0
\$REG22: .WORD 0
\$REG23: .WORD 0
\$REG24: .WORD 0
\$REG25: .WORD 0
\$REG26: .WORD 0
\$REG27: .WORD 0
\$REG30: .WORD 0
\$REG31: .WORD 0
\$REG32: .WORD 0
\$REG33: .WORD 0
\$REG34: .WORD 0
\$REG35: .WORD 0
\$REG36: .WORD 0
\$REG37: .WORD 0
\$TMP0: .WORD 0
\$TMP1: .WORD 0
\$TMP2: .WORD 0
\$TMP3: .WORD 0
\$TMP4: .WORD 0
\$TMP5: .WORD 0
\$TMP6: .WORD 0
\$TMP7: .WORD 0
\$TMP10: .WORD 0
\$TIMES: 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>
: *****
: .SBTTL APT MAILBOX-ETABLE
: *****
: .EVEN
\$MAIL: ; APT MAILBOX
\$MSGTY: .WORD AMSGTY ; MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL ; FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN ; TEST NUMBER
\$PASS: .WORD APASS ; PASS COUNT
\$DEVCT: .WORD ADEVCT ; DEVICE COUNT
\$UNIT: .WORD AUNIT ; I/O UNIT NUMBER
\$MSGAD: .WORD AMSGAD ; MESSAGE ADDRESS
\$MSGLG: .WORD AMSGLG ; MESSAGE LENGTH
\$ETABLE: ; APT ENVIRONMENT TABLE
\$ENV: .BYTE AENV ; ENVIRONMENT BYTE
\$ENVM: .BYTE AENVM ; ENVIRONMENT MODE BITS
\$SWREG: .WORD ASWREG ; APT SWITCH REGISTER
\$USW: .WORD AUSWR ; USER SWITCHES
\$CPUOP: .WORD ACPUOP ; CPU TYPE, OPTIONS
: *
: * BIT 15-11=CPU TYPE
: * 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
: * 11/70=06, PDQ=07, Q=10
: * BIT 10=REAL TIME CLOCK
: * BIT 9=FLOATING POINT PROCESSOR
: * BIT 8=MEMORY MANAGEMENT

F06

DZR6N-B - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB APT MAILBOX-ETABLE

MACY11 27(732) 03-NOV-76 21:43 PAGE 71

2362	001350	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
2363	001351	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE,BLK#1
2364			.*		MEM.TYPE BYTE -- (HIGH BYTE)
2365			.*		900 NSEC CORE=001
2366			.*		300 NSEC BIPOLAR=002
2367			.*		500 NSEC MOS=003
2368	001352	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS,BLK#1
2369			.*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2370	001354	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
2371	001355	000	\$MTYP2: .BYTE	AMTYP2	::MEM. TYPE,BLK#2
2372	001356	000000	\$MADR2: .WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
2373	001360	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
2374	001361	000	\$MTYP3: .BYTE	AMTYP3	::MEM. TYPE,BLK#3
2375	001362	000000	\$MADR3: .WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
2376	001364	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
2377	001365	000	\$MTYP4: .BYTE	AMTYP4	::MEM. TYPE,BLK#4
2378	001366	000000	\$MADR4: .WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
2379	001370	120210	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
2380	001372	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
2381	001374	177440	\$BASE: .WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
2382	001376	000377	\$DEVN: .WORD	ADEVN	::DEVICE MAP
2383	001400		\$ETEND:		
2384			.MEXIT		

.SBTTL ERROR POINTER TABLE

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

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

SERRTB:

2385					
2386					
2387					
2388					
2389					
2390					
2391					
2392					
2393					
2394					
2395					
2396					
2397					
2398					
2399	001400				
2400					
2401					
2402	001400	056645			
2403	001402	061740			
2404	001404	063524			
2405	001406	063640			
2406					
2407					
2408	001410	056671			
2409	001412	061740			
2410	001414	063524			
2411	001416	063640			
2412					
2413					
2414	001420	056715			
2415	001422	061740			
2416	001424	063524			
2417	001426	063640			
2418					
2419					
2420	001430	056740			
2421	001432	061740			
2422	001434	063524			
2423	001436	063640			
2424					
2425					
2426	001440	056761			
2427	001442	061740			
2428	001444	063524			
2429	001446	063670			
2430					
2431					
2432	001450	057000			
2433	001452	061740			
2434	001454	063524			
2435	001456	063724			
2436					
2437					
2438	001460	057024			
2439	001462	061740			
2440	001464	063524			

```

:ERROR 1 ;UNIBUS PARITY ERROR
EM1
DH100
DT100
DF01

:ERROR 2 ;NON-EXISTANT MEMORY
EM2
DH100
DT100
DF01

:ERROR 3 ;NON-EXISTANT DRIVE
EM3
DH100
DT100
DF01

:ERROR 4 ;UNIT FIELD ERROR
EM4
DH100
DT100
DF01

:ERROR 5 ;SUBSYSTEM TIMEOUT
EM5
DH100
DT100
DF02

:ERROR 6 ;SERCON PARITY ERROR
EM6
DH100
DT100
DF03

:ERROR 7 ;DRIVE DETECTED PARITY ERROR
EM7
DH100
DT100

```

2441	001466	063724	DF03	
2442				
2443			:ERROR 10	
2444	001470	057060	EM10	;AC LOW
2445	001472	061740	DH100	
2446	001474	063524	DT100	
2447	001476	063670	DF02	
2448				
2449			:ERROR 11	
2450	001500	057067	EM11	;SPEED LOSS
2451	001502	061740	DH100	
2452	001504	063524	DT100	
2453	001506	063670	DF02	
2454				
2455			:ERROR 12	
2456	001510	057102	EM12	;ILLEGAL FUNCTION
2457	001512	061740	DH100	
2458	001514	063524	DT100	
2459	001516	063670	DF02	
2460				
2461			:ERROR 13	
2462	001520	057123	EM13	;PROGRAMMING ERROR
2463	001522	061740	DH100	
2464	001524	063524	DT100	
2465	001526	063640	DF01	
2466				
2467			:ERROR 14	
2468	001530	057145	EM14	;NON-EXISTANT FUNCTION
2469	001532	061740	DH100	
2470	001534	063524	DT100	
2471	001536	063670	DF02	
2472				
2473			:ERROR 15	
2474	001540	057173	EM15	;DRIVE TYPE ERROR
2475	001542	061740	DH100	
2476	001544	063524	DT100	
2477	001546	063670	DF02	
2478				
2479			:ERROR 16	
2480	001550	057214	EM16	;FORMAT ERROR
2481	001552	061740	DH100	
2482	001554	063524	DT100	
2483	001556	063670	DF02	
2484				
2485			:ERROR 17	
2486	001560	057231	EM17	;WRITE LOCK ERROR
2487	001562	061740	DH100	
2488	001564	063524	DT100	
2489	001566	063670	DF02	
2490				
2491			:ERROR 20	
2492	001570	057252	EM20	;DRIVE UNSAFE
2493	001572	061740	DH100	
2494	001574	063524	DT100	
2495	001576	063670	DF02	
2496				

2497			.ERROR 21	
2498	001600	057267	EM21	;SEEK INCOMPLETE
2499	001602	061740	DH100	
2500	001604	063524	DT100	
2501	001606	063670	DF02	
2502				
2503			.ERROR 22	
2504	001610	057307	EM22	;CYLINDER OVERFLOW
2505	001612	061740	DH100	
2506	001614	063524	DT100	
2507	001616	063670	DF02	
2508				
2509			.ERROR 23	
2510	001620	057331	EM23	;ILLEGAL CYLINDER
2511	001622	061740	DH100	
2512	001624	063524	DT100	
2513	00. 626	063670	DF02	
2514				
2515			.ERROR 24	
2516	001630	057362	EM24	;DRIVE OFF TRACK
2517	001632	061740	DH100	
2518	001634	063524	DT100	
2519	001636	063670	DF02	
2520				
2521			.ERROR 25	
2522	001640	057402	EM25	;DRIVE TIMING ERROR
2523	001642	061740	DH100	
2524	001644	063524	DT100	
2525	001646	063670	DF02	
2526				
2527			.ERROR 26	
2528	001650	057425	EM26	;DATA LATE
2529	001652	061740	DH100	
2530	001654	063524	DT100	
2531	001656	063670	DF02	
2532				
2533			.ERROR 27	
2534	001660	057437	EM27	;CONTROLLER TIMEOUT
2535	001662	061740	DH100	
2536	001664	063524	DT100	
2537	001666	063670	DF02	
2538				
2539			.ERROR 30	;OPERATION INCOMPLETE
2540	001670	057462	EM30	
2541	001672	061740	DH100	
2542	001674	063524	DT100	
2543	001676	064000	DF05	
2544				
2545			.ERROR 31	
2546	001700	057507	EM31	;HEADER VRC ERROR
2547	001702	061740	DH100	
2548	001704	063524	DT100	
2549	001706	064000	DF05	
2550				
2551			.ERROR 32	
2552	001710	057530	EM32	;DATA CHECK ERROR

2553	001712	061740	DH100	
2554	001714	063524	DT100	
2555	001716	064034	DF07	
2556				
2557			:ERROR 33	
2558	001720	057551	EM33	;WRITE CHECK ERROR
2559	001722	061740	DH100	
2560	001724	063524	DT100	
2561	001726	064070	DF10	
2562				
2563			:ERROR 34	
2564	001730	057573	EM34	;DATA MISCOMPARE(S)
2565	001732	061740	DH100	
2566	001734	063524	DT100	
2567	001736	063764	DF04	
2568				
2569			:ERROR 35	
2570	001740	057613	EM35	;NO DRIVE RESPONSE-UFE & NXD
2571	001742	061740	DH100	
2572	001744	063524	DT100	
2573	001746	063640	DF01	
2574				
2575			:ERROR 36	
2576	001750	057647	EM36	;DRIVE ERROR WILL NOT CLEAR
2577	001752	000000	0	
2578	001754	000000	0	
2579	001756	000000	0	
2580				
2581			:ERROR 37	;DRIVE STATUS CHANGE WILL NOT CLEAR
2582	001760	057702	EM37	
2583	001762	000000	0	
2584	001764	000000	0	
2585	001766	000000	0	
2586				
2587			:ERROR 40	
2588	001770	057745	EM40	;ATTENTION BUT NO STATUS CHANGE OR FAULT
2589	001772	061740	DH100	
2590	001774	063524	DT100	
2591	001776	063670	DF02	
2592				
2593			:ERROR 41	
2594	002000	060011	EM41	;ATTENTION BUT DRIVE NOT AVAILABLE
2595	002002	061740	DH100	
2596	002004	063524	DT100	
2597	002006	063670	DF02	
2598				
2599			:ERROR 42	
2600	002010	060047	EM42	;ATTENTION WHEN NOT EXPECTED
2601	002012	061740	DH100	
2602	002014	063524	DT100	
2603	002016	063670	DF02	
2604				
2605			:ERROR 43	
2606	002020	060077	EM43	;ERROR WHILE GATHERING DRIVE STATUS
2607	002022	061740	DH100	
2608	002024	063524	DT100	

K06

DZR6N-B - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB ERROR POINTER TABLE

MACY11 27(732) 03-NOV-76 21:43 PAGE 76

2609	002026	064134	DF12	
2610				
2611			:ERROR 44	
2612	002030	060354	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2613	002032	061740	DH100	
2614	002034	063524	DT100	
2615	002036	064134	DF12	
2616				
2617			:ERROR 45	
2618	002040	060421	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2619	002042	061740	DH100	
2620	002044	063524	DT100	
2621	002046	064134	DF12	
2622				
2623			:ERROR 46	
2624	002050	060464	EM65	;UNSOLICITED ATTENTION
2625	002052	061740	DH100	
2626	002054	063524	DT100	
2627	002056	064134	DF12	
2628				
2629			:ERROR 47	
2630	002060	060512	EM66	;UNEXPECTED DATA TYPE ERROR
2631	002062	061740	DH100	
2632	002064	063524	DT100	
2633	002066	064134	DF12	
2634				
2635			:ERROR 50	
2636	002070	060545	EM67	;ATTENTION DID NOT RESET WITH CLEAR
2637	002072	061740	DH100	
2638	002074	063524	DT100	
2639	002076	064134	DF12	
2640				
2641			:ERROR 51	
2642	002100	060604	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2643	002102	061740	DH100	
2644	002104	063524	DT100	
2645	002106	064134	DF12	
2646				
2647			:ERROR 52	
2648	002110	060134	EM52	;MULTIPLE DRIVE SELECT
2649	002112	061740	DH100	
2650	002114	063524	DT100	
2651	002116	064134	DF12	
2652				
2653			:ERROR 53	
2654	002120	060162	EM53	;ABREVIATED HCE ERROR
2655	002122	061740	DH100	
2656	002124	063524	DT100	
2657	002126	064164	DF13	
2658				
2659			:ERROR 54	
2660	002130	057462	EM30	;OPERATION INCOMPLETE ERROR
2661	002132	061740	DH100	
2662	002134	063524	DT100	
2663	002136	064214	DF14	
2664				

2665			.ERROR 55	
2666	002140	057507	EM31	;ABREVIATED HVRC ERROR
2667	002142	061740	DH100	
2668	002144	063524	DT100	
2669	002146	064164	DF13	
2670				
2671			.ERROR 56	
2672	002150	060207	EM56	;2 TIMEOUT ERROR
2673	002152	061740	DH100	
2674	002154	063524	DT100	
2675	002156	064244	DF15	
2676				
2677			.ERROR 57	;2ND LEVEL IN SUBSYSTEM TIMEOUT
2678	002160	060207	EM56	
2679	002162	061740	DH100	
2680	002164	063524	DT100	
2681	002166	064310	DF16	
2682				
2683			.ERROR 60	
2684	002170	060226	EM60	;ERROR IN RECAL FOR RECOVERY
2685	002172	000000	0	
2686	002174	000000	0	
2687	002176	000000	0	
2688				
2689			.ERROR 61	
2690	002200	060262	EM61	;ABORT MESSAGE
2691	002202	000000	0	
2692	002204	000000	0	
2693	002206	000000	0	
2694				
2695			.ERROR 62	
2696	002210	060330	EM62	;CYLINDER MISCOMPARE
2697	002212	061740	DH100	
2698	002214	063524	DT100	
2699	002216	064354	DF17	
2700				
2701			.ERROR 63	;DATA ERROR WORDS
2702	002220	000000	0	
2703	002222	000000	0	
2704	002224	063616	DT602	
2705	002226	064464	DF25	
2706				
2707			.ERROR 64	
2708	002230	060354	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2709	002232	061740	DH100	
2710	002234	063524	DT100	
2711	002236	063670	DF02	
2712				
2713			.ERROR 65	
2714	002240	060421	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2715	002242	061740	DH100	
2716	002244	063524	DT100	
2717	002246	063670	DF02	
2718				
2719			.ERROR 66	
2720	002250	060464	EM65	;UNSOLICITED ATTENTION

2721	002252	061740	DH100	
2722	002254	063524	DT100	
2723	002256	063670	DF02	
2724				
2725			:ERROR 67	
2726	002260	060512	EM66	;UNEXPECTED DATA TYPE ERROR
2727	002262	061740	DH100	
2728	002264	063524	DT100	
2729	002266	063670	DF02	
2730				
2731			:ERROR 70	
2732	002270	060545	EM67	;ATTENTION DID NOT RESET WITH CLEAR
2733	002272	061740	DH100	
2734	002274	063524	DT100	
2735	002276	063670	DF02	
2736				
2737			:ERROR 71	
2738	002300	060604	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR ATT
2739	002302	061740	DH100	
2740	002304	063524	DT100	
2741	002306	063670	DF02	
2742				
2743			:ERROR 72	
2744	002310	060653	EM71	;DATA LATE WHEN UNLOADING HEADER
2745	002312	061740	DH100	
2746	002314	063524	DT100	
2747	002316	063670	DF02	
2748				
2749			:ERROR 73	
2750	002320	060713	EM72	;CONTROLLER ERROR DURING DRIVER SERVICE
2751	002322	061740	DH100	
2752	002324	063524	DT100	
2753	002326	063670	DF02	
2754				
2755			:ERROR 74	
2756	002330	060762	EM73	;DRIVE DETECTED PARITY ERROR
2757	002332	061740	DH100	
2758	002334	063524	DT100	
2759	002336	063670	DF02	
2760				
2761			:ERROR 75	
2762	002340	061016	EM74	;UNDEFINED ERROR
2763	002342	061740	DH100	
2764	002344	063524	DT100	
2765	002346	063670	DF02	
2766				
2767			:ERROR 76	
2768	002350	061036	EM75	;MARKING SECTOR BAD MESSAGE
2769	002352	000000	0	
2770	002354	000000	0	
2771	002356	000000	0	
2772				
2773			:ERROR 77	
2774	002360	061066	EM76	;BAD DATA VERIFICATION WITH READ
2775	002362	062725	DH605	
2776	002364	063610	DT601	

2777	002366	064410	DF21	
2778				
2779			:ERROR 100	
2780	002370	061150	EM77	;RETRY SUCCESSFUL MESSAGE
2781	002372	000000	0	
2782	002374	063610	DT601	
2783	002376	064450	DF23	
2784				
2785			:ERROR 101	
2786	002400	061150	EM77	;ANOTHER RETRY SUCCESSFUL MESSAGE
2787	002402	000000	0	
2788	002404	063610	DT601	
2789	002406	064450	DF23	
2790				
2791			:ERROR 102	
2792	002410	061171	EM100	;RETRY UNSUCCESSFUL MESSAGE
2793	002412	000000	0	
2794	002414	063610	DT601	
2795	002416	064450	DF23	
2796				
2797			:ERROR 103	
2798	002420	061214	EM101	;NO VALID HEADERS IN TRACK JUST READ
2799	002422	062707	DH6042	
2800	002424	063610	DT601	
2801	002426	064460	DF24	
2802				
2803			:ERROR 104	
2804	002430	061272	EM102	;BSE ERROR ON SECTOR NOT LISTED AS BAD
2805	002432	061740	DH100	
2806	002434	063524	DT100	
2807	002436	063670	DF02	
2808				
2809			:ERROR 105	
2810	002440	061344	EM103	;TIMED-OUT ON READ HEADER
2811	002442	061740	DH100	
2812	002444	063524	DT100	
2813	002446	063724	DF03	
2814				
2815			:ERROR 106	
2816	002450	061372	EM104	;TIMED-OUT ON SEEK
2817	002452	061740	DH100	
2818	002454	063524	DT100	
2819	002456	063724	DF03	
2820				
2821			:ERROR 107	
2822	002460	061414	EM105	;DRIVE SIEZED BY OTHER PORT
2823	002462	061740	DH100	
2824	002464	063524	DT100	
2825	002466	063670	DF02	
2826				
2827			:ERROR 110	
2828	002470	061447	EM106	; "DATA MISCMPR WHILE BAI SET"
2829	002472	061740	DH100	
2830	002474	063524	DT100	
2831	002476	064500	DF27	
2832				

E07

2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002

:RKDB (DATA BUFFER)
15 14 13 12 11 10 9 8

DB15 ! DB14 ! DB13 ! DB12 ! DB11 ! DB10 ! DB9 ! DB8 !
READ/WRITE

7 6 5 4 3 2 1 0

DB7 ! DB6 ! DB5 ! DB4 ! DB3 ! DB2 ! DB1 ! DB0 !
READ/WRITE

:RKASOF (ATTENTION SUMMARY AND OFFSET)
15 14 13 12 11 10 9 8

ATN7 ! ATN6 ! ATN5 ! ATN4 ! ATN3 ! ATN2 ! ATN1 ! ATN0 !
READ ONLY

7 6 5 4 3 2 1 0

UN ? ! OF6 ! OF5 ! OF4 ! OF3 ! OF2 ! OF1 ! OF0 !
READ/WRITE

:RKWC (WORD COUNT)
15 14 13 12 11 10 9 8

WC15 ! WC14 ! WC13 ! WC12 ! WC11 ! WC10 ! WC9 ! WC8 !
READ/WRITE

7 6 5 4 3 2 1 0

WC7 ! WC6 ! WC5 ! WC4 ! WC3 ! WC2 ! WC1 ! WC0 !
READ/WRITE

:RKBA (BUS ADDRESS)
15 14 13 12 11 10 9 8

BA15 ! BA14 ! BA13 ! BA12 ! BA11 ! BA10 ! BA9 ! BA8 !
READ/WRITE

7 6 5 4 3 2 1 0

BA7 ! BA6 ! BA5 ! BA4 ! BA3 ! BA2 ! BA1 ! BA0 !
READ/WRITE !ALWYSC!

F07

D2R6N8.B - Rk611
D2R6N8.CMB

RK06 SUBSYSTEM VERIFICATION : PART 2
BIT ASSIGNMENTS IN THE RK611 REGISTERS

MACY11 27(732) 03-NOV-76 21:43 PAGE 84

3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041

:RKR (ERROR REGISTER)							
15	14	13	12	11	10	9	8
DCK	! UNS	! OPI	! DTE	! WLE	! IDAE	! COE	! M'RC
: READ ONLY							
7	6	5	4	3	2	1	0
BSE	! ECH	! DTYPE	! FMTE	! DRPAR	! ILF	! SKI	! ILC
: READ ONLY							
:RKDS (DRIVE STATUS)							
15	14	13	12	11	10	9	8
SVAL	! DSC	! PIP	! SPON	! WRL	! UN	! UN	! DTP
: READ ONLY							
7	6	5	4	3	2	1	0
DRDY	! VV	! DROT	! SPLS	! ACLO	! OFFSET	! UN	! DRA
: READ ONLY							
:RKMRI (MAINTENANCE REG 1)							
15	14	13	12	11	10	9	8
RD	! WRT	! ECCW	! PCD	! PCA	! MEWD	! MERD	! MCLK
GATE	! GATE	: READ ONLY			: READ/WRITE		
7	6	5	4	3	2	1	0
MIND	! MSP	! DMD	! PAT	! MS3	! MS2	! MS1	! MS0
: READ/WRITE							

3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079

```
;RKECC/PAT
: 15 14 13 12 11 10 9 8
-----
: UN ! UN ! UN ! UN ! UN ! EPA10! EPA9 ! EPA8 !
: READ ONLY
-----
```

```
: 7 6 5 4 3 2 1 0
-----
: EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
: READ ONLY
-----
```

```
;RKECC/POS
: 15 14 13 12 11 10 9 8
-----
: UN ! UN ! UN ! EP012! EP011! EP010! EP09 ! EP08 !
: READ ONLY
-----
```

```
: 7 6 5 4 3 2 1 0
-----
: EP07 ! EP06 ! EP05 ! EP04 ! EP03 ! EP02 ! EP01 ! EP00 !
: READ ONLY
-----
```

;DRIVE STATUS INFORMATION

```
;LINE A MESSAGE 00
: 15 14 13 12 11 10 9 8
-----
: PAR ! DSC ! PIP ! SPON ! WRLK ! OFFON! FMT ! DRTP !
-----
```

```
: 7 6 5 4 3 2 1 0
-----
: DRDY ! VV ! DRA ! NU ! NU ! DRIVE SELECT CODE !
-----
```

H07

3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117

:LINE B MESSAGE 00							
15	14	13	12	11	10	9	8
PAR	UNS	DROT	DCLO	WLE	SKI	PERR	ILF
7	6	5	4	3	2	1	0
FLT	ACLO	IVDA	UN	UN	UN	0	0
:LINE A MESSAGE 01							
15	14	13	12	11	10	9	8
PAR	UNLDG	RTZ	LDG	REV	FWD	SPEED	CART
	HCS		HDS			OK	PRES
7	6	5	4	3	2	1	0
DOOR	BRUSH	HEADS	TRK	UN	DRIVE SELECT CODE		
LTCH	HOME	HOME	FOLOW				
			OK				
:LINE B MESSAGE 01							
15	14	13	12	11	10	9	8
PAR	SERVO	LIMIT	SEEK	PLO	DIBIT	INDEX	MULTI
	BRAKE	ON SK	NO MO	ERR	ERR	ERR	HD SEL
7	6	5	4	3	2	1	0
HEAD	WR GTE	WR CUR	SEC	NU	NU	0	1
FAULT	AND NO	AND NO	ERR				
	XISTON	WR GTE					

```

3118 ;LINE A MESSAGE 10
3119 ; 15 14 13 12 11 10 9 8
3120 -----
3121 ; PAR ! NU ! NU ! CYLINDER DIFF/OFFSET VALUE !
3122 -----
3123 ; 7 6 5 4 3 2 1 0
3124 -----
3125 ; CYLINDER DIFF/OFFSET VALUE! NU ! DRIVE SELECT CODE !
3126 -----
3127
3128 ;LINE B MESSAGE 10
3129 ; 15 14 13 12 11 10 9 8
3130 -----
3131 ; PAR ! ALIGN! NU ! CYLINDER ADDRESS !
3132 ; SIGN !
3133 -----
3134 ; 7 6 5 4 3 2 1 0
3135 -----
3136 ; CYLINDER ADDRESS ! UN ! UN ! 1 ! 0 !
3137 -----
3138
3139 ;LINE A MESSAGE 11
3140 ; 15 14 13 12 11 10 9 8
3141 -----
3142 ; PAR ! DRIVE SERIAL NUMBER !
3143 -----
3144 ; 7 6 5 4 3 2 1 0
3145 -----
3146 ; DRIVE SERIAL NUMBER ! DRIVE SELECT CODE !
3147 -----
3148
3149 ;LINE B MESSAGE 11
3150 ; 15 14 13 12 11 10 9 8
3151 -----
3152 ; PAR ! NU ! NU ! DECODED HEAD ! SECTOR!
3153 ; ADDRESS ! COUNT !
3154 -----
3155 ; 7 6 5 4 3 2 1 0
3156 -----
3157 ; SECTOR COUNT ! UN ! UN ! 1 ! 1 !
3158 -----
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
  
```

.SBTTL RK06 CONTROLLER REGISTER DEFINITION

000000
 000002
 000004
 000006
 000010

RKCS1= 0
 RKWC= 2
 RKBA= 4
 RKDA= 6
 RKCS2= 10

;CONTROL AND STATUS REGISTER 1
 ;WORD COUNT REGISTER
 ;BUS ADDRESS REGISTER
 ;DESIRED TRACK SECTOR REGISTER
 ;CONTROL AND STATUS REGISTER 2


```

3174      000012      RKDS=   12      ;DRIVE STATUS REGISTER
3175      000014      RKER=   14      ;ERROR REGISTER
3176      000016      RKASOF=  16      ;ATTENTION SUMMARY AND OFFSET REGISTER
3177      000020      RKDC=   20      ;DESIRED CYLINDER REGISTER
3178      000020      RKDCYL= 20      ;DESIRED CYLINDER REGISTER
3179      000024      RKDB=   24      ;DATA BUFFER
3180      000026      RKMR1=  26      ;MAINTENANCE REGISTER 1
3181      000034      RKMR2=  34      ;MAINTENANCE REGISTER 2
3182      000036      RKMR3=  36      ;MAINTENANCE REGISTER 3
3183      000030      RKPOS=  30      ;ECC POSITION INFORMATION
3184      000030      RKECPS= 30      ;ECC POSITION INFORMATION
3185      000032      RKPAT=  32      ;ECC PATTERN INFORMATION
3186      000032      RKECPT= 32      ;ECC PATTERN INFORMATION
3187
3188      .SBTTL  DRIVE COMMANDS
3189
3190      000101      SELDRV= 101     ;SELECT DRIVE
3191      000103      PACK=   103    ;PACK ACKNOWLEDGE
3192      000105      CLEAR=  105    ;DRIVE CLEAR
3193      000107      UNLOAD= 107    ;UNLOAD
3194      000111      SRTSPL= 111    ;START SPINDLE
3195      000113      RECAL=  113    ;RECALIBRATE
3196      000115      OFFSET= 115    ;OFFSET
3197      000117      SEEK=   117    ;SEEK
3198      000121      RDDATA= 121    ;READ DATA
3199      000123      WRDATA= 123    ;WRITE DATA
3200      000125      RDHEAD= 125    ;READ HEADER
3201      000127      WRHEAD= 127    ;WRITE HEADER AND DATA
3202      000131      WRTCHK= 131    ;WRITE CHECK
3203
3204      ;          THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
3205      ;          TO SIMULATE A SPECIFIC DESIRED OPERATION
3206
3207      000140      RELEAS= 140    ;RELEASE DRIVE
3208      000141      RDSTAT= 141    ;GET ALL STATUS FROM DRIVE
3209      000164      RDALHD= 164    ;READ ALL HEADERS
3210      000176      CONCLR= 176    ;CONTROLLER CLEAR (BIT 15 OF CS1)
3211      000177      SUBCLR= 177    ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
3212      000300      INTR=   300    ;GENERATE INTERRUPT TO CPU
3213
3214      ;          DRIVER ISSUED SERVICE COMMANDS
3215
3216      000001      DR.SEL= 001    ;DRIVE SELECT
3217      000005      DR.CLR= 005    ;DRIVE CLEAR
3218
3219      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
3220
3221      000001      GO=     BIT0    ;GO BIT
3222      000100      IE=     BIT6    ;INTERRUPT ENABLE
3223      000200      RDY=    BIT7    ;CONTROLLER READY
3224      000400      BA16=   BIT8    ;BUS ADDRESS BIT 16
3225      001000      BA17=   BIT9    ;BUS ADDRESS BIT 17
3226      002000      CDT=    BIT10   ;CONTROLLER DRIVE TYPE (0=RK06)
3227      004000      CTO=    BIT11   ;CONTROLLER TIMED OUT WAITING FOR
3228      ;          DRIVE RESPONSE
3229      010000      CFMT=   BIT12   ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

```

```

3230      020000      SPAR=   BIT13      ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
3231      040000      DI=     BIT14      ;DRIVE INTERRUPT
3232      100000      CERR=   BIT15      ;CONTROLLER ERROR
3233      100000      CCLR=   BIT15      ;CONTROLLER CLEAR
3234
3235      :           ;           THESE BIT DEFINITIONS ARE USED FOR ADDRESS
3236      :           ;           THE HIGH BYTE OF RKCS1
3237
3238      000001      B.BA16=  BIT0      ;BUS ADDRESS BIT 16
3239      000002      B.BA17=  BIT1      ;BUS ADDRESS BIT 17
3240      000004      B.CDT=   BIT2      ;CONTROLLER DRIVE TYPE (0=RK06)
3241      000020      B.CFMT=  BIT4      ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
3242
3243      .SBTTL CONTROL AND STATUS REGISTER 2 BITS
3244
3245      000007      DRVMSK= 7           ;MASK FOR DRIVE SELECTION CODE
3246      000010      DESL=   BIT3      ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3247      000010      RLS=   BIT3      ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3248      000020      BAI=   BIT4      ;BUS ADDRESS INCREMENT INHIBIT
3249      000040      CLR=   BITS      ;CLEAR CONTROLLER AND ALL DRIVES
3250      000040      SCLR=  BITS      ;CLEAR CONTROLLER AND ALL DRIVES
3251      000100      IR=    BIT6      ;INPUT READY
3252      000200      OR=    BIT7      ;OUTPUT READY
3253      000400      UFE=   BIT8      ;UNIT FIELD ERROR
3254      001000      MDS=   BIT9      ;MULTIPLE DRIVE SELECT
3255      002000      PGE=   BIT10     ;PROGRAMMING ERROR
3256      004000      NEM=   BIT11     ;NON-EXISTENT MEMORY
3257      010000      NED=   BIT12     ;NON-EXISTENT DRIVE
3258      020000      UPE=   BIT13     ;UNIBUS PARITY ERROR
3259      040000      WCE=   BIT14     ;WRITE CHECK ERROR
3260      100000      DLT=   BIT15     ;DATA LATE ERROR
3261
3262      .SBTTL ERROR REGISTER BIT DEFINITION
3263
3264      000001      ILC=    BIT0      ;ILLEGAL FUNCTION CODE
3265      :*ILF=    BIT0      ;ILLEGAL FUNCTION CODE
3266      000002      SKI=    BIT1      ;SEEK INCOMPLETE
3267      000004      ILF=    BIT2      ;ILLEGAL DRIVE FUNCTION
3268      000004      NXF=    BIT2      ;ILLEGAL DRIVE FUNCTION
3269      000010      DRPAR=  BIT3      ;DRIVE DETECTED DRIVE BUS PARITY ERROR
3270      000020      FMTE=   ;FORMAT ERROR
3271      000040      DTYE=   B. 3      ;DRIVE TYPE ERROR
3272      000100      ECH=    BIT6      ;ECC HARD
3273      000200      BSE=    BIT7      ;BAD SECTOR ERROR
3274      000400      HCRC=   BIT8      ;HEADER CRC ERROR
3275      000400      HVRC=   BIT8      ;HEADER VRC ERROR
3276      001000      COE=    BIT9      ;CYLINDER ADDRESS OVERFLOW ERROR
3277      002000      IDAE=   BIT10     ;INVALID DISK ADDRESS ERROR
3278      004000      WLE=    BIT11     ;WRITE LOCK ERROR
3279      010000      DTE=    BIT12     ;DRIVE TIMING ERROR
3280      020000      OPI=    BIT13     ;OPERATION (SEARCH) INCOMPLETE
3281      040000      UNS=    BIT14     ;DRIVE UNSAFE
3282      100000      DCK=    BIT15     ;DATA CHECK
3283
3284      .SBTTL STATUS REGISTER BIT DEFINITION
3285

```

```

3286      000001      DRA=      BIT0      ;DRIVE AVAILABLE (CONTROLLER IS SET IF
3287      ;          THIS BIT IS RESET)
3288      000004      OFST=      BIT2      ;DRIVE OFFSET
3289      000010      ACLO=      BIT3      ;AC LOW
3290      000020      SPDLSS=     BIT4      ;SPEED LOSS
3291      000020      DCLO=      BIT4      ;DC LOW
3292      000040      DROT=      BIT5      ;DRIVE OFF TRACK
3293      000100      VV=        BIT6      ;VOLUME VALID
3294      000200      DRY=        BIT7      ;DRIVE READY
3295      000200      DRDY=      BIT7      ;DRIVE READY
3296      000400      DDT=        BIT8      ;DRIVE TYPE (0=RK06)
3297      004000      WRL=        BIT11     ;WRITE LOCK
3298      020000      PIP=        BIT13     ;POSITIONING IN PROGRESS
3299      040000      DSC=        BIT14     ;DRIVE STATUS CHANGE
3300      100000      SVAL=      BIT15     ;STATUS VALID
3301
3302      .SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION
3303
3304      000017      MESMSK= 17      ;MESSAGE MASK
3305
3306      000020      PAT=        BIT4      ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
3307      000040      DMD=        BITS      ;DIAGNOSTIC MODE
3308      000100      MSP=        BIT6      ;MAINTENANCE SECTOR PULSE
3309      000200      MIND=      BIT7      ;MAINTENANCE INDEX
3310      000400      MCLK=      BIT8      ;MAINTENANCE CLOCK
3311      001000      MERD=      BIT9      ;MAINTENANCE ENCODED READ DATA
3312      002000      MEWD=      BIT10     ;MAINTENANCE ENCODED WRITE DATA
3313      004000      PCA=        BIT11     ;PRECOMPENSATION ADVANCE
3314      010000      PCD=        BIT12     ;PRECOMPENSATION DELAY
3315      020000      ECCW=      BIT13     ;ECC WORD IS BEING READ OR WRITTEN
3316      040000      WRGAT=     BIT14     ;WRITE GATE
3317      100000      RDGATE=    BIT15     ;READ GATE
3318
3319      .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
3320
3321      000040      S.DRA=     BITS      ;DRIVE AVAILIABLE
3322      000100      S.VV=      BIT6      ;VOLUME VALID
3323      000200      S.DRY=     BIT7      ;DRIVE READY
3324      000400      S.TYPE=    BIT8      ;DRIVE TYPE
3325      001000      S.FORM=    BIT9      ;DRIVE FORMAT
3326      002000      S.OFF=     BIT10     ;OFFSET
3327      004000      S.WRL=     BIT11     ;WRITE LOCK
3328      010000      S.SPIN=    BIT12     ;SPINDLE ON
3329      020000      S.PIP=     BIT13     ;POSITIONING IN PROGRESS
3330      040000      S.DSC=     BIT14     ;DRIVE STATUS CHANGE
3331
3332      .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
3333
3334      000040      S.ICYL=    BITS      ;ILLEGAL CYLINDER ADDRESS
3335      000100      S.ACLO=    BIT6      ;AC LOW
3336      000200      S.FLT=     BIT7      ;DRIVE FAULT
3337      000400      S.ILF=     BIT8      ;ILLEGAL FUNCTION
3338      001000      S.PAR=     BIT9      ;DRIVE DETECTED DRIVE BUS PARITY ERROR
3339      002000      S.SKI=     BIT10     ;SEEK INCOMPLETE
3340      004000      S.WLE=     BIT11     ;WRITE LOCK ERROR
3341      010000      S.SPLS=    BIT12     ;SPEED LOSS

```

```

3342      010000      S.DCLO= BIT12      ;DC LOW
3343      020000      S.DROT= BIT13      ;DRIVE OFF TRACK
3344      040000      S.UNS= BIT14      ;DRIVE UNSAFE
3345
3346      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3347
3348      000020      S.XDOK= BIT4      ;TRANSDUCER OK
3349      000040      S.HDHM= BIT5      ;HEADS HOME
3350      000100      S.BRHM= BIT6      ;BRUSHES HOME
3351      000200      S.DOOR= BIT7      ;DOOR INTERLOCKED
3352      000400      S.CART= BIT8      ;CARTRAGE INTERLOCK
3353      001000      S.SPOK= BIT9      ;SPEED OK
3354      002000      S.FWD= BIT10     ;FORWARD
3355      004000      S.REV= BIT11     ;REVERSE
3356      010000      S.LOAD= BIT12     ;HEADS LOADING
3357      020000      S.RTZ= BIT13     ;RETURN TO ZERO
3358      040000      S.UNLD= BIT14     ;HEADS UNLOADING
3359
3360      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3361
3362      000020      S.SECT= BIT4      ;SECTOR ERROR
3363      000040      S.WCLK= BIT5      ;WRITE CLOCK AND NO WRITE GATE
3364      000100      S.WGAT= BIT6      ;WRITE GATE AND NO TRANSISTIONS
3365      000200      S.HDFL= BIT7      ;HEAD FAULT
3366      000400      S.MHD= BIT8      ;MULTIPLE HEAD SELECT
3367      001000      S.XERR= BIT9      ;INDEX ERROR
3368      002000      S.DIB= BIT10     ;DIBIT ERROR
3369      004000      S.PLO= BIT11     ;PLO ERROR
3370      010000      S.NMOV= BIT12     ;SEEK AND NO MOTION
3371      020000      S.LIMD= BIT13     ;LIMIT DETECT ON SEEK
3372      040000      S.BRAKE= BIT14    ;SERVO-BRAKE
3373
3374      .SBTTL  COMMON MASKS
3375
3376      000007      M.DRV= 7      ;DRIVE CODE
3377      100000      M.PAR= BIT15     ;PARITY
3378      000003      M.ID= 3      ;BYTE ID
3379      017760      M.CDIF= 17760   ;CYLINDER DIFFERENCE/OFFSET
3380      017760      M.CADD= 17760   ;CYLINDER ADDRESS
3381      077770      M.SER= 77770    ;DRIVE SERIAL NUMBER
3382      000760      M.SECT= 760     ;SECTOR COUNT
3383      007000      M.HEAD= 7000    ;HEAD DECODE

```

3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439

.SBTTL PHARAMETER BLOCK ALLOCATION

```

*****
: * 1 : COMMAND : DRIVE NO.
: * 3 : CYLINDER ADDRESS
: * 5 : TRACK : SECTOR
: * 7 : BA16-17 FORMAT, DRV TYPE: OFFSET
: * 11 : BUS ADDRESS (LOW 16 BITS)
: * 13 : WORD COUNT (2'S COMPLEMENT)
: * 15 : PROGRAM DRIVE STATUS INFORMATION
: * 17 : COMMAND AND STATUS REGISTER 1
: * 21 : COMMAND AND STATUS REGISTER 2
: * 23 : WORD COUNT REGISTER
: * 25 : BUS ADDRESS REGISTER
: * 27 : DESIRED TRACK AND SECTOR
: * 31 : DESIRED CYLINDER
: * 33 : ATTENTION SUMMARY AND DRIVE OFFSET
: * 35 : ERROR REGISTER
: * 37 : STATUS REGISTER
: * 41 : MESSAGE LINE A STATUS BYTE 00
: * 43 : MESSAGE LINE B STATUS BYTE 00
: * 45 : MESSAGE LINE A STATUS BYTE 01
: * 47 : MESSAGE LINE B STATUS BYTE 01
: * 51 : MESSAGE LINE A STATUS BYTE 10
: * 53 : MESSAGE LINE B STATUS BYTE 10
: * 55 : MESSAGE LINE A STATUS BYTE 11
: * 57 : MESSAGE LINE B STATUS BYTE 11
: * 61 : ECC POSITION INFORMATION
: * 63 : ECC PATTERN INFORMATION
*****

```

0
2
4
6
10
12
14
16
20
22
24
26
30
32
34
36
40
42
44
46
50
52
54
56
60
62

.SBTTL PARAMETERS PASSED TO THE DRIVER

THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS TO THE RK06 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

```

```

000100          CMD0=  BIT6          ;NO TERMINATION TO COMMAND FOR AT
                                ;LEAST 1 SECOND
000200          W.WCK=  BIT7          ;WRITE FOR WRITE WRITE CHECK
000400          WCHK=  BIT8          ;NO CHECK, DO NOT SET INTERRUPT ENABLE
001000          PBSVAL= BIT9          ;PARAMETER STATUS WCPDS VALID
                                ;(SET WHEN ERROR TERMINATION OR
                                ;READ STATUS COMMAND)
002000          DRPCRV= BIT10         ;DROP DRIVE FROM TEST SEQUENCE
004000          WOSCO=  BIT11         ;ATTENTION SET BUT DCS AND FAULT RESET
000000          CRVSDZ= BIT12         ;DRIVE SEIZED BY OTHER PORT
000000          F.UNLD= BIT13         ;DRIVE UNLOADED DUE TO ERROR
000000          Q.INIT= BIT14         ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
100000          DTBAII= BIT15         ;INHIBIT BUS ADDRESS INCREMENT
    
```

.SBTTL PARAMETERS PASSED FROM DRIVER TO PROGRAM

THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS FROM THE DRIVER TO THE CALLING PROGRAM

```

000016          P.CS1=  16          ;COMMAND AND STATUS REGISTER 1
000020          P.CS2=  20          ;COMMAND AND STATUS REGISTER 2
000022          P.WCR=  22          ;WORD COUNT REGISTER
000024          P.BAR=  24          ;BUS ADDRESS REGISTER
000026          P.DTS=  26          ;DESIRED TRACK SECTOR REGISTER
000030          P.DCYL= 30          ;DESIRED CYLINDER REGISTER
000032          P.ASOFF= 32         ;ATTENTION SUMMARY/OFFSET REGISTER
000034          P.ER=   34          ;ERROR REGISTER
000036          P.DC=   36          ;STATUS REGISTER
000040          P.A00=  40          ;MESSAGE A STATUS BYTE 00
000042          P.B00=  42          ;MESSAGE B STATUS BYTE 00
000044          P.A01=  44          ;MESSAGE A STATUS BYTE 01
000046          P.B01=  46          ;MESSAGE B STATUS BYTE 01
000050          P.A10=  50          ;MESSAGE A STATUS BYTE 10
000052          P.B10=  52          ;MESSAGE B STATUS BYTE 10
000054          P.A11=  54          ;MESSAGE A STATUS BYTE 11
000056          P.B11=  56          ;MESSAGE B STATUS BYTE 11
000060          P.EPOS=  60          ;ECC POSITION INFORMATION
000062          P.EPAT=  62          ;ECC PATTERN INFORMATION
    
```

.SBTTL PARAMETER BLOCK 0 FOR DRIVE

```

002620          PARM0: .BYTE 0          ;DRIVE NUMBER
002621          .BYTE 00          ;COMMAND
002622          .WORD 000000        ;CYLINDER ADDRESS
002624          .BYTE 00          ;SECTOR ADDRESS
002625          .BYTE 00          ;TRACK ADDRESS
002626          .BYTE 00          ;OFFSET VALUE
002627          .BYTE 00          ;BUS ADDRESS (BITS 16 AND 17)
002630          .WORD 000000        ;BUS ADDRESS (BITS 0 - 15)
002632          .WORD 000000        ;WORD COUNT (2'S COMPLEMENT)
002634          .WORD 000000        ;PROGRAM DRIVE STATUS INFORMATION
002636          .WORD 000000        ;COMMAND AND STATUS REGISTER 1
002640          .WORD 000000        ;COMMAND AND STATUS REGISTER 2
002642          .WORD 000000        ;WORD COUNT REGISTER
002644          .WORD 000000        ;BUS ADDRESS REGISTER
002646          .WORD 0          ;DESIRED TRACK AND SECTOR REGISTER
    
```

```

3496 002650 000000
3497 002652 000000
3498 002654 000000
3499 002656 000000
3500 002660 000000
3501 002662 000000
3502 002664 000000
3503 002666 000000
3504 002670 000000
3505 002672 000000
3506 002674 000000
3507 002676 000000
3508 002700 000000
3509 002702 000000
3510
3511
3512
3513 002704 000
3514 002705 000
3515 002706 000000
3516 002710 000
3517 002711 000
3518 002712 000
3519 002713 000
3520 002714 000000
3521 002716 000000
3522 002720 000000
3523 002722 000000
3524 002724 000000
3525 002726 000000
3526 002730 000000
3527 002732 000000
3528 002734 000000
3529 002736 000000
3530 002740 000000
3531 002742 000000
3532 002744 000000
3533 002746 000000
3534 002750 000000
3535 002752 000000
3536 002754 000000
3537 002756 000000
3538 002760 000000
3539 002762 000000
3540 002764 000000
3541 002766 000000
3542
3543
3544
3545 002770 000000
3546
3547 002772 000000
3548
3549 002774 000000
3550 002776 000000
3551 003000 000000

```

```

        .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

.SBTL PARAMETER BLOCK 1 FOR DRIVE
PARAM1: .BYTE 0 ; DRIVE NUMBER
        .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

.SBTL TEMPORARY CONTROLLER REGISTER STORAGE
T.CS1: .WORD 0 ; TEMPORARY STORAGE FOR COMMAND AND STATUS
        ; REGISTER 1
T.CS2: .WORD 0 ; TEMPORARY STORAGE FOR COMMAND AND STATUS
        ; REGISTER 2
T.WCR: .WORD 0 ; TEMPORARY STORAGE FOR WORD COUNT REGISTER
T.BA: .WORD 0 ; TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
T.DA: .WORD 0 ; TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

```

3552	003002	000000	T.DC: .WORD	0	: TEMPORARY STORAGE FOR DRIVE CYLINDER
3553	003004	000000	T.ASOF: .WORD	0	: TEMPORARY STORAGE FOR ATTENTION SUMMARY AND OFFSET
3554					
3555	003006	000000	T.ER: .WORD	0	: TEMPORARY STORAGE FOR ERROR REGISTER
3556	003010	000000	T.DS: .WORD	0	: TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3557	003012	000000	T.MR1: .WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3558	003014	000000	T.MR2: .WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3559	003016	000000	T.MR3: .WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3560	003020	000000	T.POS: .WORD	0	: TEMPORARY STORAGE FOR ECC POSITION
3561	003022	000000	T.PAT: .WORD	0	: TEMPORARY STORAGE FOR ECC PATTERN
3562	003024	000000	T.DB: .WORD	0	: TEMPORARY STORAGE FOR DATA BUFFER REGISTER

.SBTTL DRIVER PARAMENTERS

3563					
3564					
3565					
3566	003026	177440	RKBAS: .WORD	177440	: ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3567	003030	000210	RKVEC: .WORD	210	: ADDRESS OF R611 VECTOR
3568	003032	000240	RKPRI: .WORD	PR5	: RK611 INTERRUPT PRIORITY
3569	003034	041142	A.NORM: ERRFRE		: ADDRESS OF NORMAL RETURN FROM DRIVER
3570	003036	042410	A.ABNL: ERRHDL		: ADDRESS OF ABNORMAL RETURN FROM DRIVER
3571	003040	041704	A.CONT: CONERR		: ADDRESS OF CONTROLLER ERROR RETURN
3572	003042	000000	E.CONT: .WORD	0	: CONTROLLER ERROR STATUS THIS LOCATION IS CLEARED WHEN EVERY COMMAND IS INITIATED. IF A CONTROLLER ERROR OCCURS THE FOLLOWING BIT ASSIGNMENT IS USED:
3573					
3574					
3575					
3576					
3577					

3578		000001	E.CCLR= BIT0		: CLEAR CONTROLLER DID NOT CLEAR ERROR
3579		000002	E.NOAT= BIT1		: NO ATTENTION IN ATTENTION SUMMARY REG
3580		000004	E.UATT= BIT2		: UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3581		000010	E.UDAT= BIT3		: UNEXPECTED DATA TYPE ERROR
3582		000020	E.CLAT= BIT4		: ATTENTION DID NOT RESET WITH CLEAR
3583		000040	E.SCLR= BITS		: SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
3584					
3585		000100	E.ILLD= BIT6		: ILLEGAL DRIVER COMMAND
3586		000400	E.DLT= BIT8		: DATA LATE WHEN UNLOADING HEADER
3587		001000	E.CERR= BIT9		: CONTROLLER ERROR DURING DRIVER SERVICING
3588		002000	E.DPAR= BIT10		: DRIVE DETECTED PARITY ERROR
3589		040000	E.CMTO= BIT14		: CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3590		100000	E.MDS= BIT15		: MULTIPLE DRIVE SELECT

3591					
3592	003044	000000	G.WAIT: .WORD	0	: PARAMETER BLOCK OF THE DRIVE
3593					: WAITING FOR COMMAND COMPLETION
3594	003046	000400	W.MTIM: .WORD	400	: LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
3595	003050	000400	W.MILI: .WORD	400	: 16 MILLISECOND TIME FOR PROGRAM

	CPU	VALUE
	---	-----
3596		
3597		
3598		
3599		
3600	11/05	100
3601	11/10	
3602	11/20	
3603	11/34	
3604	11/40	
3605	11/45	400
3606	11/50	
3607	11/70	

3608	003052	000100			W.SEC: .WORD	100		: SECOND COUNT COUNT FOR ALL COMMANDS
3609								: EXCEPT START SPINDLE
3610	003054	001000			W.8SEC: .WORD	1000		: 8 SECOND FOR DRIVE CYCLE DOWN
3611	003056	010000			W.MIN: .WORD	10000		: MINUTE TIME FOR START SPINDLE
3612	003060	000000			HDR.AD: .WORD	0		: ADDRESS USED FOR READ ALL HEADERS
3613	003062	000000			HDR.CT: .WORD	0		: NUMBER OF HEADERS LEFT TO READ FOR READ
3614								: ALL HEADERS
3615	003064	000			I.ISRL: .BYTE	0		: INTERRUPT OR RELEASED COMMAND ISSUED
3616	003065	002	004	010	H.HEAD: .BYTE	2,4,10		: HEAD DECODES
3617	003070	000			W.TIME: .BYTE	0		: DRIVES BEING WATCH-DOG TIMED
3618								
3619					.SBTTL	INTERRUPT MASKS		
3620								
3621	003071	000			INTMSK: .BYTE	0		: INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
3622								
3623								: INTERRUPT MASK TABLE
3624								
3625	003072	001			I.DRV: .BYTE	1		: INTERRUPT MASK FOR DRIVE 0
3626	003073	002			.BYTE	2		: INTERRUPT MASK FOR DRIVE 1
3627	003074	004			.BYTE	4		: INTERRUPT MASK FOR DRIVE 2
3628	003075	010			.BYTE	10		: INTERRUPT MASK FOR DRIVE 3
3629	003076	020			.BYTE	20		: INTERRUPT MASK FOR DRIVE 4
3630	003077	040			.BYTE	40		: INTERRUPT MASK FOR DRIVE 5
3631	003100	100			.BYTE	100		: INTERRUPT MASK FOR DRIVE 6
3632	003101	200			.BYTE	200		: INTERRUPT MASK FOR DRIVE 7
3633								
3634					.SBTTL	PARAMETER BLOCK TABLE		
3635								
3636	003102	002620			PBLKT: PARM0			: ADDRESS OF PARAMETER BLOCK GIVEN WITH
3637								: DRIVE CALL. MUST BE LOADED INTO PBLKT
3638								
3639								
3640					.SBTTL	TIME FOR WATCH-DOG TIMER		
3641								
3642	003104	000000			W.DRV: .WORD	0		: TIME FOR INSTRUCTION IN PARAMETER BLOCK
3643					.SBTTL	PROGRAM SPECIFIC RESERVED LOCATIONS		
3644	003106	000			MDFLAG: .BYTE	0		: FLAG TO INDIC. DEFLT OR PARAM MODE
3645	003107	000			XXDPCH: .BYTE	0		: XXDP CHAIN MODE FLAG
3646	003110	000			TSTING: .BYTE	0		: CURRENTLY RUNNING TESTS IF = 1
3647	003111	000			DERCNT: .BYTE	0		: DATA ERROR COUNT
3648	003112	000			OPCOMP: .BYTE	0		: OPERATION COMPLETE FLAG
3649	003113	000			DONE: .BYTE	0		: DONE SWITCH
3650	003114	000			TYPFMT: .BYTE	0		: DRIVE TYPE & FORMAT CONTROL
3651	003115	000			FORMAT: .BYTE	0		: DRIVE FORMAT IN BIT 4 OF BYTE
3652	003116	000			ERRCNT: .BYTE	0		: ERROR COUNT
3653	003117	004			ERRLMT: .BYTE	4		: ERROR LIMIT
3654	003120	000			DRVERS: .BYTE	0		: ERROR COUNT FOR CURRENT DRIVE
3655	003121	000			OPCONT: .BYTE	0		: OPERATION CONTROL SWITCHES
3656	003122	000			PCLKF: .BYTE	0		: IF BYTE=1, KW11-P CLOCK IS PRESENT
3657	003123	000			DOTIM: .BYTE	0		: IF BYTE=1, DO TIMING TESTS
3658	003124	000			XOVLAD: .BYTE	0		: FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
3659	003125	000			XDPSVD: .BYTE	0		: FLAG = 1 IF XXDP IS SAVED
3660	003126	000			DULACS: .BYTE	0		: FLAG=1 IF DUAL ACCESS TEST
3661	003127	000			DRNAFG: .BYTE	0		: =1 INDICATES DRIVE SIEZED BY OTHER PORT
3662	003130	000			REISSU: .BYTE	0		: DUAL-ACC FLAG TO RE-ISSUE COMMAND
3663	003131	000			WCEFLG: .BYTE	0		: WRITE CHECK ERROR FLAG

3664	003132	000	DLTFLG: .BYTE	0	: DATA LATE ERROR FLAG
3665	003133	000	NCRTRY: .BYTE	0	: "NO-RETRY" FLAG
3666	003134	000	UBMPRS: .BYTE	0	: UNIBUS MAP PRESENT IF = 1
3667	003135	000	MEMABT: .BYTE	0	: SET BYTE = 1 FOR NO ABORT
3668					: ON MEMORY PARITY ERRORS
3669	003136	000	HLPOVL: .BYTE	0	: HELP FILE OVERLAID INDICATOR
3670	003137	000	NOTYPE: .BYTE	0	: INDICATES PROGRAM JUST LOADED IF 0
3671		000001	WHDSW=BIT0		: WRITE HEADER & DATA SWITCH
3672		000002	VHDSW=BIT1		: VERIFY HEADERS SWITCH
3673		000004	WCDASW=BIT2		: WRITE CHECK DATA SWITCH
3674		000010	RCDASW=BIT3		: READ CHECK DATA SWITCH
3675		000020	OREGSW=BIT4		: OFFSET REQUIRED SWITCH
3676			.EVEN		
3677					
3678	003140	177546	LKS: .WORD	177546	: KW11-L CLOCK STATUS REGISTER
3679	003142	172540	PKS: .WORD	172540	: KW11-P CONTROL AND STATUS REGISTER
3680	003144	172542	PKSB: .WORD	172542	: KW11-P COUNT SET BUFFER REGISTER
3681	003146	172544	PKRB: .WORD	172544	: KW11-P COUNTER REGISTER
3682	003150	000100	LCVEC: .WORD	100	: KW11-L VECTOR STORAGE
3683	003152	000104	PCVEC: .WORD	104	: KW11-P VECTOR STORAGE
3684	003154	000000	HZ: .WORD	0	: LINE FREQ. FLAG - 0 = 60 HZ.
3685		000114	MEMVEC=114		: MEMORY PARITY TRAP VECTOR
3686	003156	000000	TCONLO: .WORD	0	: LO BITS OF CALIBRATION TIME CONSTANT
3687	003160	000000	TCONHI: .WORD	0	: HI BITS OF CALIB TIME CONST
3688	003162	000000	BLWMIN: .WORD	0	: COUNT OF TIMES BELOW MIN
3689	003164	000000	ABVMX1: .WORD	0	: COUNT OF FORWARD TIMES ABOVE MAX
3690	003166	000000	ABVMX2: .WORD	0	: COUNT OF REVERSE TIMES ABOVE MAX
3691	003170	000000	SUMLO1: .WORD	0	: LO BITS OF FORWARD TIME SUM
3692	003172	000000	SUMHI1: .WORD	0	: HI BITS OF FORWARD TIME SUM
3693	003174	000000	SUMLO2: .WORD	0	: LO BITS OF REVERSE TIME SUM
3694	003176	000000	SUMHI2: .WORD	0	: HI BITS OF REVERSE TIME SUM
3695	003200	000000	SAVPAR: .WORD	0	: SAVE WORD FOR PAR CONSTANT
3696	003202	000000	SAVWRD: .WORD	0	: SCRATCH WORD
3697	003204	000010	CYLLST: .BLKW	↑D8	: LIST OF PSEUDO-RAND CYL ADDRESSES
3698	003224	000400	BSSOFT: .BLKW	↑D256	: RECORD OF BAD SECTORS FROM SOFTWARE
3699	004224	000400	BSFACT: .BLKW	↑D256	: RECORD OF BAD SECTORS FROM FACTORY
3700	005224	000006	PRVCMO: .BLKW	6	: PREVIOUS COMMAND STORAGE
3701	005240	000006	COMSTR: .BLKW	6	: CURRENT COMMAND STORAGE
3702	005254	000000	PRMPLO: .WORD	0	: PREV. U.B. MAP REG 0
3703	005256	000000	PRMPHO: .WORD	0	
3704	005260	000000	CRMPLO: .WORD	0	: CURRENT U.B. MAP REG 0
3705	005262	000000	CRMPHO: .WORD	0	
3706	005264	000102	BUFFO: .BLKW	↑D66	: OUTPUT BUFFER 1
3707	005470	000000	LOWOCT: .WORD	0	: LOW 16 BITS OF CONVERTED BINARY NUMBER
3708	005472	000000	HIGOCT: .WORD	0	: HIGH BITS OF CONVERTED BINARY NO.
3709	005474	000000	BUFPRT: .WORD	0	: BUFFER POINTER
3710	005476	000000	RECODE: .WORD	0	: RECOVERY CODE WORD
3711	005500	000000	ERRCOM: .WORD	0	: ERROR COMMAND
3712	005502	000000	DRIVE: .WORD	0	: NO. OF DRIVE IN USE
3713	005504	000000	STALLS: .WORD	0	: CURRENT NO. OF UNIT STALLS TO APPLY
3714	005506	000000	CYLNRD: .WORD	0	: CURRENT CYLINDER NUMBER
3715	005510	000000	FS: .WORD	0	: FIRST SECTOR LIMIT
3716	005512	000000	LS: .WORD	0	: LAST SECTOR LIMIT
3717	005514	000000	NCYL1: .WORD	0	: NEXT CYL SCRATCH WORD
3718	005516	000000	NCYL2: .WORD	0	: NEXT CYL SCRATCH WORD
3719	005520	000000	OFINUS: .WORD	0	: OFFSET IN USE

3720	005522	000000	TRACK:	.WORD	0	; TRACK IN USE
3721	005524	000000	INTCHR:	.WORD	0	; TTY INTERRUPT INPUT WORD
3722	005526	000000	SELECT:	.WORD	0	; ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3723	005530	000000	ONLINE:	.WORD	0	; ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3724	005532	000000	NEWON:	.WORD	0	; NEW LOOK AT ONLINE DRIVES
3725	005534	000000	SCRACH:	.WORD	0	; ALL-PURPOSE SCRATCH WORD
3726	005536	000000	PATRN:	.WORD	0	; DATA PATTERN LIST WORD
3727	005540	000000	XXDPAD:	.WORD	0	; STARTING ADDRESS OF XXDP LOADER
3728	005542	000002	XDPSAV:	.BLKW	2	; XXDP PHYSICAL SAVE ADDRESS
3729	005546	000000	PLOFST:	.WORD	0	; (+) OFFSET VALUE
3730	005550	000000	NGOFST:	.WORD	0	; (-) OFFSET VALUE
3731	005552	000005	SAVPRS:	.BLKW	5	; SAVE INITIAL PARAMS FOR XFER
3732	005564	000000	WDSXFR:	.WORD	0	; NO. OF WORDS ACTUALLY XFERRED
3733	005566	000000	FINCYL:	.WORD	0	; FINAL CYLINDER ADRS
3734	005570	000	FINTRK:	.BYTE	0	; FINAL TRACK ADRS
3735	005571	000	FINSEC:	.BYTE	0	; FINAL SECTOR ADRS
3736	005572	000000	LASTWC:	.WORD	0	; ACTUAL FINAL WC
3737	005574	157776	MAHILM:	.WORD	157776	; MA UPPER LIMIT
3738	005576	000077		.WORD	77	
3739	005600	064574	MA:	.WORD	RWBUF	; LO BITS OF PHYS MEM ADRS
3740	005602	000000		.WORD	0	; HI BITS OF PHYS MEM ADRS
3741	005604	000002	PMA:	.BLKW	2	; PARTIAL TRANSFER MEMORY START ADRS

3745 ;LIST OF DESIRED DRIVES TO TEST (IF BYTE=0, DON'T TEST DRIVE)

3746	005610		DRVLST:			
3747	005610	001		.BYTE	1	; DRIVE 0
3748	005611	001		.BYTE	1	; DRIVE 1
3749	005612	001		.BYTE	1	; DRIVE 2
3750	005613	001		.BYTE	1	; DRIVE 3
3751	005614	001		.BYTE	1	; DRIVE 4
3752	005615	001		.BYTE	1	; DRIVE 5
3753	005616	001		.BYTE	1	; DRIVE 6
3754	005617	001		.BYTE	1	; DRIVE 7

3758 ;LIST OF ITERATION COUNTS FOR DESIRED TESTS (IF=0, TEST NOT RUN)
 3759 ;MAXIMUM ALLOWABLE NUMBER OF ITERATIONS = 77776 (OCT)

3760	005620		TSTLST:			
3761	005620	000002		.WORD	2	; TEST 1
3762	005622	000002		.WORD	2	; TEST 2
3763	005624	000002		.WORD	2	; TEST 3
3764	005626	000002		.WORD	2	; TEST 4
3765	005630	000002		.WORD	2	; TEST 5
3766	005632	000100		.WORD	100	; TEST 6

3770 ;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS

3771	005634		DFLTST:			
3772	005634	000002		.WORD	2	; TEST 1
3773	005636	000002		.WORD	2	; TEST 2
3774	005640	000002		.WORD	2	; TEST 3
3775	005642	000002		.WORD	2	; TEST 4

3776	005644	000002	.WORD	2	;TEST 5
3777	005646	000100	.WORD	100	;TEST 6
3778					
3779					
3780		000006			
3781					
3782					
3783					
3784					
3785					
3786					
3787					
3788					
3789					
3790					
3791					
3792					
3793					
3794					
3795					
3796					
3797					
3798					
3799					

NMTSTS=(DFLTST-TSTLST)/2 ;TOTAL NO. OF AUTOMATIC TESTS

:OPERATING PARAMETER LIST

3794	005650		PRMLST:		
3795	005650	000000	FC:	.WORD	0 ;FIRST CYLINDER
3796	005652	000632	LC:	.WORD	632 ;LAST CYLINDER
3797	005654	000000	FT:	.WORD	0 ;FIRST TRACK
3798	005656	000002	LT:	.WORD	2 ;LAST TRACK
3799	005660	000000	SO:	.WORD	0 ;FIRST SECTOR IF 20(DEC) SECTOR FMT
3800	005662	000023	S1:	.WORD	23 ;LAST SECTOR IF 20(DEC) SECTOR FMT
3801	005664	000000	S2:	.WORD	0 ;FIRST SECTOR IF 22(DEC) SECTOR FMT
3802	005666	000025	S3:	.WORD	25 ;LAST SECTOR IF 22(DEC) SECTOR FMT
3803	005670	000000	PT:	.WORD	0 ;DATA PATTERN SELECT WORD
3804	005672	000000	CS:	.WORD	0 ;CONTROL SWITCH WORD
3805	005674	000000	ST:	.WORD	0 ;NUMBER OF UNIT STALLS

:DEFAULT OPERATING PARAMETER VALUES

3800	005676	000000	PROFLT:	.WORD	0 ;FC DEFAULT
3801	005700	000632		.WORD	632 ;LC DEFAULT
3802	005702	000000		.WORD	0 ;FT DEFAULT
3803	005704	000002		.WORD	2 ;LT DEFAULT
3804	005706	000000		.WORD	0 ;SO DEFAULT
3805	005710	000023		.WORD	23 ;S1 DEFAULT
3806	005712	000000		.WORD	0 ;S2 DEFAULT
3807	005714	000025		.WORD	25 ;S3 DEFAULT
3808	005716	000000		.WORD	0 ;PT DEFAULT
3809	005720	000000		.WORD	0 ;CS DEFAULT
3810	005722	000000		.WORD	0 ;ST DEFAULT

:OPERATING PARAMETER VALUE LOW AND HIGH LIMITS

3815	005724		PRMLIM:	.WORD	0 ;FC LIMITS
3816	005724	000000		.WORD	0 ;FC LIMITS
3817	005726	000631		.WORD	631 ;LC LIMITS
3818	005730	000000		.WORD	0 ;LC LIMITS
3819	005732	000632		.WORD	632 ;FC LIMITS
3820	005734	000000		.WORD	0 ;FT LIMITS
3821	005736	000002		.WORD	2 ;LT LIMITS
3822	005740	000000		.WORD	0 ;LT LIMITS
3823	005742	000002		.WORD	2 ;LT LIMITS
3824	005744	000000		.WORD	0 ;SO LIMITS
3825	005746	000023		.WORD	23 ;S1 LIMITS
3826	005750	000000		.WORD	0 ;S1 LIMITS
3827	005752	000023		.WORD	23 ;S1 LIMITS
3828	005754	000000		.WORD	0 ;S2 LIMITS
3829	005756	000025		.WORD	25 ;S2 LIMITS
3830	005760	000000		.WORD	0 ;S3 LIMITS
3831	005762	000025		.WORD	25 ;S3 LIMITS

3832	005764	000000	.WORD	0	;PT LIMITS
3833	005766	177777	.WORD	177777	
3834	005770	000000	.WORD	0	;CS LIMITS
3835	005772	000062	.WORD	000062	
3836	005774	000000	.WORD	0	;ST LIMITS
3837	005776	177777	.WORD	177777	

3838
3839
3840

;ASCII PARAMETER MNEMONICS
PRMNEM:

3842	006000		.ASCII	/FC/
3843	006000	041506	.ASCII	/LC/
3844	006002	041514	.ASCII	/FT/
3845	006004	052106	.ASCII	/LT/
3846	006006	052114	.ASCII	/SO/
3847	006010	030123	.ASCII	/SI/
3848	006012	C30523	.ASCII	/S2/
3849	006014	031123	.ASCII	/S3/
3850	006016	031523	.ASCII	/PT/
3851	006020	052120	.ASCII	/CS/
3852	006022	051503	.ASCII	/ST/
3853	006024	052123	.ASCII	

3854
3855
3856

;DATA PATTERN 00
PAT00: HI-LO FREQ. MIX

3859	006026			177777
3860	006026	177777		177777
3861	006030	177777		177777
3862	006032	177777		177777
3863	006034	052525		052525
3864	006036	052525		052525
3865	006040	052525		052525
3866	006042	177777		177777
3867	006044	177777		177777
3868	006046	052525		052525
3869	006050	052525		052525
3870	006052	177777		177777
3871	006054	052525		052525
3872	006056	177252		177252
3873	006060	177252		177252
3874	006062	172765		172765
3875	006064	172765		172765

3876
3877
3878

;DATA PATTERN 01
PAT01: HI FREQ. PHASE MIX

3881	006066			000000
3882	006066	000000		000000
3883	006070	000000		000000
3884	006072	000000		000000
3885	006074	177777		177777
3886	006076	177777		177777
3887	006100	177777		177777

3888	006102	000000	000000
3889	006104	000000	000000
3890	006106	177777	177777
3891	006108	177777	177777
3892	006112	000000	000000
3893	006114	177777	177777
3894	006116	000000	000000
3895	006120	177777	177777
3896	006122	000000	000000
3897	006124	177777	177777
3898			
3899			
3900			

```

;DATA PATTERN 02
; PAT02: LO FREQ. PHASE MIX

```

3903	006126		
3904	006126	052525	052525
3905	006130	052525	052525
3906	006132	052525	052525
3907	006134	125252	125252
3908	006136	125252	125252
3909	006140	125252	125252
3910	006142	052525	052525
3911	006144	052525	052525
3912	006146	125252	125252
3913	006150	125252	125252
3914	006152	052525	052525
3915	006154	125252	125252
3916	006156	052525	052525
3917	006160	125252	125252
3918	006162	052525	052525
3919	006164	125252	125252
3920			
3921			
3922			

```

;DATA PATTERN 03
; PAT03: MAX. PRECOMP. PHASE MIX

```

3925	006166		
3926	006166	133333	133333
3927	006170	066666	066666
3928	006172	155555	155555
3929	006174	155555	155555
3930	006176	133333	133333
3931	006200	066666	066666
3932	006202	066666	066666
3933	006204	155555	155555
3934	006206	155555	155555
3935	006210	133333	133333
3936	006212	133333	133333
3937	006214	133333	133333
3938	006216	133333	133333
3939	006220	133333	133333
3940	006222	133333	133333
3941	006224	133333	133333
3942			
3943			

4000	006350	000000	000000
4001	006352	000000	000000
4002	006354	000000	000000
4003	006356	000000	000000
4004	006360	000000	000000
4005	006362	000000	000000
4006	006364	000000	000000

4007			
4008			
4009			
4010			
4011			
4012	006366		
4013	006366	177777	177777
4014	006370	177777	177777
4015	006372	177777	177777
4016	006374	177777	177777
4017	006376	177777	177777
4018	006400	177777	177777
4019	006402	177777	177777
4020	006404	177777	177777
4021	006406	177777	177777
4022	006410	177777	177777
4023	006412	177777	177777
4024	006414	177777	177777
4025	006416	177777	177777
4026	006420	177777	177777
4027	006422	177777	177777
4028	006424	177777	177777

```

;DATA PATTERN 07
;
;PAT07: ALL ONES
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777

```

4029			
4030			
4031			
4032			
4033	006426		
4034	006426	000001	000001
4035	006430	000002	000002
4036	006432	000004	000004
4037	006434	000010	000010
4038	006436	000020	000020
4039	006440	000040	000040
4040	006442	000100	000100
4041	006444	000200	000200
4042	006446	000400	000400
4043	006450	001000	001000
4044	006452	002000	002000
4045	006454	004000	004000
4046	006456	010000	010000
4047	006460	020000	020000
4048	006462	040000	040000
4049	006464	100000	100000

```

;DATA PATTERN 08
;
;PAT08: SHIFTED 1 IN FIELD OF ZEROS
000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

```

4050			
4051			
4052			
4053			
4054			
4055	006466		

```

;DATA PATTERN 09
;
;PAT09: SHIFTED 0 IN FIELD OF ONES

```


4056	006466	177776	177776
4057	006470	177775	177775
4058	006472	177773	177773
4059	006474	177767	177767
4060	006476	177757	177757
4061	006500	177737	177737
4062	006502	177677	177677
4063	006504	177577	177577
4064	006506	177377	177377
4065	006510	176777	176777
4066	006512	175777	175777
4067	006514	173777	173777
4068	006516	167777	167777
4069	006520	157777	157777
4070	006522	137777	137777
4071	006524	077777	077777

```

;DATA PATTERN 10
:
PAT10: ALTERNATING 0-1

```

4077	006526		
4078	006526	052525	052525
4079	006530	052525	052525
4080	006532	052525	052525
4081	006534	052525	052525
4082	006536	052525	052525
4083	006540	052525	052525
4084	006542	052525	052525
4085	006544	052525	052525
4086	006546	052525	052525
4087	006550	052525	052525
4088	006552	052525	052525
4089	006554	052525	052525
4090	006556	052525	052525
4091	006560	052525	052525
4092	006562	052525	052525
4093	006564	052525	052525

```

;DATA PATTERN 11
:
PAT11: ALTERNATING 1-0

```

4094			
4095			
4096			
4097			
4098			
4099	006566		
4100	006566	125252	125252
4101	006570	125252	125252
4102	006572	125252	125252
4103	006574	125252	125252
4104	006576	125252	125252
4105	006600	125252	125252
4106	006602	125252	125252
4107	006604	125252	125252
4108	006606	125252	125252
4109	006610	125252	125252
4110	006612	125252	125252
4111	006614	125252	125252

4112	006616	125252	125252
4113	006620	125252	125252
4114	006622	125252	125252
4115	006624	125252	125252

4116
4117
4118
4119
4120

```

;DATA PATTERN 12
;PAT12: SHIFTING ZEROS AND ONES

```

4121	006626		000001
4122	006626	000001	000001
4123	006630	000003	000003
4124	006632	000007	000007
4125	006634	000017	000017
4126	006636	000037	000037
4127	006640	000077	000077
4128	006642	000177	000177
4129	006644	000377	000377
4130	006646	000777	000777
4131	006650	001777	001777
4132	006652	003777	003777
4133	006654	007777	007777
4134	006656	017777	017777
4135	006660	037777	037777
4136	006662	077777	077777
4137	006664	177777	177777

4138
4139
4140

```

;DATA PATTERN 13
;PAT13: COMPOSITE ROTATING

```

4141			
4142			
4143	006666		072307
4144	006666	072307	072307
4145	006670	135143	135143
4146	006672	156461	156461
4147	006674	167230	167230
4148	006676	073514	073514
4149	006700	035646	035646
4150	006702	016723	016723
4151	006704	107351	107351
4152	006706	143564	143564
4153	006710	061672	061672
4154	006712	030735	030735
4155	006714	114356	114356
4156	006716	046167	046167
4157	006720	123073	123073
4158	006722	151453	151453
4159	006724	164616	164616

4160
4161
4162
4163

```

;DATA PATTERN 14
;PAT14: PSEUDO-RANDOM (COMPUTED BY PROGRAM)

```

4164			
4165	006726		.WORD
4166	006726	000000	.WORD
4167	006730	000000	.WORD

4168	006738	000000	.WORD
4169	006739	000000	.WORD
4170	00673A	000000	.WORD
4171	00673B	000000	.WORD
4172	00673C	000000	.WORD
4173	00673D	000000	.WORD
4174	00673E	000000	.WORD
4175	00673F	000000	.WORD
4176	006740	000000	.WORD
4177	006741	000000	.WORD
4178	006742	000000	.WORD
4179	006743	000000	.WORD
4180	006744	000000	.WORD
4181	006745	000000	.WORD
4182	006746	000000	.WORD
4183	006747	000000	.WORD
4184	006748	000000	.WORD
4185	006749	000000	.WORD
4186	00674A	000000	.WORD
4187	00674B	000000	.WORD
4188	00674C	000000	.WORD
4189	00674D	000000	.WORD
4190	00674E	000000	.WORD
4191	00674F	000000	.WORD
4192	006750	000000	.WORD
4193	006751	000000	.WORD
4194	006752	000000	.WORD
4195	006753	000000	.WORD
4196	006754	000000	.WORD
4197	006755	000000	.WORD
4198	006756	000000	.WORD
4199	006757	000000	.WORD
4200	006758	000000	.WORD
4201	006759	000000	.WORD
4202	00675A	000000	.WORD
4203	00675B	000000	.WORD
4204	00675C	000000	.WORD
4205	00675D	000000	.WORD
4206	00675E	000000	.WORD
4207	00675F	000000	.WORD
4208	006760	000000	.WORD
4209	006761	000000	.WORD
4210	006762	000000	.WORD
4211	006763	000000	.WORD
4212	006764	000000	.WORD

.USER-DEFINED DATA PATTERN 15
PAT15:

4186	006766	072307	.WORD 00
4187	006767	135143	.WORD 01
4188	006768	156461	.WORD 02
4189	006769	167230	.WORD 03
4190	00676A	073514	.WORD 04
4191	007000	035646	.WORD 05
4192	007002	016723	.WORD 06
4193	007004	107351	.WORD 07
4194	007006	143564	.WORD 10
4195	007010	061672	.WORD 11
4196	007012	030735	.WORD 12
4197	007014	114356	.WORD 13
4198	007016	046167	.WORD 14
4199	007020	123073	.WORD 15
4200	007022	151453	.WORD 16
4201	007024	164616	.WORD 17

4205	057467	MINL01=1024375	:LO BITS OF SPEC'D MIN ROT. LATENCY
4206	000000	MINHI1=0	:HI BITS OF SPEC'D MIN ROT. LATENCY
4207	062031	MAXL01=1025625	:LO BITS OF SPEC'D MAX ROT. LATENCY
4208	000000	MAXHI1=0	:HI BITS OF SPEC'D MAX ROT. LATENCY
4209	017500	MAXL02=108000	:LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
4210	000000	MAXHI2=0	:HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
4211	112160	MAXL03=1038000	:LO BITS OF SPEC'D MAX AVG SEEK TIME
4212	000000	MAXHI3=0	:HI BITS OF SPEC'D MAX AVG SEEK TIME
4213	022370	MAXL04=109464	:LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
4214	000001	MAXHI4=1	:HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
4215	002734	RNDSHT=101500	:SHORT RANDOM SEEKS = 1500(10)
4216	016514	RNDLNG=107500	:LONG RANDOM SEEKS = 7500(10)
4217	000632	LSTCYL=632	:LAST CYL. = 410(10)
4218	000002	LSTRK=2	:LAST TRACK = 2
4219	000365	ALNCYL=365	:ALIGNMENT CYLINDER =245(10)
4220	000002	BSERR=BIT1	:BSE ERROR
4221	000004	HVRCER=BIT2	:HVRC ERROR
4222	000010	OPIERR=BIT3	:OPI ERROR
4223	000020	DCKERR=BIT4	:DATA CHECK ERROR

4220	000040	ECCNC=BIT5	:ECC NON-CORRECTABLE
4221	000100	WCERR=BIT6	:WRITE CHECK ERROR
4222	000200	ABORT=BIT7	:ABORT
4223	000400	LEV2ER=BIT8	:LEVEL TWO ERROR
4224	001000	BADSEC=BIT9	:BAD SECTOR FLAG
4225	002000	TWOTOS=BIT10	:TWO TIME OUTS
4226	004000	RCLREQ=BIT11	:RECALIBRATE REQUIRED
4227	010000	DRNAVL=BIT12	:DRIVE NOT RELSD BY OTHER PCPT
4228	100000	ANYDER=BIT15	:ANY ERROR DETECTED FLAG
4229	007026	005015 055104 033122	DZR6N: .ASCIZ <15><12>/DZR6N-B/
4230	007034	026516 000102	
4231	007040	026440 051040 033113	SUBVER: .ASCIZ 3 - RK611/RK06 SUBSYSTEM VERIFICATION : PART 3
4232	007046	030461 051057 030113	
4233	007054	020066 052523 051502	
4234	007062	051531 042524 020115	
4235	007070	042526 044522 044506	
4236	007076	040503 044524 047117	
4237	007104	035040 050040 051101	
4238	007112	020124 000	
4239	007115	061 005015 000	PART1: .ASCIZ /1/<15><12>
4240	007121	062 005015 000	PART2: .ASCIZ /2/<15><12>
4241	007125	015 005012 040514	LSTMEM: .ASCIZ <15><12><12>/LAST PHYS MEM ADR = /
4242	007132	052123 050040 054510	
4243	007140	020123 042515 020115	
4244	007146	042101 020122 020075	
4245	007154	000	
4246	007155	117 042526 046122	OVLODR: .ASCIZ /OVERLAY LOADER ? (Y OR N) * /
4247	007162	054501 046040 040517	
4248	007170	042504 020122 020077	
4249	007176	054450 047440 020122	
4250	007204	024516 025040 000040	
4251	007212	005015 040520 040522	PRMIMP: .ASCIZ <15><12>/PARAMETER INPUT MODE/<15><12><12>
4252	007220	042515 042524 020122	
4253	007226	047111 052520 020124	
4254	007234	047515 042504 005015	
4255	007242	000012	
4256	007244	045522 033060 041040	RKBADR: .ASCIZ /RK06 BUS ADR = /
4257	007252	051525 040440 051104	
4258	007260	036440 000040	
4259	007264	045522 033060 053040	RKVADR: .ASCIZ /RK06 VEC ADR = /
4260	007272	041505 040440 051104	
4261	007300	036440 000040	
4262	007304	045522 033060 050040	RKPRTY: .ASCIZ /RK06 PRIORITY = /
4263	007312	044522 051117 052111	
4264	007320	020131 000075	
4265	007324	053523 020122 020075	SWRMSG: .ASCIZ /SWR = /
4266	007332	000	
4267	007333	040 047040 053505	NEWMSG: .ASCIZ / NEW = /
4268	007340	036440 000040	
4269	007344	005015 051104 053111	DRVSEQ: .ASCIZ <15><12>/DRIVE(S) = /
4270	007352	024105 024523 036440	
4271	007360	000040	
4272	007362	051104 053111 020105	BADDRV: .ASCIZ /DRIVE /
4273	007370	020040 000	
4274	007373	116 047117 042455	NXDRIV: .ASCIZ /NON-EXISTENT/<15><12>

4280	007400	044530	052123	.047105	
4281	007406	006524	000012		
4282	007412	047516	020124	042522	NTREDY: .ASCIZ /NOT READY/<15><12>
4283	007420	042101	006531	000012	
4284	007426	051127	052111	026505	WRTLOK: .ASCIZ WRITE-LOCKED/<15><12>
4285	007434	047514	045503	042105	
4286	007442	005015	000		
4287	007445	114	040517	042504	ALNPAK: .ASCIZ /LOADED WITH ALIGN PACK/<15><12>
4288	007452	020104	044527	044124	
4289	007460	040440	044514	047107	
4290	007466	050040	041501	006513	
4291	007474	000012			
4292	007476	043111	054040	042130	PEPLPK: .ASCIZ /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", & REPLACE IT : /
4293	007504	020120	040520	045503	
4294	007512	047440	020116	051104	
4295	007520	020126	026060	052040	
4296	007526	050131	020105	054442	
4297	007534	036040	051103	021076	
4298	007542	020054	020046	051040	
4299	007550	050105	040514	042503	
4300	007556	044440	020124	020072	
4301	007564	000			
4302	007565	015	005012	025052	NOVRTS: .ASCII <15><12><12>/** NO DRIVES TO TEST/<15><12>
4303	007572	047040	020117	051104	
4304	007600	053111	051505	052040	
4305	007606	020117	042524	052123	
4306	007614	005015			
4307	007616	051120	051505	020123	CNTRDY: .ASCIZ /PRESS "CONT" WHEN RDY/<15><12>
4308	007624	041442	047117	021124	
4309	007632	053440	042510	020116	
4310	007640	042122	006531	000012	
4311	007646	040510	052114	051040	HLTRQD: .ASCIZ /HALT REQUESTED/<15><12>
4312	007654	050505	042525	052123	
4313	007662	042105	005015	000	
4314	007667	104	044522	042526	DROXDP: .ASCIZ /DRIVE 0 IS LOAD MEDIUM/<15><12>
4315	007674	030040	044440	020123	
4316	007702	047514	042101	046440	
4317	007710	042105	052511	006515	
4318	007716	000012			
4319	007720	005015	052012	020117	ALDRVS: .ASCIZ <15><12><12>/TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>/<15><12>/*
4320	007726	042524	052123	040440	
4321	007734	046114	042040	044522	
4322	007742	042526	020123	054524	
4323	007750	042520	021040	021101	
4324	007756	036040	051103	026076	
4325	007764	042440	051514	020105	
4326	007772	041474	037122	005015	
4327	010000	020052	000		
4328	010003	015	046012	036440	TSTMDS: .ASCII <15><12>/L = LIST TESTS/<15><12>
4329	010010	046040	051511	020124	
4330	010016	042524	052123	006523	
4331	010024	012			
4332	010025	103	036440	041440	.ASCII /C = CHANGE TESTS/<15><12>
4333	010032	040510	043516	020105	
4334	010040	042524	052123	006523	
4335	010046	012			

4336	010047	111	036440	044440	.ASCIZ	/I = INPUT PARAMS, RUN TESTS/<15>'12>
4337	010054	050116	052125	050040		
4338	010062	051101	046501	026123		
4339	010070	051040	047125	052040		
4340	010076	051505	051524	005015		
4341	010104	000				
4342	010105	015	042412	052116	ENTLCI: .ASCIZ	<15><12>/ENTER L,C, OR I/<15><12>/* /
4343	010112	051105	046040	041454		
4344	010120	020054	051117	044440		
4345	010126	005015	020052	000		
4346	010133	124	020117	042504	DFTEST: .ASCIZ	/TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12>/* /
4347	010140	040506	046125	020124		
4348	010146	042524	052123	020123		
4349	010154	054524	042520	042040		
4350	010162	036040	051103	026076		
4351	010170	042440	051514	020105		
4352	010176	041474	037122	005015		
4353	010204	020052	000			
4354	010207	015	052012	051505	TLSTHD: .ASCIZ	<15><12>/TEST ITERATIONS/<15><12>
4355	010214	020124	020040	020040		
4356	010222	052111	051105	052101		
4357	010230	047511	051516	005015		
4358	010236	000				
4359	010237	015	052012	036440	EXPLAN: .ASCII	<15><12>/T = TYPE PARAM LIST/<15><12>
4360	010244	052040	050131	020105		
4361	010252	040520	040522	020115		
4362	010260	044514	052123	005015		
4363	010266	020117	020075	050117	.ASCII	/O = OPEN PARAM LIST/<15><12>
4364	010274	047105	050040	051101		
4365	010302	046501	046040	051511		
4366	010310	006524	012			
4367	010313	123	036440	051440	.ASCII	/S = SET INDIVIDUAL PARAM/<15><12>
4368	010320	052105	044440	042116		
4369	010326	053111	042111	040525		
4370	010334	020114	040520	040522		
4371	010342	006515	012			
4372	010345	122	036440	051040	.ASCIZ	/R = RUN TESTS/<15><12>
4373	010352	047125	052040	051505		
4374	010360	051524	005015	000		
4375	010365	015	042412	052116	PARAMDE: .ASCIZ	<15><12>/ENTER T,O,S, OR R/<15><12>
4376	010372	051105	052040	047454		
4377	010400	051454	020054	051117		
4378	010406	051040	005015	000		
4379	010413	015	025012	042040	DUACES: .ASCIZ	<15><12>/* DUAL-ACCESS DATA TEST */<15><12>
4380	010420	040525	026514	041501		
4381	010426	042503	051523	042040		
4382	010434	052101	020101	042524		
4383	010442	052123	025040	005015		
4384	010450	000				
4385	010451	115	054101	053440	MAWRDC: .ASCIZ	/MAX WORD COUNT = /
4386	010456	051117	020104	047503		
4387	010464	047125	020124	020075		
4388	010472	000				
4389	010473	052	020052	051440	SECNLI: .ASCII	/** SO>S1/
4390	010500	037060	030523			
4391	010504	047040	052117	040440	NOTALD: .ASCIZ	/ NOT ALLOWED/<15><12>

4392	010512	046114	053517	042105	
4393	010520	005015	000		
4394	010523	052	020052	051440	SECNL2: .ASCIZ /** S2>S3/
4395	010530	037062	031523	000	
4396	010535	052	020052	043040	TRKNLW: .ASCIZ /** FT>LT/
4397	010542	037124	052114	000	
4398	010547	052	020052	053440	WC2BIG: .ASCIZ /** WC OR MA TOO LARGE/<15><12>
4399	010554	020103	051117	046440	
4400	010562	020101	047524	020117	
4401	010570	040514	043522	006505	
4402	010576	000012			
4403	010600	047524	042040	043105	DFQUES: .ASCIZ /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR> /<15><12>/** /
4404	010606	052501	052114	040440	
4405	010614	046114	050040	051101	
4406	010622	046501	026123	052040	
4407	010630	050131	020105	020104	
4408	010636	041474	037122	020054	
4409	010644	046105	042523	036040	
4410	010652	051103	006476	025012	
4411	010660	000040			
4412	010662	005015	051525	051105	PFIFTN: .ASCIZ <15><12>/USER-DEFINED PATTERN 15 :/<15><12>
4413	010670	042055	043105	047111	
4414	010676	042105	050040	052101	
4415	010704	042524	047122	030440	
4416	010712	020065	006472	000012	
4417	010720	005015	047515	044504	SELP15: .ASCIZ <15><12>/MODIFY PATTERN 15 :/<15><12>
4418	010726	054506	050040	052101	
4419	010734	042524	047122	030440	
4420	010742	020065	006472	000012	
4421	010750	047524	046440	042117	MDFY15: .ASCIZ /TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR> /<15><12>/** /
4422	010756	043111	020131	040520	
4423	010764	052124	051105	020116	
4424	010772	032461	020054	054524	
4425	011000	042520	046440	036040	
4426	011006	051103	026076	042440	
4427	011014	051514	020105	041474	
4428	011022	037122	005015	020052	
4429	011030	000			
4430	011031	015	042412	052116	ENTPAS: .ASCIZ <15><12>/ENTER NO. OF PASSES (1-77777) :/<15><12>/** /
4431	011036	051105	047040	027117	
4432	011044	047440	020106	040520	
4433	011052	051523	051505	024040	
4434	011060	026461	033467	033467	
4435	011066	024467	035040	005015	
4436	011074	020052	000		
4437	011077	015	025012	020052	DROPDR: .ASCIZ <15><12>/** DROPPING DRIVE - 20 ERRORS/<15><12>
4438	011104	042040	047522	050120	
4439	011112	047111	020107	051104	
4440	011120	053111	020105	020055	
4441	011126	030062	042440	051122	
4442	011134	051117	006523	000012	
4443	011142	005015	052012	051505	TSTDRM: .ASCII <15><12><12>/TESTING DRIVE /
4444	011150	044524	043516	042040	
4445	011156	044522	042526	040	
4446	011163	040	005015	000	DRVNO: .ASCIZ //<15><12>
4447	011167	104	044522	042526	DRIV: .ASCIZ /DRIVE /

4448	011174	000040							
4449	011176	040503	052122	020056	CART:	.ASCIZ	/CART. /		
4450	011204	000							
4451	011205	123	051105	020056	SERNM:	.ASCIZ	/SER. NO. /		
4452	011212	047516	020056	000040					
4453	011220	005015	040506	052103	FACTBS:	.ASCIZ	<15><12>/FACTORY /		
4454	011226	051117	020131	000					
4455	011233	012	047523	052106	SOFTBS:	.ASCII	<12>/SOFTWARE /		
4456	011240	040527	042522	040					
4457	011245	102	042101	051440	BDSECT:	.ASCIZ	/BAD SECTORS :/<15><12>		
4458	011252	041505	047524	051522					
4459	011260	035040	005015	000					
4460	011265	040	047040	047117	NOFALS:	.ASCIZ	/ NONE/		
4461	011272	000105							
4462									
4463									
4464									
4465	011274	005015	047412	043106	:MESSAGES USED IN OFFSET-TO-FAILURE TEST				
4466	011302	042523	026524	047524	OFSHED:	.ASCII	<15><12><12>/OFFSET-TO-FAILURE MEASUREMENTS :/<15><12><12>		
4467	011310	043055	044501	052514					
4468	011316	042522	046440	040505					
4469	011324	052523	042522	042515					
4470	011332	052116	020123	006472					
4471	011340	005012							
4472	011342	051124	041501	020113		.ASCII	/TRACK CYLN SECT +OFST -OFST/<15><12>		
4473	011350	041440	046131	020116					
4474	011356	051440	041505	020124					
4475	011364	025440	043117	052123					
4476	011372	020040	047455	051506					
4477	011400	006524	012						
4478	011403	011	020011	020040		.ASCIZ	<011><011>/ (UIN) (UIN)/<15><12>		
4479	011410	052450	047111	020051					
4480	011416	024040	044525	024516					
4481	011424	005015	000						
4482									
4483									
4484	011427	000			UNLOD:	.BYTE	0 ;DRIVE UNLOADED INDICATOR		
4485	011430	000			VERIFY:	.BYTE	0 ;VERIFY MODE FLAG		
4486	011431	015	005012	025052	IDENT:	.ASCIZ	<15><12><12>/*** RK06 HEAD ALIGNMENT AID ***/<15><12>		
4487	011436	020052	045522	033060					
4488	011444	044040	040505	020104					
4489	011452	046101	043511	046516					
4490	011460	047105	020124	044501					
4491	011466	020104	025052	006452					
4492	011474	000012							
4493	011476	047506	020122	042510	FORHLP:	.ASCIZ	/FOR HELP TYPE H, ELSE <CR>/<15><12>		
4494	011504	050114	052040	050131					
4495	011512	020105	026110	042440					
4496	011520	051514	020105	041474					
4497	011526	037122	005015	000					
4498	011533	012	040515	052516	TYPMOD:	.ASCIZ	<12>/MANUAL OR AUTO MODE (M OR A)?/<15><12>		
4499	011540	046101	047440	020122					
4500	011546	052501	047524	046440					
4501	011554	042117	020105	046450					
4502	011562	047440	020122	024501					
4503	011570	006477	000012						

4504	011574	005012	020052	040515	TYPMAN: .ASCIZ	<12><12> /* MANUAL SELECT MODE */ <15><12>
4505	011602	052516	046101	051440		
4506	011610	046105	041505	020124		
4507	011616	047515	042504	025040		
4508	011624	005015	000			
4509	011627	012	047105	042524	ENTDRV: .ASCIZ	<12> / ENTER DRIVE NO. (0-7): / <15><12>
4510	011634	020122	051104	053111		
4511	011642	020105	047516	020056		
4512	011650	030050	033455	035051		
4513	011656	005015	000			
4514	011661	052	020052	042040	NOTRDY: .ASCII	/** DRIVE /
4515	011666	044522	042526	040		
4516	011673	040	047040	052117	DRNRDY: .ASCII	/ NOT READY ! START DRIVE IF NECESSARY. / <15><12>
4517	011700	051040	040505	054504		
4518	011706	020440	020040	052123		
4519	011714	051101	020124	051104		
4520	011722	053111	020105	043111		
4521	011730	047040	041505	051505		
4522	011736	040523	054522	006456		
4523	011744	012				
4524	011745	040	020040	053440	.ASCIZ /	WHEN DRIVE IS READY, PRESS "CONT" ON CPU. / <15><12>
4525	011752	042510	020116	051104		
4526	011760	053111	020105	051511		
4527	011766	051040	040505	054504		
4528	011774	020054	051120	051505		
4529	012002	020123	041442	047117		
4530	012010	021124	047440	020116		
4531	012016	050103	027125	005015		
4532	012024	000				
4533	012025	052	020052	042040	NONEXD: .ASCII	/** DRIVE /
4534	012032	044522	042526	040		
4535	012037	040	047040	047117	DRVNE: .ASCIZ	/ NON-EXISTENT OR OFF-LINE ! PRESS "CONT" TO RESTART. / <15><12>
4536	012044	042455	044530	052123		
4537	012052	047105	020124	051117		
4538	012060	047440	043106	046055		
4539	012066	047111	020105	020041		
4540	012074	051120	051505	020123		
4541	012102	041442	047117	021124		
4542	012110	052040	020117	042522		
4543	012116	052123	051101	027124		
4544	012124	005015	000			
4545	012127	040	020040	041440	HALTER: .ASCIZ	/ CPU WILL HALT. / <15><12>
4546	012134	052520	053440	046111		
4547	012142	020114	040510	052114		
4548	012150	006456	000012			
4549	012154	025052	020040	051104	NOTLOK: .ASCII	/** DRIVE /
4550	012162	053111	020105			
4551	012166	020040	047516	020124	NODRLK: .ASCII	/ NOT WRITE-LOCKED ! PLEASE SET WRITE LOCK SWITCH. / <15><12>
4552	012174	051127	052111	026505		
4553	012202	047514	045503	042105		
4554	012210	020440	050040	042514		
4555	012216	051501	020105	042523		
4556	012224	020124	051127	052111		
4557	012232	020105	047514	045503		
4558	012240	051440	044527	041524		
4559	012246	020110	006456	012		

4560	012253	040	020040	052040	.ASCIZ / THEN PRESS "CONT" ON CPU./<15><12>
4561	012260	042510	020116	051120	
4562	012266	051505	020123	041442	
4563	012274	047117	021124	047440	
4564	012302	020116	050103	027125	
4565	012310	005015	000		
4566	012313	052	020052	046440	MULSEL: .ASCIZ /** MULTIPLE DRIVES AUTO-SELECTED ! PRESS 'CONT' TO RESTART./<15><12>
4567	012320	046125	044524	046120	
4568	012326	020105	051104	053111	
4569	012334	051505	040440	052125	
4570	012342	026517	042523	042514	
4571	012350	052103	042105	020440	
4572	012356	050040	042522	051523	
4573	012364	023440	047503	052116	
4574	012372	020047	047524	051040	
4575	012400	051505	040524	052122	
4576	012406	006456	000012		
4577	012412	040412	044514	047107	ENTMOD: .ASCIZ <12>/ALIGN,VERIFY, OR EXERCISE (A,V, OR E) ?/<15><12>
4578	012420	053054	051105	043111	
4579	012426	026131	047440	020122	
4580	012434	054105	051105	044503	
4581	012442	042523	024040	026101	
4582	012450	026126	047440	020122	
4583	012456	024505	037440	005015	
4584	012464	000			
4585	012465	012	025012	051040	MANEXR: .ASCII <12><12>/** RANDOM SEEK EXERCISES IN PROGRESS /
4586	012472	047101	047504	020115	
4587	012500	042523	045505	042440	
4588	012506	042530	041522	051511	
4589	012514	051505	044440	020116	
4590	012522	051120	043517	042522	
4591	012530	051523	040		
4592	012533	117	020116	051104	.ASCII /ON DRIVE /
4593	012540	053111	020105		
4594	012544	020040	006452	000012	DRIEXR: .ASCIZ / */<15><12>
4595	012552	005012	020052	040515	MANALN: .ASCIZ <12><12>/** MANUAL SELECT ALIGNMENT */<15><12>
4596	012560	052516	046101	051440	
4597	012566	046105	041505	020124	
4598	012574	046101	043511	046516	
4599	012602	047105	020124	006452	
4600	012610	000012			
4601	012612	005012	020052	040515	MANVRF: .ASCIZ <12><12>/** MANUAL SELECT VERIFY */<15><12>
4602	012620	052516	046101	051440	
4603	012626	046105	041505	020124	
4604	012634	042526	044522	054506	
4605	012642	025040	005015	000	
4606	012647	110	040505	051504	HEDPOS: .ASCIZ /HEADS POSITIONED AT CYLINDER 365 (OCT)/<15><12>
4607	012654	050040	051517	052111	
4608	012662	047511	042516	020104	
4609	012670	052101	041440	046131	
4610	012676	047111	042504	020122	
4611	012704	033063	020065	047450	
4612	012712	052103	006451	000012	
4613	012720	042412	052116	051105	ENTHED: .ASCIZ <12>/ENTER HEAD NO. (0-2):/<15><12>
4614	012726	044040	040505	020104	
4615	012734	047516	020056	030050	

4616	012742	031055	035051	005015	
4617	012750	000			
4618	012751	124	050131	020105	RWNRDY: .ASCIZ /TYPE <R> WHEN READY :/<15><12>
4619	012756	051074	020076	044127	
4620	012764	047105	051040	040505	
4621	012772	054504	035040	005015	
4622	013000	000			
4623	013001	110	040505	020104	HEDSEL: .ASCII /HEAD /
4624	013006	020040	042523	042514	HEADNO: .ASCIZ / SELECTED/<15><12>
4625	013014	052103	042105	005015	
4626	013022	000			
4627	013023	012	025012	040440	TYPAUT: .ASCIZ <12><12> /* AUTO SELECT MODE */<15><12>
4628	013030	052125	020117	042523	
4629	013036	042514	052103	046440	
4630	013044	042117	020105	006452	
4631	013052	000012			
4632	013054	005012	020052	052501	AUTALN: .ASCIZ <12><12> /* AUTO SELECT ALIGNMENT */<15><12>
4633	013062	047524	051440	046105	
4634	013070	041505	020124	046101	
4635	013076	043511	046516	047105	
4636	013104	020124	006452	000012	
4637	013112	005012	020052	052501	AUTVRF: .ASCIZ <12><12> /* AUTO SELECT VERIFY */<15><12>
4638	013120	047524	051440	046105	
4639	013126	041505	020124	042526	
4640	013134	044522	054506	025040	
4641	013142	005015	000		
4642	013145	012	051104	053111	DRISEL: .ASCII <12>/DRIVE /
4643	013152	020105			
4644	013154	020040	042523	042514	DRVSEL: .ASCIZ / SELECTED/<15><12><12>
4645	013162	052103	042105	005015	
4646	013170	000012			
4647	013172	005012	020052	052501	AUTEXR: .ASCII <12><12> /* AUTO SELECT EXERCISES */<15><12>
4648	013200	047524	051440	046105	
4649	013206	041505	020124	054105	
4650	013214	051105	044503	042523	
4651	013222	020123	006452	012	
4652	013227	012	044123	051117	.ASCIZ <12>/SHORT OR LONG (S OR L) ?/<15><12>
4653	013234	020124	051117	046040	
4654	013242	047117	020107	051450	
4655	013250	047440	020122	024514	
4656	013255	037440	005015	000	
4657	013263	105	042530	041522	AUTOEX: .ASCII /EXERCISING DRIVE /
4658	013270	051511	047111	020107	
4659	013276	051104	053111	020105	
4660	013304	006440	000012		AUTODR: .ASCIZ / /<15><12>
4661					
4662	013310	025052	020040	040503	BAD632: .ASCIZ /** CANNOT READ BAD SECTOR TRACK!/<15><12>
4663	013316	047116	052117	051040	
4664	013324	040505	020104	040502	
4665	013332	020104	042523	052103	
4666	013340	051117	052040	040522	
4667	013346	045503	006441	000012	
4668	013354	041536	005015	000	CNTRLC: .ASCIZ /↑C/<15><12>
4669	013361	136	006532	000012	CNTRLZ: .ASCIZ /↑Z/<15><12>
4670	013366	051136	005015	000	CNTRLR: .ASCIZ /↑R/<15><12>
4671	013373	136	006525	000012	CNTRLU: .ASCIZ /↑U/<15><12>

```

4672 013400 043536 005015 000 CNTRLG: .ASCIZ /1G<15><12>
4673 013405 015 005012 000 CR2LF: .ASCIZ <15><12><12>
4674 013411 134 000 BKSLSH: .ASCIZ /\
4675 013413 054 000 COMMA: .ASCIZ /,
4676 013415 040 040 SPACE6: .ASCII / /
4677 013417 040 SPACE4: .ASCII / /
4678 013420 040 SPACE3: .ASCII / /
4679 013421 040 SPACE2: .ASCII / /
4680 013422 000040 SPACE1: .ASCIZ / /
4681 .EVEN
4682 013424 020040 000075 PRMBUF: .ASCIZ / = /
4683 013430 047527 042122 000040 WORDSP: .ASCIZ /WORD /
4684 013436 036440 000040 EQUALS: .ASCIZ / = /
4685 013442 020052 000 PROMPT: .ASCIZ /* /
4686 013445 076 000040 PRMPSP: .ASCIZ /> /
4687
4688 .EVEN
4689
4690
4691
4692
4693 013450 105037 003106 DFSTRT: CLRB MDFLAG ;SET FLAG FOR DEFAULT MODE
4694 013454 105037 003126 CLRB DULACS ;CLEAR DUAL-ACCESS FLAG
4695 013460 105037 003116 CLRB ERRCNT ;CLEAR ERROR COUNT FOR RESTARTS
4696 013464 022737 014520 000042 CMP #DRVTST, @#42 ;SEE IF EOP RETURN ADRS = DRVTST
4697 013472 001003 BNE 4$ ;BR IF NOT DRVTST
4698 013474 012737 017424 000042 MOV #NEWPAS, @#42 ;SET RETURN ADRS = NEWPAS
4699 013502 000405 4$: BR CMSTRT ;PROCEED
4700
4701
4702 013504 112737 000001 003106 PSTART: MOVB #1, MDFLAG ;SET FLAG FOR PARAMETER MODE
4703 013512 105037 003126 CLRB DULACS ;CLEAR DUAL-ACCESS TEST FLAG
4704
4705 013516 012737 000340 177776 CMSTRT: MOV #PR7, @#PS ;BLOCK ALL INTERRUPTS
4706 .SBTTL INITIALIZE THE COMMON TAGS
4707 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
4708 013524 012706 001100 MOV #SCMTAG, R6 ;;FIRST LOCATION TO BE CLEARED
4709 013530 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
4710 013532 022706 001140 CMP #SWR, R6 ;;DONE?
4711 013536 001374 BNE -6 ;;LOOP BACK IF NO
4712 013540 012706 001100 MOV #STACK_SP ;;SETUP THE STACK POINTER
4713 ;;INITIALIZE A FEW VECTORS
4714 013544 012737 055426 000020 MOV #SCOPE, @#IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
4715 013552 012737 000340 000022 MOV #340, @#IOTVEC+2 ;;LEVEL 7
4716 013560 012737 054722 000030 MOV #ERROR, @#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
4717 013566 012737 000340 000032 MOV #340, @#EMTVEC+2 ;;LEVEL 7
4718 013574 012737 056562 000034 MOV #TRAP, @#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
4719 013602 012737 000340 000036 MOV #340, @#TRAPVEC+2;LEVEL 7
4720 013610 012737 055272 000024 MOV #SPWRDN, @#PWRVEC ;;POWER FAILURE VECTOR
4721 013616 012737 000340 000026 MOV #340, @#PWRVEC+2 ;;LEVEL 7
4722 013624 013737 023514 023506 MOV SENDCT, SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
4723 013632 005037 001304 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
4724 013636 005037 001306 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
4725 013642 112737 000001 001115 MOVB #1, $ERMAX ;;ALLOW ONE ERROR PER TEST
4726 013650 012737 013650 001106 MOV #., $LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
4727 013656 012737 013656 001110 MOV #., $LPERR ;;SETUP THE ERROR LOOP ADDRESS

```

```

4729          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
4729          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
4730 013664 013746 000004          MOV      @#ERRVEC, -(SP)      ;;SAVE ERROR VECTOR
4731 013670 012737 013724 000004  MOV      #64$, @#ERRVEC      ;;SET UP ERROR VECTOR
4732 013676 012737 177570 001140  MOV      #DSWR, SWR          ;;SETUP FOR A HARDWARE SWICH REGISTER
4733 013704 012737 177570 001142  MOV      #DDISP, DISPLAY     ;;AND A HARDWARE DISPLAY REGISTER
4734 013712 022777 177777 165220  CMP      #-1, @SWR          ;;TRY TO REFERENCE HARDWARE SWR
4735 013720 001012          BNE      66$                ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
4736          ;;AND THE HARDWARE SWR IS NOT = -1
4737 013722 000403          BR       65$                ;;BRANCH IF NO TIMEOUT
4738 013724 012716 013732          MOV      #65$, (SP)         ;;SET UP FOR TRAP RETURN
4739 013730 000002          RTI
4740 013732 012737 000176 001140 65$:  MOV      #SWREG, SWR        ;;POINT TO SOFTWARE SWR
4741 013740 012737 000174 001142  MOV      #DISPREG, DISPLAY   ;;
4742 013746 012637 000004          MOV      (SP)+, @#ERRVEC    ;;RESTORE ERROR VECTOR
4743
4744 013752 005037 001326          CLR      $PASS              ;;CLEAR PASS COUNT
4745 013756 132737 000200 001341  BITB     #APTSIZE, $ENVM     ;;TEST USER SIZE UNDER APT
4746 013764 001403          BEQ     67$                ;;YES, USE NON-APT SWITCH
4747 013766 012737 001342 001140  MOV      #SSWREG, SWR       ;;NO, USE APT SWITCH REGISTER
4748 013774          67$:
4749 013774 000005          RESET                    ;;CLEAR THE UNIBUS
4750 013776 012737 000006 000004  MOV      #6, @#ERRVEC       ;;SET TIME-OUT VECTOR
4751 014004 005037 000006          CLR      @#ERRVEC+2
4752 014010 012737 030060 000114  MOV      #MPERHD, @#MEMVEC  ;;SET MEM PARITY TRAP VECTOR
4753 014016 012737 000340 000116  MOV      #PR7, @#MEMVEC+2   ;;SET TRAP PRIORITY = 7
4754 014024 004737 027776          JSR     PC, ENBCSR         ;;ENABLE MEMORY PARITY CHECK
4755 014030 105037 003120          CLRB   DRVERS             ;;CLEAR ERROR COUNT FOR CURRENT DRIVE
4756 014034 112737 000001 003136  MOV     #1, HLPVOL         ;;SET HLP FILE OVLD INDICTR
4757 014042 104400 007026          TYPE   ,DZR6N            ;;TYPE PROGRAM I.D. FOR PART 2
4758 014046 104400 007040          TYPE   ,SUBVER
4759 014052 104400 007121          TYPE   ,PART2
4760 014056 105737 003137          TSTB   NOTYPE
4761 014062 001004          BNE    39$                ;;SEE IF OPERATOR NOTE SHOULD BE TYPED
4762 014064 104400 064574          TYPE   ,NOTMSG           ;;BR IF NOT
4763 014070 105237 003137          INCB   NOTYPE            ;;TYPE OPERATOR NOTE
4764 014074 105737 003126          TSTB   DULACS             ;;SET FLAG FOR NEXT TIME
4765 014100 001402          BEQ    40$                ;;SEE IF DUAL-ACCESS DATA TEST
4766 014102 104400 010413          TYPE   ,DUACES           ;;BR IF NOT
4767 014106 012737 027110 000060 40$:  MOV      #KBDHDL, @#TKVEC   ;;TYPE "* DUAL-ACCESS DATA TEST *"
4768 014114 012737 000200 000062  MOV      #PR4, @#TKVEC+2    ;;LOAD VECTOR FOR TTY KBD
4769 014122 013701 003030          MOV      RKVEC, R1         ;;SET KBD PRIORITY = 4
4770 014126 012721 045542          MOV      #I.INTR, (R1)+    ;;ADDR. OF RK06 VECTOR STORAGE
4771 014132 013711 003032          MOV      RKPRI, (R1)       ;;SET IT TO RK06 HANDLER
4772 014136 012737 027702 000250  MOV      #KTERHD, @#MMVEC   ;;SET RK06 PRIORITY
4773 014144 012737 000340 000252  MOV      #PR7, @#MMVEC+2    ;;VECT FOR KT11 FAILURE
4774 014152 105037 003110          CLRB   TSTING             ;;SET PRIOR. = 7 FOR HNDLER
4775 014156 012737 000000 177776  MOV      #PRO, @#PS         ;;CLEAR "RUNNING TESTS" FLAG
4776 014164 012701 005224          MOV      #PRVCM, R1        ;;ALLOW ALL INTERRUPTS AGAIN
4777 014170 012700 000006          MOV      #6, RO            ;;ZERO OUT PREVIOUS COMMAND
4778 014174 005021          CLR    (R1)+              ;;SET COUNTER
4779 014176 005300          DEC    RO                 ;;ZERO A WORD
4780 014200 001375          BNE    42$                ;;BR IF NOT DONE YET
4781 014202 005037 005504          CLR    STALLS             ;;DON'T ALLOW STALLS YET
4782 014206 004737 027562          JSR    PC, GTSWRG         ;;OPEN SOFTWARE SWR FOR MODIFICATION
4783 014212 012737 176543 053170 44$:  MOV      #176543, $HINUM    ;;INIT. PSEUDO-RANDOM NOS.

```

```

4784 014220 012737 123456 053172      MOV      #123456,$LONUM
4785 014226 004737 027302      JSR      PC,$IZMEM      ;SIZE MEMORY, FIX MA LIMIT IN PRMLIM
4786 014232 105737 003125      TSTB    XDPSVD         ;SEE IF XXDP PREVIOUSLY SAVED
4787 014236 001404          BEQ      9$            ;BR IF NOT
4788 014240 004737 030502      JSR      PC,$GETXDP     ;RESTORE SAVED XXDP
4789 014244 105037 003125      CLRB    XDPSVD         ;CLEAR THE FLP
4790 014250 105737 003106      9$:      TSTB    MDFLAG     ;SEE IF DEFAULT MODE
4791 014254 001031          BNE     10$           ;BR IF NOT DEFAULT MODE
4792 014256 013700 001370      MOV     $VECT1,RO      ;GET APT RK06 VECTOR AND PRTY
4793 014262 013737 001374 003026      MOV     $BASE,RKBAS    ;GET APT RK06 BASE ADDRESS
4794 014270 132737 000200 001341      BITB    #BIT7,$ENVM    ;SEE IF NO SIZING
4795 014276 001005          BNE     18$           ;BR IF NO SIZING
4796 014300 012700 120210      MOV     #AVECT1,RO     ;GET DEFAULT VECTOR AND PRIORITY
4797 014304 012737 177440 003026      MOV     #ABASE,RKBAS   ;GET DEFAULT BASE ADDRESS
4798 014312 110037 003030 18$:      MOV     RO,RKVEC       ;STORE VECTOR
4799 014316 105037 003031      CLRB    RKVEC+1        ;CLEAR HI BYTE
4800 014322 000300          SWAB    RO             ;GET PRTY INTO BITS 5-7
4801 014324 042700 177437      BIC     #177437,RO     ;CLEAR OTHER BITS
4802 014330 010037 003032      MOV     RO,RKPRI       ;STORE RK06 PRIORITY
4803 014334 000137 015216      JMP     ALLDRV         ;GO CHECK ALL DRIVES
4804
4805
4806
4807                                ;BEGIN PARAMETER INPUT MODE
4808 014340 104400 007212 10$:      TYPE    ,PRMINP      ;TYPE "PARAMETER INPUT MODE"
4809
4810                                ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
4811 014344 013746 003026      MOV     RKBAS,-(SP)    ;PUT OLD VALUE ON STACK
4812 014350 104400 007244      TYPE    ,RKBADR       ;TYPE "RK06 BUS ADR = "
4813 014354 004737 027452      JSR     PC,$GETPRM    ;TYPE OLD, GET NEW RKBAS VALUE
4814 014360 012637 003026      MOV     (SP)+,RKBAS   ;STORE NEW VALUE
4815                                ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
4816 014364 013746 003030      MOV     RKVEC,-(SP)   ;PUT OLD VALUE ON STACK
4817 014370 104400 007264      TYPE    ,RKVAOR       ;TYPE "RK06 VEC ADR = "
4818 014374 004737 027452      JSR     PC,$GETPRM    ;TYPE OLD, GET NEW RKVEC VALUE
4819 014400 012637 003030      MOV     (SP)+,RKVEC   ;STORE NEW VALUE
4820                                ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
4821 014404 013746 003032 11$:      MOV     RKPRI,-(SP)   ;GET OLD VALUE OF PRIORITY
4822 014410 006316          ASL     (SP)           ;GET IT INTO BITS 0-2
4823 014412 006316          ASL     (SP)
4824 014414 006316          ASL     (SP)
4825 014416 000316          SWAB    (SP)
4826 014420 104400 007304      TYPE    ,RKPRTY       ;TYPE "RK06 PRIORITY = "
4827 014424 004737 027452      JSR     PC,$GETPRM    ;TYPE OLD, GET NEW RKPRI VALUE
4828 014430 012600          MOV     (SP)+,RO
4829 014432 020027 000604      CMP     RO,#4         ;SEE IF AT LEAST LEVEL 4
4830 014436 002414          BLT     12$           ;BR IF NEW VALUE TOO SMALL
4831 014440 020027 000007      CMP     RO,#7         ;SEE IF LEVEL 7 OR LESS
4832 014444 003011          BGT     12$           ;BR IF NEW VALUE TOO LARGE
4833 014446 000300          SWAB    RO           ;GET PRIORITY INTO BITS 5-7
4834 014450 006200          ASR     RO
4835 014452 006200          ASR     RO
4836 014454 006200          ASR     RO
4837 014456 010037 003032      MOV     RO,RKPRI      ;STORE NEW VALUE
4838 014462 104400 001315      TYPE    ,$CRLF
4839 014466 000405          BR      16$

```

```

4840 014470 104400 005264      12$:  TYPE      ,BUFFO      ;ECHO BAD INPUT
4841 014474 104400 001314      TYPE      $QUES
4842 014500 000741      BR        11$      ;GO ASK AGAIN
4843
4844 014502 105037 003116      16$:  CLRB     ERRCNT      ;CLEAR ERROR CNT FOR RESTARTS
4845 014506 012737 014520 000042      MOV      #DRVST, @#42    ;SET UP DUMP MODE RETURN FROM $EOP
4846                                     ;UPON COMPLETION OF REQUESTED PASSES
4847 014514 000137 015310      JMP      CHKLST        ;GO CHECK STATUS OF MARKED DRIVES
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860

```

```

;*****
;SBTTL DESIRED DRIVE INPUT ROUTINE
;THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
;WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
;CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
;DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
;AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
;LIST (DRVLST).
;*****

```

```

4861 014520
4862 014520 012706 001100      MOV      #STACK, SP    ;RESET THE STACK
4863 014524 005037 005504      CLR      STALLS        ;INHIBIT STALL BETWEEN OPERATIONS
4864 014530 004737 027776      JSR      PC, ENBCSR    ;ENABLE MEMORY PARITY CHECK
4865 014534 104400 007720      TYPE    ,ALDRVS        ;ASK IF ALL DRIVES DESIRED
4866 014540 004737 032214      JSR      PC, RDCHRS    ;READ RESPONSE
4867 014544 014520      DRVTST                ;(+C) RETURN ADDRESS
4868 014546 014520      DRVTST                ;(+Z) RETURN ADDRESS
4869 014550 014520      DRVTST                ;(+U) OR ERROR RETURN ADDRESS
4870 014552 005700      TST     R0            ;SEE IF NULL INPUT
4871 014554 001413      BEQ     TELDRV        ;BR IF NULL INPUT
4872 014556 022737 000101 005264      CMP     #'A, BUFFO    ;SEE IF ALL DRIVES REQUESTED
4873 014564 001002      BNE     4$            ;BR IF NOT ALL DRIVES
4874 014566 000137 015216      JMP     ALLDRV        ;CHECK ALL DRIVES
4875 014572 104400 005264      4$:    TYPE    ,BUFFO    ;ECHO BAD INPUT
4876 014576 104400 001314      TYPE    $QUES        ;TYPE '?' AND <CR>, <LF>
4877 014602 000746      BR      DRVTST        ;GO ASK AGAIN
4878
;TYPE CURRENT DRIVE LIST
4879 014604 104400 007344      TELDRV: TYPE   DRVSEQ  ;TYPE "DRIVE(S) = "
4880 014610 005001      CLR     R1            ;INITIALIZE DRIVE NUMBER
4881 014612 005000      CLR     R0            ;INIT. COUNT OF LISTED DRIVES
4882 014614 012703 000001      MOV     #BIT0, R3     ;INIT BIT POINTER
4883 014620 105761 005610      4$:    TSTB    DRVLST(R1) ;SEE IF THIS DRIVE IS LISTED
4884 014624 001414      BEQ     8$            ;BR IF DRIVE NOT LISTED
4885 014626 005700      TST     R0            ;SEE IF FIRST TIME HERE
4886 014630 001402      BEQ     6$            ;BR IF FIRST TIME HERE
4887 014632 104400 013413      TYPE    ,COMMA        ;TYPE A COMMA
4888 014636 010137 005534      6$:    MOV     R1, SCRACH ;USE SCRACH FOR BUFFER
4889 014642 052737 000060 005534      BIS     #'0, SCRACH   ;CONVERT DRIVE NO. TO ASCII
4890 014650 104400 005534      TYPE    ,SCRACH       ;TYPE DRIVE NO.
4891 014654 005200      INC     R0            ;INCREMENT NO. OF LISTED DRIVES
4892 014656 005201      8$:    INC     R1            ;INCREMENT DRIVE NO.
4893 014660 006303      ASL     R3            ;SHIFT BIT POINTER
4894 014662 022701 000010      CMP     #10, R1       ;SEE IF DONE YET
4895 014666 001354      BNE     4$            ;BR IF NOT DONE

```



```

014670 005700          TST      R0          :SEE IF ANY DRIVES WERE LISTED
014672 001006          BNE      26$        :BR IF YES
014674 104400 011265     TYPE    ,NOFALS     :TYPE " NONE"
014700 104400 007565     TYPE    ,NOORTS     :TYPE " ** NO DRIVES TO TEST"
                                "PRESS 'CONT' WHEN RDY"
014704 000137 044420     JMP      HLTPRG     :HALT PROGRAM
014710 104400 001315 26$:     TYPE    ,SCRLF      :TYPE <CR> <LF>
014714 105737 003106     TSTB    MODLAG     :SEE IF DEFAULT MODE
014720 001002          BNE      20$        :BR IF NOT DEFAULT MODE
014722 000137 015642     JMP      LCCFLS     :GO LOAD DEFLT ITER. COUNTS
014726 122737 000013 000041 20$:   CMPB    #13,DR41    :SEE IF RK06 IS XXDP MEDIM
014734 001032          BNE      19$        :BR IF NOT
014736 105737 005610     TSTB    DRVLST     :SEE IF DRIVE 0 LISTED TO TEST
014740 001427          BNE      19$        :BR IF NOT
014742 104400 007476 24$:     TYPE    ,REPLPK     :TYPE "IF XXDP PACK ON DRV 0, TYPE Y <CR>
                                AND REPLACE IT"
014750 004737 032214     JSR     PC,RDCHRS  :READ RESPONSE
014754 014744          R04$          :(<C) RETURN
014756 014744          R04$          :(<Z) RETURN
014760 014744          R04$          :(<U) RETURN
014762 005700          TST      R0          :SEE IF NULL INPUT
014764 001416          BEQ     19$        :BR IF JUST <CR> TYPED
014766 022737 000131 005264   CMP     #'Y,BUFF0   :SEE IF "Y <CR>" TYPED
014774 001363          BNE      24$        :BR IF NOT, TO ASK AGAIN
014776 104400 007616     TYPE    ,CNTROY     :TYPE "PRESS CONT WHEN DRV RDY"
015002 042762 000100 000000   BIC     #IE,RKCS1(R2) :DISABLE RK06 INTERRUPT
015010 000023          HALT                    :HALT FOR PACK CHANGE
015012 004737 030726     JSR     PC,INITSS  :INIT THE SUB-SYS
015016 000137 015310     JMP     CHKLIST    :GO CHECK STATUS OF LISTED DRIVES
015022 104400 013442 19$:     TYPE    ,PROMPT     :TYPE ASTERISK AND SPACE
:READ AND CHECK NEW DRIVE NUMBERS
015026 004737 032214     JSR     PC,RDCHRS  :READ IN INPUT STRING
015032 014520          DRVTST   :(<C) RETURN ADDRESS
015034 014520          DRVTST   :(<Z) RETURN ADDRESS
015036 014604          TELDRV   :(<U) OR ERROR RETURN ADDRESS
015040 005700          TST      R0          :SEE IF NULL INPUT
015042 001007          BNE      9$         :BR IF NEW DRIVES WILL BE SELECTED
015044 105737 003126     TSTB    DULACS     :SEE IF DUAL-ACCESS FLAG SET
015050 001002          BNE      22$        :BR IF YES
015052 000137 015354     JMP     ASKTST     :JUMP TO THE TEST INPUT ROUTINE
015056 000137 016120 22$:     JMP     INPUTP     :GO ASK FOR INPUT PARAMETERS
015062 022700 000017 9$:     CMP     #15.,R0    :SEE IF TOO MANY CHARACTERS TYPED
015066 002005          BGE     12$        :BR IF NOT TOO MANY
015070 104400 005264 10$:     TYPE    ,BUFF0     :ECHO BAD INPUT
015074 104400 001314     TYPE    ,SOLES     :TYPE <?> AND <CR>, <LF>
015100 000641          BR      TELDRV     :GO TYPE DRIVES AND ASK AGAIN
015102 005001 12$:     CLR     R1          :CLEAR CHAR. POINTER
015104 126127 005264 000060 14$:  CMPB    BUFF0(R1),#'0 :SEE IF DRIVE NO. < 0
015112 002766          BLT     10$        :BR TO ECHO BAD INPUT
015114 126127 005264 000067          CMPB    BUFF0(R1),#'7 :SEE IF > 7
015122 003362          BGT     10$        :BR TO ECHO BAD INPUT
015124 005201          INC     R1          :INCREMENT CHAR POINTER
015126 020100          CMP     R1,R0      :SEE IF MORE CHARS TO CHECK
015130 001406          BEQ     16$        :BR IF ALL CHARS CHECKED
015132 126127 005264 000054          CMPB    BUFF0(R1),#', :SEE IF THIS IS COMMA
015140 001353          BNE      10$        :BR IF NOT COMMA

```


C10

```

4952 015142 005201      INC      R1      ;INCREMENT CHAR POINTER
4953 015144 000757      BR      14$     ;BR TO CONTINUE CHECKING CHARS
4954      ;GET NEW DRIVE NUMBERS INTO LIST
4955 015146 012701 005610 16$: MOV     #DRVLST,R1 ;GET DRIVE LIST ADDRESS
4956 015152 005021      CLR     (R1)+   ;CLEAR OUT THE DRIVE LIST
4957 015154 005021      CLR     (R1)+
4958 015156 005021      CLR     (R1)+
4959 015160 005011      CLR     (R1)
4960 015162 005000      CLR     R0      ;CLEAR BUFFER POINTER
4961 015164 116001 005264 18$: MOVB   BUFFD(R0),R1 ;GET A DRIVE NUMBER
4962 015170 042701 000060      BIC     #'D,R1  ;STRIP ASCII BITS
4963 015174 112761 000001 005610 MOVB   #1,DRVLST(R1) ;MARK THIS DRIVE IN DRIVE LIST
4964 015202 062700 000002      ADD    #2,R0    ;POINT TO NEXT DRIVE NUMBER
4965 015206 105760 005263      TSTB   BUFFD-1(R0) ;SEE IF NULL CHAR YET
4966 015212 001364      BNE    18$     ;BR IF NOT DONE YET
4967 015214 000435      BR     CHKLST  ;GO CHECK NEW DRIVES FOR VALID STATUS
4968      ;COME HERE IF ALL DRIVES REQUESTED
4969 015216 005000      ALLDRV: CLR    R0 ;CLEAR DRIVE NUMBER
4970 015220 013703 001376      MOV    $DEVN,R3 ;GET APT DEVICE MAP
4971 015224 132737 000200 001341      BITB   #BIT7,$ENVM ;SEE IF NO SIZING
4972 015232 001002      BNE    1$      ;SEE IF NO SIZING
4973 015234 012703 000377      MOV    #377,R3 ;SET UP FOR SIZING
4974 015240 012701 000001 1$: MOV    #BIT0,R1 ;SET BIT POINTER
4975 015244 105060 005610 2$: CLRB   DRVLST(R0) ;INITIALIZE DRIVE ENTRY
4976 015250 030103      BIT    R1,R3   ;SEE IF THIS DRIVE IS REQUESTED
4977 015252 001403      BEQ    4$      ;BR IF NOT
4978 015254 112760 000001 005610 MOVB   #1,DRVLST(R0) ;MARK THIS DRIVE IN DRIVE LIST
4979 015262 006301 4$: ASL    R1      ;SHIFT BIT POINTER
4980 015264 005200      INC    R0      ;INCREMENT DRIVE NUMBER
4981 015266 022700 000010      CMP    #10,R0  ;SEE IF DONE MARKING ALL DRIVES
4982 015272 001364      BNE    2$      ;BR IF NOT DONE YET
4983 015274 132737 000200 001341      BITB   #BIT7,$ENVM ;SEE IF PROGRAM SHOULD SIZE
4984 015302 001402      BEQ    CHKLST ;BR IF YES
4985 015304 000137 014604      JMP    TELDRV  ;GO LIST APT-SELECTED DRIVES
4986      ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
4987 015310 005000      CHKLST: CLR   R0 ;CLEAR DRIVE NUMBER
4988 015312 105760 005610 2$: TSTB   DRVLST(R0) ;SEE IF THIS DRIVE IS MARKED IN LIST
4989 015316 001405      BEQ    4$      ;BR IF NOT MARKED
4990 015320 010037 005502      MOV    R0,DRIVE ;SET DRIVE NUMBER FOR SCNDRV
4991 015324 004737 031046      JSR    PC,SCNDRV ;CHECK STATUS OF THIS DRIVE
4992      ;IF NOT VALID, TYPE MESSAGE
4993      ;AND REMOVE IT FROM DRIVE LIST
4994 015330 015346 4$: 6$: INC    R0      ;ERROR RETURN ADDRESS FOR SCNDRV
4995 015332 005200      INC    R0      ;RETURN HERE IF DRIVE IS USEABLE
4996 015334 022700 000010      CMP    #10,R0  ;SEE IF DONE CHECKING LIST
4997 015340 001364      BNE    2$      ;BR IF NOT DONE YET
4998 015342 000137 014604      JMP    TELDRV  ;GO BACK TO LIST SELECTED DRIVES
4999 015346 105060 005610 6$: CLRB   DRVLST(R0) ;REMOVE INVALID DRIVE FROM LIST
5000 015352 000767      BR     4$      ;CONTINUE CHECKING THE LIST

*****
5001      ;SBTTL DESIRED TEST INPUT ROUTINE
5002      ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
5003      ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
5004      ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
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
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063

015354 104400 0100C3
015360 104400 010105
015364 004737 032214
015370 014520
015372 015360
015374 015360
015376 005700
015400 001005
015402 104400 005264
015406 104400 001314
015412 000762
015414 022737 000114 005264
015422 001002
015424 000137 015460
015430 022737 000103 005264
015436 001002
015440 000137 015576
015444 022737 000111 005264
015452 001353
015454 000137 016120

015460 104400 010207
015464 005001
015466 004737 027250
015472 004737 032526
015476 104400 001315
015502 005737 005524
015506 001416
015510 122737 000003 005524
015516 001002
015520 000137 014520
015524 122737 000032 005524
015532 001002
015534 000137 015360
015540 004737 027270
015544 062701 000002
015550 010102
015552 062702 005620
015556 022702 005634
015562 001341
015564 042777 000100 163352
015572 000137 015360

;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
;*THAT THE TEST IS NOT TO BE RUN.
;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
;*TYPED. (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
;*OF OPERATING PARAMETERS, AND RUN TESTS.
;*TYPING (IC) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
;* (IZ) CAUSES RETURN TO ASKTMD.
;*****

; DETERMINE L,C, OR I MODE
ASKTST: TYPE ,TSTMDS ; ASK FOR TEST INPUT MODE
ASKTMD: TYPE ,ENTLCI ; TYPE "ENTER L,C, OR I"
 JSR PC,RDCHRS ; READ RESPONSE
 DRVTST ; (IC) RETURN ADDRESS
 ASKTMD ; (IZ) RETURN ADDRESS
 ASKTMD ; (IU) OR ERROR RETURN ADDRESS
 TST RO ; SEE IF ANY INPUT
 BNE 4\$; BR IF ANY INPUT
 2\$: TYPE ,BUFF0 ; ECHO BAD INPUT
 TYPE ,SQUES ; TYPE (?) AND <CR>, <LF>
 BR ASKTMD ; GO ASK AGAIN
 4\$: CMP #'L,BUFF0 ; SEE IF (L) TYPED
 BNE 6\$; BR IF NOT (L)
 JMP LSTTST ; JUMP TO LIST TESTS
 6\$: CMP #'C,BUFF0 ; SEE IF (C) TYPED
 BNE 8\$; BR IF NOT (C)
 JMP CHGTST ; JUMP TO CHANGE TESTS
 8\$: CMP #'I,BUFF0 ; SEE IF (I) TYPED
 BNE 2\$; BR IF NOT (I), TO ECHO BAD INPUT
 JMP INPUTP ; GO INPUT PARAMS AND RUN TESTS

; LIST CURRENT TESTS AND ITERATION COUNTS
LSTTST: TYPE ,TLSTHD ; TYPE HEADING "TEST ITERATIONS"
 CLR R1 ; INITIALIZE TEST INDEX
 2\$: JSR PC,PREPKB ; PREPARE FOR POSSIBLE KBD INPUT
 JSR PC,TYPTST ; TYPE CURRENT TEST AND ITERATION NUMBER
 TYPE ,SCRLF ; TYPE <CR>, <LF>
 INTCHR ; SEE IF ANY INPUT
 BEQ 8\$; BR IF NO INPUT
 CMPB #003,INTCHR ; SEE IF (IC) TYPED
 BNE 4\$; BR IF NOT (IC)
 JMP DRVTST ; JUMP TO ASK FOR DRIVES AGAIN
 4\$: CMPB #032,INTCHR ; SEE IF (IZ) TYPED
 BNE 6\$; BR IF NOT (IZ)
 JMP ASKTMD ; JUMP TO ASK FOR NEW TEST INPUT MODE
 6\$: JSR PC,ECOBAD ; ECHO BAD INPUT
 8\$: ADD #2,R1 ; INCREMENT TEST INDEX
 MOV R1,R2 ; GET COPY OF INDEX
 ADD #TSTLST,R2 ; GET POSITION IN LIST
 CMP #DFLTST,R2 ; SEE IF DONE WITH LIST
 BNE 2\$; BR IF NOT DONE YET
 BIC #BIT6,ISTKS ; DISABLE KBD INTERRUPT
 JMP ASKTMD ; GO ASK FOR NEW TEST INPUT MODE

```

5064          :CHANGE CURRENT TESTS AND ITERATION COUNTS
5065 015576 104400 010133 CHGTST: TYPE OFTEST ;ASK IF TESTS SHOULD BE DEFAULTED
5066 015602 004737 032214 JSR PC,RDCHRS ;READ RESPONSE
5067 015606 014520 DRVTST ;(↑C) RETURN ADDRESS
5068 015610 015360 ASKTMD ;(↑Z) RETURN ADDRESS
5069 015612 015576 CHGTST ;(↑U) OR ERROR RETURN ADDRESS
5070 015614 005700 TST RO ;SEE IF NULL INPUT
5071 015616 001426 BEQ NULINP ;BR IF NULL INPUT
5072 015620 022737 000104 005264 CMP #'D,BUFF0 ;SEE IF (D) TYPED
5073 015626 001405 BEQ LODFLS ;BR IF DEFAULTS REQUESTED
5074 015630 104400 005264 TYPE ,BUFF0 ;ECHO BAD INPUT
5075 015634 104400 001314 TYPE ,SQUES ;TYPE '?' AND <CR>,<LF>
5076 015640 000756 BR CHGTST ;GO ASK AGAIN
5077          ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
5078 015642 012700 005634 LODFLS: MOV #DFLTST,R0 ;LOAD DEFAULT LIST ADDRESS
5079 015646 012702 005620 MOV #TSTLST,R2 ;LOAD TEST LIST ADDRESS
5080 015652 012022 4$: MOV (R0)+,(R2)+ ;LOAD A DEFAULT VALUE
5081 015654 022702 005634 CMP #DFLTST,R2 ;SEE IF DONE YET
5082 015660 001374 BNE 4$ ;BR IF NOT DONE YET
5083 015662 105737 003106 TSTB MDFLAG ;SEE IF DEFAULT MODE
5084 015666 001234 BNE ASKTMD ;BR IF NOT DEFAULT MODE
5085 015670 000137 016654 JMP LODFPT ;GO LOAD DEFAULT PARAMETERS
5086 015674 005001 NULINP: CLR R1 ;INITIALIZE TEST INDEX
5087 015676 104400 010207 TYPE ,TLSTHD ;TYPE HEADING "TEST ITERATIONS"
5088          ;TYPE CURRENT TEST AND ITERATION NUMBER
5089 015702 004737 032526 BS: JSR PC,TYPTST ;TYPE A TEST NO. AND ITERATION NO.
5090 015706 104400 013422 TYPE ,SPACE1 ;TYPE A SPACE
5091 015712 104400 013442 TYPE ,PROMPT ;TYPE ASTERISK AND SPACE
5092          ;READ AND CHECK INPUT, IF ANY
5093 015716 004737 032214 JSR PC,RDCHRS ;READ OCTAL DIGITS TYPED
5094 015722 014520 DRVTST ;(↑C) RETURN ADDRESS
5095 015724 015360 ASKTMD ;(↑Z) RETURN ADDRESS
5096 015726 015702 BS ;(↑U) OR ERROR RETURN ADDRESS
5097 015730 005700 TST RO ;SEE IF ANY INPUT
5098 015732 001012 BNE 12$ ;BR IF ANY INPUT
5099 015734 062701 000002 10$: ADD #2,R1 ;INCREMENT INDEX
5100 015740 010102 MOV R1,R2 ;COPY INDEX
5101 015742 062702 005620 ADD #TSTLST,R2 ;GET POSITION IN LIST
5102 015746 022702 005634 CMP #DFLTST,R2 ;SEE IF DONE WITH LIST
5103 015752 001353 BNE BS ;BR IF NOT DONE YET
5104 015754 000137 015360 JMP ASKTMD ;GO ASK FOR NEW TEST INPUT MODE
5105 015760 022737 000041 005264 12$: CMP #'!,BUFF0 ;SEE IF (!) TYPED
5106 015766 001431 BEQ 16$ ;BR TO PROPAGATE CURRENT ITER. NO.
5107 015770 005002 CLR R2 ;INITIALIZE (!) INDICATOR
5108 015772 122760 000041 005263 CMPB #'!,BUFF0-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
5109 016000 001004 BNE 14$ ;BR IF NOT (!)
5110 016002 105060 005263 CLRB BUFF0-1(R0) ;INSERT TERMINATOR BYTE
5111 016006 005300 DEC RO ;DECREMENT CHAR COUNT
5112 016010 005202 INC R2 ;SET (!) INDICATOR
5113 016012 022700 000005 14$: CMP #5,RO ;SEE HOW MANY CHARS NOW
5114 016016 002433 BLT 20$ ;BR IF TOO MANY (MAX ITER. NO. = 77776)
5115 016020 012746 005264 MOV #BUFF0,-(SP) ;GET BUF ADDR. ON STACK FOR OCTBIN
5116 016024 004737 052736 JSR PC,OCTBIN ;CHECK DIGITS AND CONVERT TO BINARY
5117 016030 016106 20$ ;ERROR RETURN ADDRESS FOR OCTBIN
5118 016032 012600 MOV (SP)+,RO ;GET ITERATION NUMBER
5119 016034 020027 077776 CMP RO,#77776 ;SEE IF > 77776

```

F10

```

5120 016040 101022      BHI      20$      ;BR IF TOO BIG
5121 016042 010061 005620  MOV      R0,TSTLST(R1) ;PUT ITERATION NUMBER INTO TEST LIST
5122 016046 005702      TST      R2        ;SEE IF (!) WAS TYPED
5123 016050 001731      BEQ      10$      ;BR IF NOT (!)
5124      ;PROPAGATE CURRENT ITERATION NO. TO THE END OF TSTLST
5125 016052 010102      16$: MOV      R1,R2      ;COPY INDEX
5126 016054 062702 005620  ADD      #TSTLST,R2    ;GET POSITION IN LIST
5127 016060 022702 005632  CMP      #DFLTST-2,R2 ;SEE IF AT END OF LIST
5128 016064 001002      BNE      17$      ;BR IF NOT DONE
5129 016066 000137 015360  JMP      ASKMD      ;GO ASK FOR NEW TEST INPUT MODE
5130 016072 016161 005620 005622 17$: MOV      TSTLST(R1),TSTLST+2(R1) ;PROPAGATE TO NEXT WORD
5131 016100 062701 000002      ADD      #2,R1        ;INCREMENT INDEX
5132 016104 000762      BR       16$      ;BR TO CONTINUE
5133 016106 104400 005264 20$: TYPE    ,BUFFO    ;ECHO BAD INPUT
5134 016112 104400 001314      TYPE    ,SQUES    ;TYPE (<?) AND <CR>,<LF>
5135 016116 000671      BR       8$       ;GO ASK AGAIN
5136
5137      ;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
5138 016120 104400 010237  INPUT: TYPE    ,EXPLAN  ;EXPLAIN INPUT MODES
5139 016124 005037 005504  ASKMDE: CLR     ,STALLS ;INHIBIT STALL BETWEEN OPERATIONS
5140 016130 004737 027776      JSR     PC,ENBCSR  ;ENABLE MEMORY PARITY CHECK
5141 016134 104400 010365      TYPE    ,PARMDE   ;ASK FOR DESIRED INPUT MODE
5142 016140 104400 013442      TYPE    ,PROMPT   ;TYPE "*"
5143 016144 004737 032214      JSR     PC,RDCHRS ;READ RESPONSE TO MODE QUESTION
5144 016150 014520      DRVTST ;
5145 016152 016124      ASKMDE ;(↑C) RETURN ADDRESS
5146 016154 016124      ASKMDE ;(↑Z) RETURN ADDRESS
5147 016156 005700      ASKMDE ;(↑U) OR ERROR RETURN ADDRESS
5148 016160 001005      TST     R0        ;SEE IF NULL INPUT
5149 016162 104400 005264 2$: TYPE    ,BUFFO    ;ECHO BAD INPUT
5150 016166 104400 001314      TYPE    ,SQUES    ;TYPE (<?) AND <CR>,<LF>
5151 016172 000754      BR       ASKMDE   ;GO ASK AGAIN
5152 016174 022737 000124 005264 4$: CMP      #'T,BUFFO  ;SEE IF (T) TYPED
5153 016202 001002      BNE     6$       ;BR IF NOT (T)
5154 016204 000137 016336      JMP     TYPLST   ;JUMP TO THE TYPE LIST ROUTINE
5155 016210 022737 000117 005264 6$: CMP      #'O,BUFFO ;SEE IF (O) TYPED
5156 016216 001002      BNE     8$       ;BR IF NOT (O)
5157 016220 000137 016610      JMP     OPNLST   ;JUMP TO THE OPEN LIST ROUTINE
5158 016224 022737 000123 005264 8$: CMP      #'S,BUFFO ;SEE IF (S) TYPED
5159 016232 001002      BNE     10$      ;BR IF NOT (S)
5160 016234 000137 017074      JMP     SETPRM   ;JUMP TO THE SET INDIV. PARAM. ROUTINE
5161 016240 022737 000122 005264 10$: CMP      #'R,BUFFO ;SEE IF (R) TYPED
5162 016246 001345      BNE     2$       ;BR IF NOT (R) TO ECHO BAD INPUT
5163 016250 023737 005660 005662  CMP      S0,S1    ;COMPARE 20(DEC) SECTOR LIMITS
5164 016256 003403      BLE     12$      ;BR IF S0 NOT > S1
5165 016260 104400 010473      TYPE    ,SECNL1  ;TYPE "S0>S1 NOT ALLOWED"
5166 016264 000717      BR      ASKMDE   ;GO ASK AGAIN FOR PARAMETERS
5167 016266 023737 005664 005666 12$: CMP      S2,S3    ;COMPARE 22(DEC) SECTOR LIMITS
5168 016274 003405      BLE     14$      ;BR IF S2 NOT > S3
5169 016276 104400 010523      TYPE    ,SECNL2  ;TYPE "S2>S3"
5170 016302 104400 010504      TYPE    ,NOTALD  ;TYPE "NOT ALLOWED"
5171 016306 000706      BR      ASKMDE   ;GO ASK AGAIN FOR PARAMETERS
5172 016310 023737 005654 005656 14$: CMP      FT,LT    ;COMPARE CHOSEN TRACK LIMITS
5173 016316 003405      BLE     20$      ;BR IF FT NOT > LT
5174 016320 104400 010535      TYPE    ,TRKNLW  ;TYPE "FT>LT"
5175 016324 104400 010504      TYPE    ,NOTALD  ;TYPE "NOT ALLOWED"
    
```

```

5176 016330 000675          BR          ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
5177 016332 000137 017330 20$: JMP          RUNTST          ;GO RUN THE TESTS
5178
5179
5180
5181          .SBTTL TYPE (T) LIST ROUTINE
5182          ;*THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
5183          ;*CONSOLE, LINE BY LINE. EACH LINE HAS THE FOLLOWING
5184          ;*FORMAT: XX=YYYYYY, WHERE XX IS A PARAMETER MNEMONIC
5185          ;*AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
5186          ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST,
5187          ;*AND (↑Z) CAUSES RETURN TO ASKMDE.
5188
5189          TYPLST:
5190 016336 005001          CLR          R1          ;INITIALIZE INDEX
5191 016340 004737 027250  JSR          PC,PREPKB  ;PREPARE FOR POSSIBLE KBD INPUT
5192 016344 004737 032562 1$: JSR          PC,TYPPRM ;TYPE CURRENT PARAMETER AND VALUE
5193 016350 104400 001315  TYPE          $CRLF      ;TYPE <CR>, <LF>
5194 016354 005737 005524  TST          INTCHR     ;SEE IF ANY INPUT AT KBD
5195 016360 001420          BEQ          8$          ;BR IF NO INPUT
5196 016362 122737 000003 005524 CMPB         #003,INTCHR ;SEE IF (↑C) TYPED
5197 016370 001002          BNE          4$          ;BR IF NOT (↑C)
5198 016372 000137 014520  JMP          DRVTST     ;JUMP TO ASK FOR DRIVE(S) AGAIN
5199 016376 122737 000032 005524 4$: CMPB         #032,INTCHR ;SEE IF (↑Z) TYPED
5200 016404 001002          BNE          6$          ;BR IF NOT (↑Z)
5201 016406 000137 016124  JMP          ASKMDE     ;JUMP TO ASK FOR NEW MODE
5202 016412 004737 027270 6$: JSR          PC,ECOBAD ;ECHO BAD INPUT
5203 016416 004737 027250  JSR          PC,PREPKB  ;PREPARE FOR POSSIBLE KBD INPUT
5204 016422 022761 040515 006000 8$: CMP          #MA,PRMNM(R1) ;SEE IF PARAM. IS (MA)
5205 016430 001002          BNE          10$         ;BR IF NOT (MA)
5206 016432 062701 000002  ADD          #2,R1      ;INCREMENT PARAMETER INDEX
5207 016436 062701 000002 10$: ADD          #2,R1      ;INCREMENT PARAMETER INDEX
5208 016442 010102          MOV          R1,R2     ;GET COPY OF INDEX
5209 016444 062702 005650  ADD          #PRMLST,R2 ;COMPUTE POSITION IN LIST
5210 016450 022702 005676  CMP          #PRDFLT,R2 ;SEE IF DONE WITH LIST
5211 016454 001333          BNE          1$          ;BR IF NOT DONE YET
5212 016456 032737 100000 005670 BIT          #BIT15,PT  ;SEE IF PATTERN IS SPECIFIED
5213 016464 001005          BNE          12$         ;BR IF PATTERN SPECIFIED
5214 016466 042777 000100 162450 BIC          #BIT6,STKS ;DISABLE KBD INTERRUPT
5215 016474 000137 016124  JMP          ASKMDE     ;JUMP TO ASK FOR NEW MODE
5216          ;TYPE OUT USER-DEFINED PATTERN 15
5217 016500 104400 010662 12$: TYPE          PFIFTN ;TYPE "USER-DEFINED PATTERN 15:"
5218 016504 005001          CLR          R1          ;INITIALIZE WORD INDEX
5219 016506 005737 005524 14$: TST          INTCHR     ;SEE IF ANY INPUT
5220 016512 001420          BEQ          20$         ;BR IF NO INPUT
5221 016514 122737 000003 005524 CMPB         #003,INTCHR ;SEE IF (↑C) TYPED
5222 016522 001002          BNE          16$         ;BR IF NOT (↑C)
5223 016524 000137 014520  JMP          DRVTST     ;JUMP TO ASK FOR DRIVE(S) AGAIN
5224 016530 122737 000032 005524 16$: CMPB         #032,INTCHR ;SEE IF (↑Z) TYPED
5225 016536 001002          BNE          18$         ;BR IF NOT (↑Z)
5226 016540 000137 016124  JMP          ASKMDE     ;JUMP TO ASK FOR NEW MODE
5227 016544 004737 027270 18$: JSR          PC,ECOBAD ;ECHO BAD INPUT
5228 016550 004737 027250  JSR          PC,PREPKB  ;PREPARE FOR POSSIBLE KBD INPUT
5229 016554 004737 032642 20$: JSR          PC,TYPPAT ;TYPE WORD XX = YYYYYY
5230 016560 104400 001315  TYPE          $CRLF      ;TYPE <CR>, <LF>
5231 016564 062701 000002  ADD          #2,R1      ;INCREMENT INDEX

```

```

5232 016570 022701 000040          CMP      #32.,R1          ;SEE IF 16 WORDS TYPED
5233 016574 001344          BNE      14$           ;BR IF NOT DONE YET
5234 016576 042777 000100 162340      BIC      #BIT6,STKS    ;DISABLE KBD INTERRUPT
5235 016604 000137 016124          JMP      ASKMDE        ;JUMP TO ASK FOR NEW MODE
5236
5237
5238
5239
5240          .SBTTL OPEN (0) LIST ROUTINE
5241          ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
5242          ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
5243          ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
5244          ;*TTY INPUT. TYPING (↑C) CAUSES IMMEDIATE RETURN TO
5245          ;*DRVTST, AND (↑Z) CAUSES RETURN TO ASKMDE.
5246
5247 016610          OPNLST:
5248 016610 104400 010600          TYPE     DFQUES        ;ASK IF ALL DEFAULT VALUES DESIRED
5249 016614 004737 032214          JSR      PC,RDCHRS    ;READ RESPONSE TO DEFAULT QUESTION
5250 016620 014520          DRVTST
5251 016622 016124          ASKMDE        ;(↑C) RETURN ADDRESS
5252 016624 016610          OPNLST        ;(↑Z) RETURN ADDRESS
5253 016626 005700          TST      R0          ;(↑U) OR ERROR RETURN ADDRESS
5254 016630 001433          BEQ      NOTDFT      ;SEE IF NULL INPUT
5255 016632 022737 000104 005264      CMP      #'D,BUFFO    ;BR IF DEFAULTS NOT REQUESTED
5256 016640 001405          BEQ      LODFPT      ;SEE IF (D) TYPED
5257 016642 104400 005264          TYPE     ,BUFFO      ;BR IF DEFAULTS DESIRED
5258 016646 104400 001314          TYPE     ,SQUES      ;ECHO BAD INPUT
5259 016652 000756          BR        OPNLST      ;GO ASK AGAIN
5260          ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
5261 016654 012700 005676      LODFPT: MOV     #PRDFLT,R0 ;LOAD DEFAULT LIST ADDRESS
5262 016660 012702 005650          MOV     #PRMLST,R2   ;LOAD PARAMETER LIST ADDRESS
5263 016664 012022          4$:  MOV     (R0)+,(R2)+ ;LOAD A DEFAULT VALUE
5264 016666 022702 005676          CMP     #PRDFLT,R2   ;SEE IF DONE YET
5265 016672 001374          BNE     4$           ;BR IF MORE VALUES TO LOAD YET
5266 016674 105737 003106          TSTB   MDFLAG        ;SEE IF DEFAULT MODE
5267 016700 001005          BNE     6$           ;BR IF NOT DEFAULT MODE
5268 016702 012737 000001 023506      MOV     #1,SEOPCT    ;SET PASS COUNT = 1
5269 016710 000137 017412          JMP     STPASS        ;GO RUN DFLT TESTS
5270 016714 000137 016124          6$:  JMP     ASKMDE        ;JUMP TO ASK FOR NEW MODE
5271 016720 005001          NOTDFT: CLR    R1     ;INITIALIZE INDEX
5272          ;TYPE CURRENT PARAMETER AND VALUE
5273 016722 004737 032562          8$:  JSR     PC,TYPRM    ;TYPE PARAMETER AND VALUE
5274 016726 104400 013422          TYPE   ,SPACE1      ;TYPE A SPACE
5275 016732 104400 013442          TYPE   ,PROMPT      ;TYPE ASTERISK AND SPACE
5276          ;READ AND CHECK INPUT, IF ANY
5277 016736 004737 032214          JSR     PC,RDCHRS    ;READ OCTAL DIGITS TYPED
5278 016742 014520          DRVTST
5279 016744 016124          ASKMDE        ;(↑C) RETURN ADDRESS FOR RDCHRS
5280 016746 016722          8$          ;(↑Z) RETURN ADDRESS FOR RDCHRS
5281 016750 005700          TST     R0          ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
5282 016752 001007          BNE     10$         ;SEE IF ANY INPUT
5283 016754 022761 040515 006000      CMP     #'MA,PRMNE(R1) ;BR IF ANY INPUT
5284 016762 001023          BNE     12$         ;SEE IF (MA) JUST DEFAULTED
5285 016764 062701 000002          ADD     #2,R1        ;BR IF NOT (MA)
5286 016770 000420          BR      12$         ;INCREMENT INDEX
5287 016772 022700 000010          10$:  CMP     #10,R0      ;BR TO MOVE ON TO NEXT PARAMETER
                    ;SEE IF 8 CHARACTERS TYPED
    
```

```

5288 016776 002431          BLT      14$          ;BR IF MORE THAN 8 TYPED
5289 017000 012746 005264    MOV      #BUFFO,-(SP) ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
5290 017004 004737 052736    JSR      PC,OCTBIN    ;CHECK DIGITS AND CONVERT TO BINARY
5291 017010 017062          14$          ;ERROR RETURN ADDRESS FOR OCTBIN
5292 017012 012637 005470    MOV      (SP)+,LOWOCT ;GET LOW BINARY BITS
5293 017016 013737 053070 005472  MOV      $HIOCT,HIGOCT ;GET HIGH BINARY BITS
5294          ;CHECK PARAMETER VALUE AND PUT INTO LIST
5295 017024 004737 033220    JSR      PC,CHKPRM    ;CHECK VALIDITY OF PARAM VALUE
5296 017030 017062          14$          ;ERROR RETURN ADDR. FOR CHKPRM
5297          ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5298 017032 004737 032674    12$: JSR      PC,MODP15
5299          ;MOVE ON TO NEXT PARAMETER
5300 017036 062701 000002    ADD      #2,R1        ;INCREMENT THE PARAMETER INDEX
5301 017042 010102          MOV      R1,R2        ;GET COPY OF INDEX
5302 017044 062702 005650    ADD      #PRMLST,R2   ;COMPUTE POSITION IN LIST
5303 017050 022702 005676    CMP      #PRDFLT,R2   ;SEE IF DONE WITH LIST
5304 017054 001322          BNE      $$          ;BR IF NOT DONE YET
5305 017056 000137 016124    JMP      ASKMDE       ;JUMP TO ASK FOR NEW MODE
5306 017062 104400 005264    14$: TYPE    ,BUFFO    ;ECHO BAD INPUT
5307 017066 104400 001314    TYPE    ,SQUES       ;TYPE (?) AND (CR), (LF)
5308 017072 000713          BR       B$          ;BR TO ASK AGAIN
5309
5310
5311
5312          .SBTTL SET (S) INDIVIDUAL PARAM ROUTINE
5313          ;*THIS ROUTINE IS USED TO ALLOW SETTING OF ANY INDIVIDUAL
5314          ;*PARAMETER VALUE BY TTY INPUT. A PROMPTER (>) IS
5315          ;*TYPED AND THE ROUTINE WAITS FOR INPUT OF THE FORM
5316          ;*XX=YYYYYY, WHERE XX IS THE MNEMONIC FOR ANY VALID
5317          ;*PARAMETER IN THE LIST, AND YYYYYY IS THE DESIRED PARAMETER
5318          ;*VALUE IN OCTAL DIGITS. THE PROGRAM THEN ASKS FOR ANOTHER
5319          ;*PARAMETER BY TYPING ">" AGAIN. TYPING (↑) CAUSES
5320          ;*IMMEDIATE RETURN TO DRVTST, AND (↑Z) CAUSES RETURN TO
5321          ;*ASKMDE.
5322
5323          SETPRM:
5324 017074 104400 013445    TYPE    PRMPSP       ;TYPE ">" PROMPTER
5325          ;READ AND CHECK INPUT, IF ANY
5326 017100 004737 032214    JSR      PC,RDCHRS   ;READ INPUT LINE
5327 017104 014520          DRVTST              ;(↑) RETURN ADDRESS FOR RDCHRS
5328 017106 016124          ASKMDE              ;(↑Z) RETURN ADDRESS FOR RDCHRS
5329 017110 017074          SETPRM              ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
5330 017112 020027 000004    CMP      R0,#4       ;SEE IF AT LEAST 4 CHARACTERS TYPED
5331 017116 002005          BGE      6$         ;BR IF AT LEAST 4 TYPED
5332 017120 104400 005264    4$: TYPE    ,BUFFO    ;ECHO BAD INPUT
5333 017124 104400 001314    TYPE    ,SQUES       ;TYPE (?) AND (CR), (LF)
5334 017130 000761          BR       SETPRM      ;BR TO ASK FOR INPUT AGAIN
5335 017132 020027 000013    6$: CMP      R0,#11.  ;SEE IF ELEVEN OR LESS CHARS TYPED
5336 017136 003370          BGT      4$         ;BR IF MORE THAN ELEVEN TYPED
5337          ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
5338 017140 005001          CLR      R1          ;INITIALIZE PARAMETER INDEX
5339 017142 023761 005264 006000 8$: CMP      BUFFO,PRMNEM(R1) ;TRY TO MATCH 2-CHAR MNEMONIC
5340 017150 001411          BEQ      10$        ;BR IF PARAMETER FOUND IN TABLE
5341 017152 062701 000002    ADD      #2,R1        ;INCREMENT INDEX
5342 017156 010102          MOV      R1,R2        ;GET COPY OF INDEX
5343 017160 062702 005650    ADD      #PRMLST,R2   ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
    
```



```

5344 017164 022702 005676      CMP      #PRDFLT,R2
5345 017170 001364              BNE      8$                ;BR IF MORE TO CHECK YET
5346 017172 000752              BR       4$                ;BR TO ECHO BAD INPUT
5347                                ;CHECK FOR VALID LINE FORMAT
5348 017174 012702 000002      10$:    MOV      #2,R2          ;INITIALIZE BUFFER POINTER
5349 017200 122762 000075 005264    CMPB    #'=,BUFFO(R2)      ;SEE IF NEXT CHAR IS "="
5350 017206 001411              BEQ      12$              ;BR IF IT IS "="
5351 017210 122762 000040 005264    CMPB    #040,BUFFO(R2)    ;SEE IF NEXT CHAR IS A SPACE
5352 017216 001340              BNE      4$                ;BR TO ECHO BAD INPUT
5353 017220 005202              INC      R2                ;INCREMENT BUFFER POINTER
5354 017222 122762 000075 005264    CMPB    #'=,BUFFO(R2)      ;SEE IF NEXT CHAR IS "="
5355 017230 001333              BNE      4$                ;BR TO ECHO INVALID INPUT
5356 017232 005202              12$:    INC      R2                ;INCREMENT BUFFER POINTER
5357 017234 020200              CMP      R2,RO            ;SEE IF MORE CHARS LEFT IN BUFFER
5358 017236 002330              BGE      4$                ;BR TO ECHO INVALID INPUT
5359 017240 122762 000040 005264    CMPB    #040,BUFFO(R2)    ;SEE IF NEXT CHARACTER IS SPACE
5360 017246 001003              BNE      14$              ;BR IF NOT A SPACE
5361 017250 005202              INC      R2                ;INCREMENT BUFFER POINTER
5362 017252 020200              CMP      R2,RO            ;SEE IF MORE CHARS LEFT IN BUFFER
5363 017254 002321              BGE      4$                ;BR IF NONE LEFT
5364 017256 160200              14$:    SUB      R2,RO            ;SUBTRACT POINTER FROM CHAR COUNT
5365 017260 020027 000010    CMP      RO,#10          ;SEE HOW MANY CHARS LEFT
5366 017264 003315              BGT      4$                ;BR IF TOO MANY LEFT
5367                                ;CHECK FOR LEGAL PARAMETER VALUE
5368 017266 062702 005264      ADD      #BUFFO,R2        ;GET ADDRESS OF DIGITS
5369 017272 010246              MOV      R2,-(SP)         ;PUT IT ON STACK FOR OCTBIN
5370 017274 004737 052736      JSR      PC,OCTBIN        ;CHECK DIGITS AND CONVERT TO BINARY
5371 017300 017120              4$                                ;ERROR RETURN ADDR. FOR OCTBIN
5372 017302 012637 005470      MOV      (SP)+,LOWOCT     ;GET LOW BINARY BITS
5373 017306 013737 053070 005472    MOV      $HI OCT,HIGOCT  ;GET HIGH BINARY BITS
5374 017314 004737 033220      JSR      PC,CHKPRM        ;CHECK VALIDITY OF PARAMETER VALUE
5375 017320 017120              4$                                ;ERROR RETURN ADDR. FOR CHKPRM
5376                                ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5377 017322 004737 032674      JSR      PC,MODP15
5378 017326 000662              BR       SETPRM          ;RETURN TO ASK FOR ANOTHER PARAMETER
5379
5380
5381
5382                                .SBTTL  RUN (R) TESTS ROUTINE
5383                                ;*THIS ROUTINE IS ENTERED TO SET UP THE RUNNING OF TESTS,
5384                                ;*WHOSE PARAMETERS HAVE BEEN DETERMINED. IT FIRST ASKS
5385                                ;*FOR THE DESIRED NUMBER OF PROGRAM PASSES, AND STORES
5386                                ;*THIS IN $EOPCT FOR THE END OF PASS ROUTINE. THEN, THE
5387                                ;*DRIVE LIST IS SCANNED FOR THE FIRST DESIRED DRIVE, AND
5388                                ;*THE HEADER ON SECTOR 0 IS READ TO GET THE FORMAT FOR THIS
5389                                ;*DRIVE. THIS IS STORED IN THE FORMAT BYTE, AND A JUMP IS
5390                                ;*MADE TO THE FIRST TEST.
5391                                ;*THE END OF PASS ROUTINE RETURNS TO "NEWDRV" TO SELECT
5392                                ;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
5393                                ;* (ALL DRIVES) IS COMPLETED.
5394
5395 017330                                RUNTST:
5396                                ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
5397 017330 104400 011031      4$:    TYPE      ENTPAS      ;ASK FOR NUMBER OF PASSES
5398 017334 004737 032214      JSR      PC,RDCHRS      ;READ RESPONSE
5399 017340 014520              DRVTST                    ;(+C) RETURN ADDRESS
    
```


U456	017672	113737	005502	011163	MOV	DRIVE,DRVNO	:GET DRIVE NO.
U457	017700	152737	000060	011163	BIS	#'D,DRVNO	:CONVERT TO ASCII
U458	017706	104400	011142		TYPE	.TSTDRN	:TYPE "TESTING DRIVE X"
U459	017712	005737	001326		TST	\$PASS	:SEE IF THIS IS FIRST PASS
U460	017716	001012			BNE	10\$:BR IF NOT FIRST PASS
U461	017720	004737	033524		JSR	PC,DRVSER	:TYPE "DRIVE SER. NO. XXX"
U462	017724	004737	033644		JSR	PC,CRTSER	:TYPE "CART. SER. NO. XXXXXXXXXXXX"
U463	017730	032737	000020	005672	BIT	#BIT4,CS	:SEE IF BAD SECTORS SHOULD BE TYPED
U464	017736	001402			BEQ	10\$:BR IF NOT
U465	017740	004737	037564		JSR	PC,TYPBSF	:TYPE BAD SECTOR FILES
U466	017744	005037	005524		CLR	INTCHR	:INIT. TTY INPUT CHAR BUFFER
U467							
U468							
U469							

M10

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
5525

```
*****
*TEST 1      OFFSET-TO-FAILURE MEASUREMENTS
*IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL
*WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC.
*THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH
* 167230(OCT) (TO ESTABLISH A KNOWN PATTERN) , BUT THEY ARE NOT TESTED.
*THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE
*OFFSETS, UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE
*OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE
*CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR
*MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS
*ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER : SECTORS FS AND LS ON
*CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR
*TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF
*INCREASING TRACKS, AS FOLLOWS :
*
* OFFSET-TO-FAILURE MEASUREMENTS :
*
* TRACK  CYLN  SECT  +OFST  -OFST
*          (UIN)  (UIN)
*   X     XXX   XX   XXXX   XXXX
*   X     XXX   XX   XXXX   XXXX
*   X     XXX   XX   XXXX   XXXX
*
*          ETC.
*
*NOTE: IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS
*TO 10(DEC), FOR THIS TEST.
*****
```

```
017750 000004
017752 012737 000001 001324
017760 004737 032060
017764 004737 032126
017770 000137 021060
017774 004737 030726
020000 004737 027250
020004 105737 003106
020010 001005
020012 005037 005510
020016 012737 000012 005512
020024 013765 005650 000002 3$:
020032 113765 005510 000004
020040 105065 000005
020044 013737 005652 005506
020052 022737 000632 005652
020060 001002
020062 005337 005506
020066 012703 072307 4$:
020072 004737 035772
020076 012703 167230
020102 005365 000002
020106 100402
020110 004737 035772
020114 062765 000002 6$:
020122 022765 000632 000002
```

```
TST1: SCOPE
      MOV #1,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
      JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
      JSR PC,CHKITR ;SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP TST2 ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
      JSR PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
      JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
;INITIALIZE PARAMETERS, WRITE THE SECTORS, AND WRITE CHECK THEM
      TSTB MDFLAG ;SEE IF ADRS 200 DEFAULT RUN
      BNE 3$ ;BR IF NOT
      CLR FS ;SET FS = 0
      MOV #10.,LS ;SET LS = 10(DEC)
3$:   MOV FC,P.CYLN(R5) ;SET CYL = FC
      MOVB FS,P.SECT(R5) ;SET SECTOR = FS
      CLRB P.TRCK(R5) ;SET TRACK = 0
      MOV LC,CYLNR
      CMP #632,LC ;SEE IF LC =632
      BNE 4$ ;BR IF NOT 632
      DEC CYLNR ;MAKE IT 631 (PRESERVE BSF)
4$:   MOV #72307,R3 ;GET DATA PATTERN
      JSR PC,WRTSEC ;DO WRITE AND WRITE CHECK
      MOV #167230,R3 ;DATA FOR AJACENT CYLS
      DEC P.CYLN(R5) ;NEXT LOWER CYL
      BMI 6$ ;BR IF CYL NEGATIVE
      JSR PC,WRTSEC ;DO WRITE AND WRITE CHECK
6$:   ADD #2,P.CYLN(R5) ;NEXT HIGHER CYL
      CMP #632,P.CYLN(R5) ;SEE IF CYL = 632
```

```

5526 020130 001004          BNE      8$          ;BR IF NOT
5527 020132 122765 000002 000005  CMPB    #2,P.TRCK(R5) ;SEE IF BSF
5528 020140 001402          BEQ     10$          ;BR IF YES, TO PROTECT IT
5529 020142 004737 035772          JSR     PC,WRTSEC   ;DO WRITE AND WRITE CHECK
5530 020146 005365 000002 10$: DEC     P.CYLN(R5) ;RESTORE CYL NO.
5531 020152 105265 000005          INCB   P.TRCK(R5)   ;INCR TRACK
5532 020156 122765 000003 000005  CMPB    #3,P.TRCK(R5) ;TRACK = 3 YET ?
5533 020164 001340          BNE     4$          ;BR IF NOT YET
5534 020166 126537 000004 005512  CMPB    P.SECT(R5),LS ;SEE IF SECTOR = LS
5535 020174 001406          BEQ     14$          ;BR IF YES
5536 020176 113765 005512 000004  MOVB   LS,P.SECT(R5) ;SET SECTOR = LS
5537 020204 105065 000005 12$: CLRB  P.TRCK(R5)   ;SET TRACK = 0
5538 020210 000726          BR      4$          ;BR TO CONTINUE WRITING
5539 020212 026537 000002 005506 14$: CMP     P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
5540 020220 001407          BEQ     16$          ;BR IF YES
5541 020222 013765 005506 000002  MOV     CYLNDR,P.CYLN(R5) ;SET CYL = LC
5542 020230 113765 005510 000004  MOVB   FS,P.SECT(R5) ;SET SECTOR = FS AGAIN
5543 020236 000762          BR      12$         ;BR TO CONTINUE WRITING
5544          ;INITIALIZE PARAMETERS FOR OFFSET MEASUREMENTS
5545 020240 032737 000002 005672 16$: BIT     #BIT1,CS   ;SEE IF REPORTS ARE INHIBITED
5546 020246 001002          BNE     18$          ;BR IF INHIBITED
5547 020250 104400 011274          TYPE   OFSHED      ;TYPE HEADINGS
5548 020254 113765 005654 000005 18$: MOVB   FT,P.TRCK(R5) ;SET TRACK = FT
5549 020262 013765 005650 000002  MOV     FC,P.CYLN(R5) ;SET CYLINDER = FC
5550 020270 113765 005510 000004  MOVB   FS,P.SECT(R5) ;SET SECTOR = FS
5551          ;LOAD THE R/W BUFFER FOR WRITE CHECKS
5552 020276 012700 064574          MOV     #RWBUR,RO   ;SET BUFFER ADDRESS
5553 020302 012701 000400          MOV     #400,R1     ;PREPARE TO LOAD 400(OCT) WORDS
5554 020306 012720 072307 19$: MOV     #072307,(RO)+ ;LOAD A WORD INTO BUFFER
5555 020312 005301          DEC     R1          ;DECR COUNTER
5556 020314 001374          BNE     19$         ;BR IF NOT DONE YET
5557 020316 005004 20$: CLR     R4          ;INIT (+) OFFSET VALUE
5558 020320 005000          CLR     RO          ;INIT OFFSET NUMBER TO 0
5559 020322 012737 177777 005546  MOV     #-1,PLOFST  ;INITIALIZE CURRENT FAILING (+) OFFSET
5560 020330 012737 177777 005550  MOV     #-1,NGOFST ;INITIALIZE CURRENT FAILING (-) OFFSET
5561          ;PERFORM OFFSETS AND WRITE CHECKS
5562 020336 112765 000117 000001 22$: MOVB   #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5563 020344 004737 040716          JSR     PC,DRVCAL   ;PERFORM SEEK
5564 020350 110065 000006          MOVB   RO,P.OFST(R5) ;SET CURRENT OFFSET
5565 020354 112765 000115 000001  MOVB   #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
5566 020362 004737 040716          JSR     PC,DRVCAL   ;PERFORM OFFSET
5567 020366 112765 000131 000001  MOVB   #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
5568 020374 012737 021024 003036  MOV     #WRCKHD,A.ABNL ;SET SPECIAL ERROR HANDLER ADDRESS
5569 020402 105037 003131          CLRB   WCEFLG      ;CLEAR WRITE CHECK ERROR FLAG
5570 020406 004737 040716          JSR     PC,DRVCAL   ;PERFORM A WRITE CHECK
5571 020412 012737 042410 003036  MOV     #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADDRESS
5572 020420 112765 000177 000001  MOVB   #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR CMND
5573 020426 004737 040716          JSR     PC,DRVCAL   ;CLEAR THE SUBSYSTEM
5574 020432 105737 003131          TSTB   WCEFLG      ;SEE IF A WRITE CHECK ERROR OCCURRED
5575 020436 001020          BNE     26$         ;BR IF AN ERROR OCCURRED
5576 020440 010001          MOV     RO,R1       ;GET A COPY OF CURRENT OFFSET
5577 020442 005200          INC     RO          ;INCREMENT OFFSET
5578 020444 062704 000031          ADD     #25,R4      ;INCR OFFSET VALUE BY 25(DEC) UIN
5579 020450 042701 177700          BIC     #177700,R1  ;MASK FOR MAGNITUDE BITS
5580 020454 022701 000060          CMP     #60,R1      ;SEE IF MAX OFFSET APPLIED IN THIS DIRECTION
5581 020460 001326          BNE     22$         ;BR IF NOT YET
    
```

```

5593 020462 032700 000200 BIT #BIT7,RO ;SEE IF NEG OFFSETS DONE YET
5594 020466 001015 BNE 32$ ;BR IF YES, TO PRINT CURRENT SECTOR RESULT
5595 020470 012700 000200 24$: MOV #200,RO ;INIT FOR NEG OFFSETS
5596 020474 005004 CLR R4 ;INIT (-) OFFSET VALUE TO 0
5597 020476 000717 BR 22$ ;BR TO APPLY NEG OFFSETS
5598 020500 032700 000200 26$: BIT #BIT7,RO ;SEE IF NEG OFFSETS DONE YET
5599 020504 001403 SEC 28$ ;BR IF NOT YET
5600 020506 010437 005550 MOV R4,NGOFST ;SAVE THIS FAILING (-) OFFSET VALUE
5601 020512 000403 BR 32$ ;BR TO PRINT CURRENT SECTOR RESULT
5602 020514 010437 005546 28$: MOV R4,PLOFST ;SAVE THIS FAILING (+) OFFSET VALUE
5603 020520 000763 BR 24$ ;BR TO APPLY NEGATIVE OFFSETS
5604 020522 032737 000002 005672 32$: BIT #BIT1,CS ;SEE IF REPORTS INHIBITED
5605 020530 001051 SNE 43$ ;BR IF INHIBITED
;TYPE OFFSET RESULTS FOR THIS SECTOR
5606 020532 116501 000005 MOVB P.TRCK(R5),R1 ;GET TRACK NO.
5607 020536 010146 MOV R1,-(SP) ;PUT IT ON STACK
5608 020540 104402 TYPOS ;TYPE IT
5609 020542 003 .BYTE 3 ;3 DIGITS
5610 020543 000 .BYTE 0 ;SUPPRESS
5611 020544 104400 013420 TYPE SPACE3
5612 020550 016546 000002 MOV P.CYLN(R5),-(SP) ;PUT CYL ON STACK
5613 020554 104402 TYPOS ;TYPE IT
5614 020556 004 .BYTE 4 ;4 DIGITS
5615 020557 000 .BYTE 0 ;SUPPRESS
5616 020560 104400 013421 TYPE SPACE2
5617 020564 116501 000004 MOVB P.SECT(R5),R1 ;GET SECTOR
5618 020570 010146 MOV R1,-(SP) ;PUT IT ON STACK
5619 020572 104402 TYPON ;TYPE SECTOR
5620 020574 104400 013422 TYPE SPACE1
5621 020600 005737 005546 TST PLOFST ;SEE IF THERE WAS A FAILING (+) OFFSET
5622 020604 100003 BPL 36$ ;BR IF THERE WAS
5623 020606 104400 011265 TYPE NOFALS ;TYPE "NONE"
5624 020612 000403 BR 38$ ;CONTINUE
5625 020614 013746 005546 36$: MOV PLOFST,-(SP) ;PUT (+) OFFSET VALUE ON STACK
5626 020620 104404 TYPDS ;TYPE IT IN DECIMAL
5627 020622 104400 013422 38$: TYPE SPACE1
5628 020626 005737 005550 TST NGOFST ;SEE IF THERE WAS A FAILING (-) OFFSET
5629 020632 100003 BPL 40$ ;BR IF THERE WAS
5630 020634 104400 011265 TYPE NOFALS ;TYPE "NONE"
5631 020640 000403 BR 42$ ;CONTINUE
5632 020642 013746 005550 40$: MOV NGOFST,-(SP) ;PUT (-) OFFSET VALUE ON STACK
5633 020646 104404 TYPDS ;TYPE IT IN DECIMAL
5634 020650 104400 001315 42$: TYPE $CRLF ;TYPE <CR>,<LF>
;SET UP TO TEST NEXT SECTOR
5635 020654 126537 000004 005512 43$: CMPB P.SECT(R5),LS ;SEE IF SECTOR = LS
5636 020662 001404 BEQ 44$ ;BR IF YES
5637 020664 113765 005512 000004 MOVB LS,P.SECT(R5) ;SET SECTOR = LS
5638 020672 000611 BR 20$ ;GO TEST THIS SECTOR
5639 020674 026537 000002 005506 44$: CMP P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
5640 020702 001410 BEQ 48$ ;BR IF YES
5641 020704 013765 005506 000002 MOV CYLNDR,P.CYLN(R5) ;SET CYL = LC
5642 020712 113765 005510 000004 46$: MOVB FS,P.SECT(R5) ;SET SECTOR = FS
5643 020720 000137 020316 JMP 20$ ;GO TEST THIS SECTOR
5644 020724 105265 000005 48$: INCB P.TRCK(R5) ;INCREMENT TRACK
5645 020730 122765 000003 000005 CMPB #3,P.TRCK(R5) ;SEE IF TRACK = 3
5646 020736 001404 BEQ 50$ ;BR IF YES

```

```

5638 020740 013765 005650 000002      MOV      FC,P.CYLN(R5)      ;SET CYL = FC
5639 020746 000761 000000 000000      BR      45$                ;GO TEST THIS SECTOR
5640 020750 112765 000113 000001 50$:  MOVB   #RECAL,P.CMND(P5)   ;SET RECAL COMMAND
5641 020756 004737 040716 000000      JSR     PC,DRV'CAL        ;DO CLEANUP RECALIBRATE
5642 020762 013737 005664 005510      MOV     S2,FS              ;RESTORE FS,LS FOR 22 SECTORS
5643 020770 013737 005666 005512      MOV     S3,LS
5644 020776 105737 003115 000000      TSTB   FORMAT             ;SEE IF 22 SECTORS
5645 021002 001406 000000 000000      BEQ     54$                ;BR IF YES
5646 021004 013737 005660 005510      MOV     S0,FS              ;RESTORE FS,LS FOR 20 SECTORS
5647 021012 013737 005662 005512      MOV     S1,LS
5648 021020 000137 021060 000000 54$:  JMP     TST2                ;JUMP TO NEXT TEST
5649 021020
5650
5651
5652

```

```

; *THIS IS THE ABNORMAL RETURN FROM THE DRIVER, USED WHEN A
; * "DATA TYPE" ERROR IS EXPECTED (AND VALID).
5653 021024 032765 040000 000020 WRCKHD: BIT  #WCE,P.CS2(R5) ;SEE IF WRITE CHECK ERROR OCCURRED
5654 021032 001006 000000 000000      BNE     4$                 ;BR IF WCE
; CHECK FOR OTHER "DATA TYPE" ERROR
5656 021034 032765 130400 000034      BIT     #HVRC!DTE!OPI!DCK,P.ER(R5)
5657 021042 001002 000000 000000      BNE     4$                 ;BR IF A DATA ERROR OCCURRED
5658 021044 000137 042410 000000      JMP     ERRHDL              ;GO HANDLE OTHER ERROR
5659 021050 105237 003131 000000 4$:  INCB   WCEFLG              ;SET WRITE CHECK ERROR FLAG
5660 021054 000137 044624 000000      JMP     RETNML              ;TAKE NORMAL RETURN
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683

```

```

; *****
; *TEST 2      NPR/MEMORY WORD ADDRESSING TEST
; *IN THIS TEST, MEMORY WORD ADDRESSING CAPABILITY DURING RK06 NPP'S,
; *IS TESTED.
; *BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
; *THEN, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK
; *BEYOND ADDRESS RWBUF+6000(OCTAL), WRITE UNIQUE NUMBERS
; *(STARTING WITH 1), INTO EACH OF UP TO 64K WORDS (DEPENDING UPON THE
; *AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES
; *WITHIN THE 64K BLOCK. NEXT, WRITE THE 64K WORDS (MAX) ONTO THE DISK
; *AT ADDRESS FC,FT,FS (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW
; *OR DESTRUCTION OF THE BAD SECTOR FILE). THEN ZERO THE ENTIRE BLOCK
; *IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL
; *ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING
; *VIRTUAL (IF MEM. MGT.) AS WELL AS PHYSICAL ADDRESSES, AND THE GOOD
; *AND BAD DATA, FOR UP TO THE FIRST 10(DEC) FAILING LOCATIONS.
; * IF MEMORY MANAGEMENT IS INSTALLED AND THERE IS ADDITIONAL MEMORY TO
; *EXERCISE, REPEAT THE ABOVE WORD ADDRESSING TEST, FOR EACH OF THE
; *REMAINING 64K MEMORY BLOCKS.
; *****

```

```

5684 021060 000004 000000 000000 001324 TST2:  SCOPE
5685 021062 012737 000002 000000      MOV     #2,STESTN          ;:SET TEST NUMBER IN APT MAIL BOX
5686 021070 004737 032060 000000      JSR     PC,SETUP           ;:SET UP FOR LOOP ON ERROR
5687 021074 004737 032126 000000      JSR     PC,CHKITR          ;:SEE IF ITER. NO. = 0 FOR THIS TEST
5688 021100 000137 021370 000000      JMP     TST3                ;:JUMP TO NEXT TEST
5689 021104 004737 030726 000000 ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5690 021110 004737 027250 000000      JSR     PC,INITSS          ;:INITIALIZE DRIVER PARAMS AND SUB-SYS
5691 021114 004737 030422 000000      JSR     PC,PREPKB          ;:PREPARE FOR POSSIBLE KBD INPUT
5692
5693

```

```

;SAVE XXDP LOADER, IF PRESENT
;COMPUTE STARTING ADR OF XXDP

```

```

5694 021120 012737 064574 005542      MOV      #RWBUFF,XDPSAV      ;SET XXDP SAVE AREA ADR
5695 021126 005037 065544      CLR      XDPSAV+2
5696 021132 004737 030474      JSR      PC,SAVXDP          ;GO SAVE XXDP LOADER
5697 021136 112737 000001 003124      MOVVB   #1,XOVLAD          ;SET INDICATORS
5698 021144 112737 000001 003125      MOVVB   #1,XDPSVD
5699 021152 012737 177777 005536      MOV      #-1,PATRN          ;SET FLAG FOR SVPRMS
5700                                     ;INIT PARAMETERS, BEGIN TESTING
5701 021160 004737 037022      JSR      PC,SETUPM          ;INIT PARAMS
5702 021164 004737 033450      4$:     JSR      PC,MXWRDC      ;COMPUTE MAX WORD COUNT FOR THIS MA
5703 021170 005701 000000      TST      R1                ;SEE IF ALL MEMORY TESTED YET
5704 021172 100467 000000      BMI     50$                ;BR IF ALL DONE
5705 021174 005065 000012      CLR      P.WC(R5)          ;INIT WORD COUNT TO 65,536(DEC)
5706 021200 005701 000000      TST      R1                ;SEE IF WRD CNT SHOULD BE 65,536
5707 021202 001003 000000      BNE     6$                 ;BR IF YES
5708 021204 005400 000000      NEG      R0
5709 021206 010065 000012      MOV      R0,P.WC(R5)       ;SET WORD COUNT
5710 021212 013765 005600 000010 6$:     MOV      MA,P.BALO(R5)     ;SET BA BITS 0-15
5711 021220 013701 005602      MOV      MA+2,R1           ;GET MA BITS 16-21
5712 021224 042701 177774      BIC     #177774,R1        ;MASK FOR LO 2 BITS
5713 021230 150165 000007      BISB    R1,P.BAHI(R5)     ;SET BA BITS 16,17
5714 021234 005737 056216      TST      $K11             ;SEE IF MEM MGT PRESENT
5715 021240 100010 000000      BPL     14$                ;BR IF NOT
5716                                     ;SET UP MEM MGT
5717 021242 013737 005600 005604      MOV      MA,PMA            ;SET BUFFER ADDRESS
5718 021250 013737 005602 005606      MOV      MA+2,PMA+2
5719 021256 004737 030300      JSR      PC,PAPRAR        ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5720                                     ;PERFORM THE TESTS
5721 021262 012737 021262 001110 14$:   MOV      #,$LPERA        ;SET NEW LOOP ON ERROR ADRS
5722 021270 004737 032060      JSR      PC,SETUP          ;SET UP FOR LOOP ON ERROR
5723 021274 004737 040226      JSR      PC,SVPRMS        ;SAVE PARAMS FOR THIS XFER
5724 021300 004737 036066      JSR      PC,LDMEM1        ;LOAD THIS MEM BLK WITH INCREMENTING DATA
5725 021304 112765 000123 000001      MOVVB   #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5726 021312 004737 040340      JSR      PC,TRANSFR       ;WRITE THE DATA
5727 021316 005003 000000      CLR      R3                ;SET DATA WORD = 0
5728 021320 004737 036100      JSR      PC,LDMEM2        ;LOAD MEM BLK WITH ALL ZEROS
5729 021324 112765 000121 000001      MOVVB   #RDATA,P.CMND(R5) ;SET READ COMMAND
5730 021332 004737 040340      JSR      PC,TRANSFR       ;READ THE DATA FROM DISK
5731 021336 004737 036232      JSR      PC,CKMEM1        ;PERFORM SOFTWARE COMPARE OF DATA
5732 021342 104410 000000      SCOPER                                     ;CHECK FOR INTERNAL LOOP ON ERROR
5733                                     ;GET NEW MA FOR NEXT TRANSFER
5734 021344 004737 037230      JSR      PC,INCRMA        ;COMPUTE NEXT MA
5735 021350 000705 000000      BR      4$                 ;GO SEE IF MORE MEMORY TO TEST
5736 021352 004737 030502      50$:   JSR      PC,GETXDP      ;RESTORE XXDP LOADER
5737 021356 105737 003134      TSTB   UBMPRS            ;SEE IF UNIBUS MAP PRESENT
5738 021362 001402 000000      BEQ     54$                ;BR IF NOT
5739 021364 005037 172516      CLR      #SR3             ;DISABLE UNIBUS MAP
5740 021370
5741
5742
5743
5744
5745
5746
5747
5748
5749

```

```

:*****
: *TEST 3      NPR/MEMORY BLOCK ADDRESSING TEST
: *IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06
: *NPR'S IS TESTED.
: *BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBJF.
: *MEMORY WILL BE TESTED STARTING AT THE FIRST ADRS OF THE NEXT 1K

```



```

5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761 021370 000004
5762 021372 012737 000003 001324
5763 021400 004737 032060
5764 021404 004737 032126
5765 021410 000137 021730
5766
5767 021414 004737 030726
5768 021420 004737 027250
5769
5770 021424 004737 030422
5771 021430 012737 064574 005542
5772 021436 005037 005544
5773 021442 004737 030474
5774 021446 112737 000001 003124
5775 021454 112737 000001 003125
5776 021462 012737 177777 005536
5777
5778 021470 005037 005526
5779 021474 004737 037022
5780 021500 012704 000001
5781 021504 004737 033450
5782 021510 005701
5783 021512 100473
5784 021514 005065 000012
5785 021520 005701
5786 021522 001003
5787 021524 005400
5788 021526 010065 000012
5789 021532 013765 005600 000010
5790 021540 013701 005602
5791 021544 042701 177774
5792 021550 150165 000007
5793 021554 005737 056216
5794 021560 100010
5795
5796 021562 013737 005600 005604
5797 021570 013737 005602 005606
5798 021576 004737 030300
5799 021602 005737 005527
5800 021606 001026
5801
5802 021610 004737 040226
5803 021614 010403
5804 021616 004737 036100
5805 021622 112765 000123 000001

```

```

; *BLOCK BEYOND RWBUF+6000(OCTAL).
; *THE TEST BEGINS BY WRITING A SINGLE NUMBER INTO ALL LOCATIONS OF
; *EACH MEMORY BLOCK OF UP TO 64K WORDS. THIS NUMBER UNIQUELY DEFINES
; *EACH BLOCK. THEN WRITE EACH BLOCK ONTO DISK AT ADDRESS FC FS FT
; *(SCALED, IF NECESSARY TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
; *SECTOR FILE). ZERO THE BLOCK IN MEMORY, AND THEN READ THE DATA
; *BACK INTO THE PROPER PHYSICAL ADDRESSES. DO THIS FOR ALL 64K BLOCKS,
; *AND THEN COMPARE THE CONTENTS OF ALL THE MEMORY TO EXPECTED VALUES.
; *TYPE FAILURES AS DESCRIBED ABOVE FOR THE FIRST 10(DEC) FAILING
; *LOCATIONS IN EACH 64K MEMORY BLOCK.
; *****
TST3: SCOPE
      MOV     #3,$TESTN      ;; SET TEST NUMBER IN APT MAIL BOX
      JSR    PC,SETUP        ;; SET UP FOR LOOP ON ERROR
      JSR    PC,CHKITR       ;; SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP    TST4            ;; JUMP TO NEXT TEST
; RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
      JSR    PC,INITSS       ;; INITIALIZE DRIVER PARAMS AND SUB-SYS
      JSR    PC,PREPKB       ;; PREPARE FOR POSSIBLE KBD INPUT
; SAVE XXDP LOADER, IF PRESENT
      JSR    PC,FNDXDP       ;; COMPUTE STARTING ADRS OF XXDP
      MOV    #RWBUF,XDPSAV   ;; SET XXDP SAVE AREA ADDRESS
      CLR    XDPSAV+2
      JSR    PC,SAVXDP       ;; GO SAVE XXDP LOADER
      MOVB  #1,XOVLAD        ;; SET INDICATORS
      MOVB  #1,XDPSVD
      MOV    #-1,PATRN       ;; SET FLAG FOR SVPRMS
; INIT PARAMETERS, WRITE AND READ BACK ALL MEMORY BLOCKS
      CLR    SELECT         ;; INIT COMPARE FLAG
2$:   JSR    PC,SETUPM       ;; INIT PARAMS
      MOV    #1,R4          ;; INIT DATA WORD
4$:   JSR    PC,MXWRDC       ;; COMPUTE MAX WRD CNT FOR THIS MA
      TST   R1              ;; SEE IF ALL MEMORY TESTED YET
      BMI  30$              ;; BR IF DONE WRITING AND READING
      CLR  P.WC(R5)         ;; INIT WRD CNT TO 65,536(DEC)
      TST  R1              ;; SEE IF WRD CNT SHOULD BE 65535
      BNE  6$              ;; BR IF YES
      NEG  R0
      MOV  R0,P.WC(R5)     ;; SET WORD COUNT
6$:   MOV  MA,P.BALO(R5)   ;; SET BA BITS 0-15
      MOV  MA+2,R1         ;; GET MA BITS 16-21
      BIC  #177774,R1     ;; MASK FOR LO 2 BITS
      BISB R1,P.BAHI(R5)  ;; SET BA BITS 16,17
      TST  $KT11          ;; SEE IF MEM MGT PRESENT
      BPL  14$            ;; BR IF NO
; SET UP MEMORY MANAGEMENT
      MOV  MA,PMA          ;; SET BUFFER ADDRESS
      MOV  MA+2,PMA+2
      JSR  PC,PREPAR       ;; PREPARE MEM MGT REG'S AND UNIBUS MAP
14$:  TST  SELECT         ;; SEE IF COMPARES SHOULD BE DONE YET
      BNE  20$            ;; BR IF YES
; WRITE AND READ THIS MEM BLK ON RK06
16$:  JSR  PC,SVPRMS       ;; SAVE PARAMS FOR THIS XFER
      MOV  R4,R3          ;; GET DATA WORD
      JSR  PC,LDMEM2      ;; LOAD MEMORY BLK WITH THIS WORD
      MOVB #WRDATA,P.CMND(R5) ;; SET WRITE COMMAND

```



```

5806 021630 004737 040340 JSR PC,TRNSFR ;WRITE THE DATA
5807 021634 005003 CLR R3
5808 021636 004737 036100 JSR PC,LDMEM2 ;ZERO THE BLK IN MEMORY
5809 021642 112765 000121 000001 MOVB #R0DATA.P.CMND(R5) ;SET READ COMMAND
5810 021650 004737 040340 JSR PC,TRNSFR ;READ THIS DATA BACK
5811 021654 005204 INC R4 ;INCREMENT THE DATA WORD
5812 021656 004737 037230 JSR PC,INCRMA ;COMPUTE NEW MA TO TRY
5813 021662 000710 BR 4$ ;GO WRITE AND READ NEXT BLK
5814 ;PERFORM SOFTWARE COMPARES OF ALL MEMORY BLOCKS
5815 021664 010403 20$: MOV R4,R3 ;GET DATA WORD
5816 021666 004737 036244 JSR PC,CKMEM2 ;COMPARE THIS BLK TO GOOD DATA WORD
5817 021672 005204 INC R4 ;INCREMENT THE DATA WORD
5818 021674 004737 037230 JSR PC,INCRMA ;GET NEW MA TO TRY
5819 021700 000701 BR 4$ ;GO COMPARE NEXT MEM BLK
5820
5821 021702 005137 005526 30$: COM SELECT ;COMPLEMENT THE FLAG
5822 021706 001401 BEQ 50$ ;BR IF ALL DONE NOW
5823 021710 000671 BR 2$ ;BR IF COMPARES SHOULD BE DONE NOW
5824 021712 004737 030502 50$: JSR PC,GETXDP ;RESTORE SAVED XXDP
5825 021716 105737 003134 TSTB UBMPRS ;SEE IF UNIBUS MAP PRESENT
5826 021722 001402 BEQ 54$ ;BR IF NOT
5827 021724 005037 172516 CLR @*SR3 ;DISABLE UNIBUS MAP
5828 021730
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849 021730 000004
5850 021732 012737 000004 001324
5851 021740 004737 032060
5852 021744 004737 032126
5853 021750 000137 022300
5854
5855 021754 004737 030726
5856 021760 004737 027250
5857
5858 021764 004737 030422
5859 021770 012737 064574 005542
5860 021776 005037 005544
5861 022002 004737 030474

```

```

*****
*TEST 4 NPR/MEMORY DATA PATTERN TEST
*BEFORE TESTING, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS
*RWBUF. THEN, ALL PHYSICAL MEMORY LOCATIONS, STARTING AT THE
*FIRST ADRS OF THE NEXT 1K MEMORY BLOCK BEYOND RWBUF+6000(OCT),
* ARE EXERCISED WITH UP TO 16 DATA PATTERNS
* CHOSEN IN PARAMETER PT. (THESE ARE THE SAME PATTERNS
*PROVIDED FOR USE IN THE READ/WRITE DATA TEST). IF PT = 0, THE
*DATA PATTERNS WILL DEFAULT TO PATS 08,09,10, AND 11.
*MEMORY IS EXERCISED IN BLOCKS OF UP TO 64K WORDS, STARTING AT
*THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE BUFFER.
*EACH BLOCK IS LOADED WITH DATA, AND WRITTEN ONTO DISK AT ADDRESS
*FC,FT,FS (SCALED TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
*SECTOR FILE). THEN THE MEMORY BLOCK IS ZEROED AND READ BACK AND
*COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN.
*POSSIBLY INCLUDING THE USER-DEFINED PATTERN 15, IF SPECIFIED.
*****

```

```

TST4: SCOPE
MOV #4,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
JSR PC,SETUP ;:SET UP FOR LOOP ON ERROR
JSR PC,CHKITR ;:SEE IF ITER. NO. = 0 FOR THIS TEST
JMP TST5 ;:JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
JSR PC,INITSS ;:INITIALIZE DRIVER PARAMS AND SUB-SYS
JSR PC,PREPKB ;:PREPARE FOR POSSIBLE KBD INPUT
;SAVE XXDP LOADER, IF PRESENT
JSR PC,FNDXDP ;:COMPUTE STARTING ADR OF XXDP
MOV #RWBUF,XDPSAV ;:SET XXDP SAVE AREA ADR
CLR XDPSAV+2
JSR PC,SAVXDP ;:GO SAVE XXDP LOADER

```

```

5862 022006 112737 000001 003124      MOVB      #1,XOVLAD      ;SET INDICATORS
5863 022014 112737 000001 003125      MOVB      #1,XDPSVD
5864 022022 004737 034212      JSR       PC,LODP14    ;GENERATE PSEUDO-RAND PAT 14
5865 022026 013737 005670 005536      MOV       PT,PATRN     ;GET A COPY OF PT
5866 022034 001003      BNE       2$           ;BR IF PT NON-ZERO
5867 022036 012737 007400 005536      MOV       #7400,PATRN  ;SET DEFLT PATRNS = PAT 8,5,10,11
5868                                     ;INIT PARAMETERS, BEGIN TESTING
5869 022044 004737 037022      2$: JSR     PC,SETUPM   ;INIT PARAMS
5870 022050 004737 033450      4$: JSR     PC,MXWRDC  ;COMPUTE MAX WORD COUNT FOR THIS MA
5871 022054 005701      TST      R1           ;SEE IF ALL MEMORY TESTED YET
5872 022056 100501      BMI      50$         ;BR IF ALL DONE
5873 022060 005065 000012      CLR      P.WC(R5)     ;INIT WORD COUNT TO 65,536(DEC)
5874 022064 005701      TST      R1           ;SEE IF WRD CNT SHOULD BE 65,536
5875 022066 001003      BNE      6$          ;BR IF YES
5876 022070 005400      NEG      R0
5877 022072 010065 000012      MOV      R0,P.WC(R5)  ;SET WORD COUNT
5878 022076 013765 005600 000010 6$: MOV      MA,P.BALO(R5) ;SET BA BITS 0-15
5879 022104 013701 005602      MOV      MA+2,R1     ;GET MA BITS 16-21
5880 022110 042701 177774      BIC      #177774,R1  ;MASK FOR LO 2 BITS
5881 022114 150165 000007      BISB     R1,P.BAHI(R5) ;SET BA BITS 16,17
5882 022120 005737 056216      TST      $K11        ;SEE IF MEM MGT PRESENT
5883 022124 100010      BPL      14$         ;BR IF NOT
5884                                     ;SET UP MEM MGT
5885 022126 013737 005600 005604      MOV      MA,PMA       ;SET BUFFER ADDRESS
5886 022134 013737 005602 005606      MOV      MA+2,PMA+2
5887 022142 004737 030300      JSR      PC,PREPAR    ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5888 022146 012704 000001      14$: MOV     #1,R4     ;SET PATTERN BIT POINTER
5889 022152 030437 005536      16$: BIT     R4,PATRN  ;SEE IF THIS PATTERN IS CHOSEN
5890 022156 001003      BNE      20$         ;BR IF YES
5891 022160 006304      18$: ASL     R4        ;SHIFT TO POINT TO NEXT PATTERN
5892 022162 001434      BEQ      24$         ;BR IF NO MORE LEFT TO CHECK
5893 022164 000772      BR       16$         ;GO CHECK FOR ANOTHER PATTERN
5894 022166 010401      20$: MOV     R4,R1     ;GET A COPY OF BIT POINTER
5895                                     ;PERFORM THE TESTS
5896 022170 012737 022170 001110      22$: MOV     #.,$LPERR ;SET NEW LOOP ON ERROR ADRS
5897 022176 004737 032060      JSR      PC,SETUP     ;SET UP FOR LOOP ON ERROR
5898 022202 004737 040226      JSR      PC,SVPRMS    ;SAVE THE PARAMS FOR THIS XFER
5899 022206 004737 034426      JSR      PC,LODBUF    ;LOAD THIS MEM BLK WITH SELECTED DATA
5900 022212 112765 000123 000001      MOVB     #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5901 022220 004737 040340      JSR      PC,TRNSFR    ;WRITE THE DATA
5902 022224 005003      CLR      R3          ;SET DATA WORD = 0
5903 022226 004737 036100      JSR      PC,LDMEM2    ;LOAD MEM BLK WITH ALL ZEROS
5904 022232 112765 000121 000001      MOVB     #R0DATA,P.CMND(R5) ;SET READ COMMAND
5905 022240 004737 040340      JSR      PC,TRNSFR    ;READ THE DATA FROM DISK
5906 022244 004737 034654      JSR      PC,CMPBUF    ;PERFORM SOFTWARE COMPARE OF DATA
5907 022250 104410      SCOPER
5908 022252 000742      BR       18$         ;CHECK FOR INTERNAL LOOP ON ERROR
5909                                     ;CHECK FOR NEXT PATTERN
5910 022254 004737 037230      24$: JSR     PC,INCRMA  ;COMPUTE NEXT MA
5911 022260 000673      BR       4$          ;GO SEE IF MORE MEMORY TO TEST
5912 022262 004737 030502      50$: JSR     PC,GETXDP  ;RESTORE XXDP LOADER
5913 022266 105737 003134      TSTB     UBMPRS      ;SEE IF UNIBUS MAP PRESENT
5914 022272 001402      BEQ      54$         ;BR IF NOT PRESENT
5915 022274 005037 172516      CLR      @#SR3       ;DISABLE UNIBUS MAP
5916 022300
5917

```

5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
5951
5952
5953
5954
5955
5956
5957
5958
5959
5960
5961
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971
5972
5973

022300	000004		
022302	012737	000005	001324
022310	004737	032160	
022314	004737	032126	
022320	000137	022604	
022324	004737	030726	
022330	004737	027250	
022334	013765	005650	000002
022342	113765	005654	000005
022350	105065	000004	
022354	012765	165000	000012
022362	105737	003115	
022366	001403		
022370	012765	166000	000012
022376	052765	100000	000014
022404	012700	006766	
022410	005737	005670	
022414	100402		
022416	012700	006666	
022422	012701	000020	
022426	012703	000024	

```

*****
*TEST 5 UNIBUS CONTENTION TEST
*THE PURPOSE OF THIS TEST IS TO EXERCISE THE NPR CONTROL LOGIC
*WITHIN THE CONTROLLER IN SUCH A MANNER THAT A CONTENTION FOR
*UNIBUS CYCLES ARISES BETWEEN THE PROCESSOR AND THE RK06.
*
*FIRST, A WRITE DATA IS BEGUN AT ADDRESS FC FT SECTOR 0, WITH
*THE BUS ADDRESS INHIBIT BIT (BAI) SET IN THE CONTROLLER.
*USING A WORD COUNT OF 12000(OCT) OR 13000(OCT) (DEPENDING ON THE
*FORMAT), THE ENTIRE TRACK IS WRITTEN WITH THE FIRST WORD OF PATTERN
*15 (IF SELECTED) OR THE FIRST WORD OF PAT 13 (BY DEFAULT).
* THEN, AN INSTRUCTION LOOP IS ENTERED IN WHICH A NON-
*EXISTENT UNIBUS DEVICE ADDRESS IS REFERENCED ONCE PER PASS. THIS
*WILL RESULT IN A PROCESSOR SEIZURE OF THE UNIBUS, FOR 5-20 USEC,
*(DEPENDING ON THE PROCESSOR), FOLLOWED BY A NON-EXISTENT MEMORY
*PROCESSOR TRAP, ONCE PER PASS. THUS, FOR EACH PASS THROUGH THE
*INSTRUCTION LOOP THE CONTROLLER IS FORCED TO LOSE FROM ONE
*TO FOUR NPR BUS CYCLES. THE EXECUTION TIME OF THE LOOP DETERMINES THE
*REPETITION RATE OF THE NON-EXISTENT MEMORY REFERENCES, AND THIS
*IS CHOSEN TO ALLOW ENOUGH NPR'S SO THAT THERE SHOULD BE NO
*DATA LATE ERRORS IN A FAULT-FREE CONTROLLER.
*THE DATA IS THEN WRITE CHECKED FOR VERIFICATION, AND THE ABOVE
*TEST IS REPEATED USING A READ DATA COMMAND. NO ERRORS ARE EXPECTED.
*
*THE ABOVE OPERATIONS ARE REPEATED 15 MORE TIMES, USING THE NEXT WORD
*OF THE PROPER PATTERN (13 OR 15) EACH TIME, TO WRITE AND READ
*THE ENTIRE TRACK. IF A DATA LATE ERROR OCCURS ON A GIVEN TRANSFER,
*THAT DATA IS NOT WRITE-CHECKED OR COMPARED.

```

```

*****
TSTS: SCOPE
MOV #5,STESTN ;SET TEST NUMBER IN APT MAIL BOX
JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
JSR PC,CHKITR ;SEE IF ITER. NO. = 0 FOR THIS TEST
JMP TST6 ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
JSR PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
;INITIALIZE PARAMETERS
MOV FC,P.CYLN(R5) ;SET CYL = FC
MOVB FT,P.TRCK(R5) ;SET TRACK = FT
CLRB P,SECT(R5) ;SET SECTOR = 0
MOV #-13000,P.WC(R5) ;INIT WORD COUNT FOR FULL TRACK
TSTB FORMAT ;DETERMINE THE FORMAT
BEQ 4$ ;BR IF 22 SECTORS
MOV #-12000,P.WC(R5) ;SET WORD COUNT FOR FULL TRACK
4$: BIS #DTBAII,P.PRST(R5) ;SET BUS ADDRESS INHIBIT
;WRITE DATA WITH ALLOWABLE UNIBUS CONTENTION, AND WRITE CHECK IT
MOV #PAT15,R0 ;SET DATA PATTERN ADDRESS
TST PT ;SEE IF USER-DEFINED PATTERN 15 SELECTED
BMI 14$ ;BR IF YES
MOV #PAT13,R0 ;SET DATA PATTERN 13 ADDRESS
14$: MOV #16.,R1 ;SET COUNTER = 16
MOV #20.,R3 ;SET NON-EXISTENT MEMORY TIMER

```

```

5974 022432 010065 000010      16$:  MOV      RO,P.BALO(R5)      ;SET BA BITS
5975 022436 105037 003132      CLR      DLIFLG              ;CLEAR DATA LATE FLAG
5976 022442 112737 000001 003133  MOV      #1,NORTRY          ;SET "NO-RETRY" FLAG
5977 022450 112765 000123 000001  MOV      #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5978 022456 004737 037126      JSR      PC,REFNEM          ;WRITE DATA DURING NEM REF.
5979 022462 105737 003132      TSTB    DLIFLG              ;SEE IF DATA LATE ERROR OCCURRED
5980 022466 001011 18$:      SNE      18$                ;BR IF NOT
5981 022470 032737 000002 005476  BIT      #BSERR,RECODE      ;SEE IF BAD SECTOR ERROR
5982 022476 001042 24$:      BNE      24$                ;BR IF YES
5983 022500 112765 000131 000001  MOV      #WATCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
5984 022506 004737 040716      JSR      PC,DRVCAL          ;DO WRITE CHECK
5985                                ;READ DATA WITH ALLOWABLE UNIBUS CONTENTION, AND COMPARE LAST WORD
5986 022512 012765 064574 000010 18$:  MOV      #RWBUF,P.BALO(R5) ;SET BA = RWBUF ADDRESS
5987 022520 005037 064574      CLR      RWBUF              ;CLEAR THE BUFFER WORD
5988 022524 112765 000121 000001  MOV      #RDATA,P.CMND(R5) ;SET READ COMMAND
5989 022532 004737 037126      JSR      PC,REFNEM          ;READ DATA DURING NEM REF.
5990 022536 105737 003132      TSTB    DLIFLG              ;SEE IF DATA LATE ERROR OCCURRED
5991 022542 001014 20$:      BNE      20$                ;BR IF YES
5992 022544 022037 064574      CMP      (RO)+,RWBUF        ;COMPARE LAST DATA WORD
5993 022550 001411 20$:      BEQ      20$                ;BR IF EQUAL
5994 022552 004737 042130      JSR      PC,REPSUP          ;GATHER STATUS FOR PRINTOUT
5995 022556 016037 177776 001174  MOV      -2(RO),SREG5        ;GOOD DATA WORD
5996 022564 013737 064574 001176  MOV      RWBUF,SREG6        ;BAD DATA WORD
5997 022572 104110      ERROR   110                ;"READ ERROR WHILE BAI SET"
5998                                ;SET UP FOR NEXT PASS THROUGH LOOP
5999 022574 005301 20$:      DEC      R1                ;SEE IF ALL DONE YET
6000 022576 001402 24$:      BEQ      24$                ;BR IF YES
6001 022600 000137 022432      JMP      16$                ;JUMP TO CONTINUE TESTING
6002 022604
6003
6004
6005
6006
6007
6008
6009
6010
6011
6012
6013
6014
6015
6016
6017
6018
6019
6020
6021
6022
6023
6024
6025
6026
6027
6028
6029

```

```

;*****
; *TEST 6 MULTI-DRIVE INTERFERENCE TEST
; *THIS TEST PERFORMS MULTI-DRIVE POSITIONING OPERATIONS, WHILE A
; *LARGE DATA TRANSFER IS IN PROGRESS ON THE DRIVE UNDER TEST, FOR
; *THE PURPOSE OF DETECTING PROBLEMS OF CONCURRENT DRIVE OPERATION.
; *THE TEST IS RUN ONLY IF THERE IS MORE THAN 1 DRIVE ON THE SUBSYSTEM.
; *
; *THE TEST PROCEEDS AS FOLLOWS : IT IS FIRST DETERMINED WHICH DRIVE(S)
; *(BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES
; *ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH.NEXT, A SEEK IS
; *DONE TO CYLINDER FC (SCALED, IF NECESSARY,) ON THE DRIVE UNDER
; *TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH
; *TO A DIFFERENT PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE
; *FROM 10-67 MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN,
; *A WRITE DATA COMMAND IS BEGUN ON THE DRIVE UNDER TEST AT THE CURRENT
; *CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT
; *INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616
; *(DEC) WORDS IF 20 SECTOR FORMAT OR 17,152(DEC) WORDS IF 22 SECTOR
; *FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA.
; *THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN
; *THAT TIME, ALL THE OTHER DRIVES SHOULD HAVE COMPLETED POSITIONING.
; *NEXT, A WRITE CHECK IS DONE TO VERIFY THE DATA WRITTEN ON THE DRIVE
; *UNDER TEST, AND A CHECK IS MADE TO INSURE THAT ALL OTHER DRIVES WHICH
; *PERFORMED SEEKS WERE ABLE TO COMPLETE THEM SUCCESSFULLY, BY CHECKING

```

```

6030      : *ATTENTION BITS AND COMPARING DRIVE CYLINDER ADDRESSES TO EXPECTED
6031      : *VALUES.
6032      : *****
6033 022604 000004          TST6: SCOPE
6034 022606 012737 000006 001324  MOV    #6, $TESTN      ; SET TEST NUMBER IN APT MAIL BOX
6035 022614 004737 032060          JSR    PC, SETUP      ; SET UP FOR LOOP ON ERROR
6036 022620 004737 032126          JSR    PC, CHKITR     ; SEE IF ITER. NO. = 0 FOR THIS TEST
6037 022624 000137 023442          JMP    $EOP          ; JUMP TO END-OF-PASS ROUTINE
6038      ; RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
6039 022630 004737 030726          JSR    PC, INITSS    ; INITIALIZE DRIVER PARAMS AND SUB-SYS
6040 022634 004737 027250          JSR    PC, PREPKB    ; PREPARE FOR POSSIBLE KBD INPUT
6041      ; CHECK DRVLST FOR OTHER AVAILABLE DRIVES, AND RECALIBRATE THEM
6042 022640 112765 000117 000001  MOVB   #SEEK, P.CMND(R5) ; SET SEEK COMMAND
6043 022646 005001          CLR    R1            ; INIT MULTIPLE-DRIVES FLAG
6044 022650 005000          CLR    R0            ; INIT DRIVE NO. TO 0
6045 022652 005065 000002          CLR    P.CYLN(R5)    ; SET CYL = 0
6046 022656 022700 000010 4$:   CMP    #10, R0       ; SEE IF ALL DRIVES CHECKED YET
6047 022662 001415          BEQ    $8$          ; BR IF ALL DONE
6048 022664 105760 005610          TSTB   DRVLST(R0)    ; SEE IF THIS DRIVE IS MARKED IN LIST
6049 022670 001410          BEQ    $6$          ; BR IF NOT MARKED
6050 022672 120037 005502          CMPB   R0, DRIVE     ; SEE IF MARKED DRIVE IS DRIVE UNDER TEST
6051 022676 001405          BEQ    $6$          ; BR IF YES
6052 022700 110065 000000          MOVB   R0, P.DRVN(R5) ; SET DRIVE NO. PARAMETER
6053 022704 004737 040716          JSR    PC, DRVCAL    ; SEEK TO CYL 0 ON CURRENT DRIVE
6054 022710 005201          INC    R1            ; SET FLAG TO INDICATE MORE THAN 1 DRIVE
6055 022712 005200 6$:   INC    R0            ; INCR DRIVE NO.
6056 022714 000760          BR     $4$          ; KEEP CHECKING DRIVES
6057 022716 005701 8$:   TST    R1            ; SEE IF MORE THAN 1 DRIVE PRESENT
6058 022720 001002          BNE    $9$          ; BR IF YES, TO RUN TEST
6059 022722 000137 023442          JMP    MULT11        ; NO, SKIP TEST
6060      ; SEEK TO CYLINDER FC ON DRIVE UNDER TEST, SET PARAMS FOR DATA XFER
6061 022726 004737 030726 9$:   JSR    PC, INITSS    ; INIT S.S.
6062 022732 013765 005650 000002  MOV    FC, P.CYLN(R5) ; SET CYL = FC
6063 022740 005065 000004          CLR    P.SECT(R5)    ; SET TRACK AND SECTOR = 0
6064 022744 012765 136400 000012  MOV    #-17152., P.WC(R5) ; SET WC FOR 67(DEC) SECTORS
6065 022752 105737 003115          TSTB   FORMAT        ; CHECK THE FORMAT
6066 022756 001403          BEQ    $12$         ; BR IF 22-SECTOR FORMAT
6067 022760 012765 141400 000012  MOV    #-15616., P.WC(R5) ; SET WC FOR 61(DEC) SECTORS
6068 022766 012765 006666 000010 12$:  MOV    #PAT13, P.BALO(R5) ; SET BA BITS
6069 022774 052765 100000 000014  BIS    #DTBA11, P.PRST(R5) ; SET BUS ADDRESS INCREMENT INHIBIT
6070 023002 112765 000117 000001  MOVB   #SEEK, P.CMND(R5) ; SET SEEK COMMAND
6071 023010 004737 040716          JSR    PC, DRVCAL    ; SEEK TO CYL FC ON DRIVE UNDER TEST
6072      ; DO PSEUDO-RANDOM SEEKS ON ALL OTHER DRIVES
6073 023014 105037 003110          CLRB   TSTING        ; INHIBIT ↑C, ↑Z ESCAPE
6074 023020 012701 000007          MOV    #7, R1        ; SET LOOP COUNTER
6075 023024 012700 003204          MOV    #CYLLST, R0   ; ADRS OF RAND CYL LIST
6076 023030 004737 034044 15$:  JSR    PC, RNDADR     ; GENERATE A PSEUDO-RANDOM CYL NO.
6077 023034 013720 005506          MOV    CYLNR, (R0)+  ; SET CYL NO. IN TABLE
6078 023040 001002          BNE    $16$         ; BR IF NOT 0
6079 023042 005260 177776          INC    -2(R0)        ; MAKE IT NON-ZERO
6080 023046 005301 16$:  DEC    R1            ; DECREMENT LOOP COUNTER
6081 023050 001367          BNE    $15$         ; BR IF NOT DONE LOADING LIST YET
6082 023052 012701 003204          MOV    #CYLLST, R1   ; GET ADRS OF RAND CYL TABLE
6083 023056 010537 023126          MOV    R5, $19$     ; SET PARAM BLK ADRS FOR DRIVER
6084 023062 005000          CLR    R0            ; INIT DRIVE NO. TO 0
6085 023064 022700 000010 18$:  CMP    #10, R0       ; SEE IF ALL DRIVES CHECKED YET

```

```

6086 023070 001421          BEQ      22$          ;BR IF YES
6087 023072 105760 005610  TSTB    DRVLST(RO)  ;SEE IF THIS DRIVE PRESENT
6088 023076 001414          BEQ      20$          ;BR IF NO
6089 023100 120037 005502  CMPB    RO,DRIVE    ;SEE IF THIS IS DRIVE UNDER TEST
6090 023104 001411          BEQ      20$          ;BR IF YES
6091 023106 110065 000000  MOVB    RO,P.DRVN(R5) ;SET DRIVE NO. PARAMETER
6092 023112 012155 000002  MOV     (R1)+,P.CYLN(R5) ;SET CYLINDER NO.
6093 023116 004737 041032  JSR     PC,STRCMD    ;STORE PREV AND CURRENT CMNDS
6094 023122 004737 051042  JSR     PC,C.INIT    ;GO START SEEK ON THIS DRIVE
6095 023126 000000          19$:    .WORD        ;P.B. ADRS GOES HERE
6096 023130 005200          20$:    INC      RO      ;INCR DRIVE NO.
6097 023132 000754          BR       18$          ;BR TO SEEK ON NEXT DRIVE
6098                                     ;WRITE DATA ON DRIVE UNDER TEST, WITH BAI SET
6099 023134 112765 000123 000001  22$:    MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
6100 023142 113765 005502 000000  MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6101 023150 013765 005650 000002  MOV     FC,P.CYLN(R5) ;SET CYL NO.
6102 023156 012737 023354 003034  MOV     #MULHDL,A.NORM ;SET SPECIAL DRIVER RETURN ADDRESS
6103 023164 004737 040716          JSR     PC,DRVCAL    ;WRITE 1 CYL + 1 SECTOR AT CYL FC
6104 023170 032737 000002 005476  BIT     #BSERR,RECODE ;SEE IF BSE ERROR
6105 023176 001121          BNE     MULT11       ;BR IF YES
6106 023200 112765 000131 000001  MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6107 023206 004737 040716          JSR     PC,DRVCAL    ;WRITE CHECK THE DATA WRITTEN
6108                                     ;CHECK ATTENTIONS AND CHECK CYLINDER ADDRESSES FROM ALL OTHER DRIVES
6109 023212 112737 000001 003110  MOVB    #1,TSTING     ;ALLOW ↑C, ↑Z ESCAPE
6110 023220 112765 000141 000001  MOVB    #R0STAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
6111 023226 012701 003204          MOV     #CYLLST,R1   ;ADRS OF RAND CYL LIST
6112 023232 005000          CLR     RO           ;INIT DRIVE NO.
6113 023234 022700 000010          26$:    CMP     #10,RO      ;SEE IF ALL DRIVES CHECK'D YET
6114 023240 001500          BEQ     MULT11       ;BR IF YES
6115 023242 105760 005610  TSTB    DRVLST(RO)  ;SEE IF THIS DRIVE MARKED IN LIST
6116 023246 001440          BEQ     32$          ;BR IF NOT
6117 023250 120037 005502  CMPB    RO,DRIVE    ;SEE IF THIS IS DRIVE UNDER TEST
6118 023254 001435          BEQ     32$          ;BR IF YES
6119 023256 136037 003072 003202  BITB    I.DRV(RO),SAVWRD ;SEE IF GOT ATT'N FROM THIS DRIVE
6120 023264 001003          BNE     28$          ;BR IF YES
6121 023266 004737 042130          JSR     PC,REPSUP    ;PREPARE STATUS FOR PRINTOUT
6122 023272 104113          ERROR   113         ;"NO ATTENTION ON SEEK"
6123 023274 110065 000000          28$:    MOVB    RO,P.DRVN(R5) ;SET THIS DRIVE NO.
6124 023300 004737 040716          JSR     PC,DRVCAL    ;READ STATUS OF THIS DRIVE
6125 023304 016503 000052  MOV     P.B10(R5),R3 ;GET STATUS BYTE B10
6126 023310 006003          ROR     R3           ;GET CYL ADRS RIGHT-JUSTIFIED
6127 023312 006003          ROR     R3
6128 023314 006003          ROR     R3
6129 023316 006003          ROR     R3
6130 023320 042703 177000  BIC     #177000,R3   ;CLEAR OFF UNUSED BITS
6131 023324 022103          CMP     (R1)+,R3    ;COMPARE TO EXPECTED CYLINDER
6132 023326 001411          BEQ     32$          ;BR IF EQUAL
6133 023330 016137 177776 001212  MOV     -2(R1),SREG14 ;GOOD CYL NO.
6134 023336 010337 001214          MOV     R3,SREG15   ;BAD CYL NO.
6135 023342 004737 042130          JSR     PC,REPSUP    ;GATHER STATUS FOR PRINTOUT
6136 023346 104114          ERROR   114         ;"DRIVE'S CYLINDER INCORRECT"
6137 023350 005200          32$:    INC     RO           ;INCREMENT DRIVE NO.
6138 023352 000730          BR       26$          ;BR IF NOT DONE WITH ALL DRIVES YET
6139
6140
6141

```

;THIS IS THE NORMAL RETURN FROM THE DRIVER WHICH IS USED TO CHECK
;THE RESULTS OF MULTI-DRIVE OPERATIONS.


```

6142 023354 032737 000200 002770 MULHDL: BIT #RDY,T.CS1 ;SEE IF CNTRLR RDY IS SET
6143 023362 001010 BNE 6$ ;BR IF RDY SET
6144 023364 004737 050442 JSR PC,I.CSTS ;READ CONTROLLER REGISTERS
6145 023370 013765 002770 000016 MOV T.CS1,P.CS1(R5) ;GET CS1 BITS
6146 023376 004737 042130 JSR PC,REPSUP ;PREPARE STATUS FOR PRINTOUT
6147 023402 104112 ERROR 112 ;"INTRPT WHEN CNTRLR NOT RDY"
6148 023404 013737 003004 003202 6$: MOV T.ASOF,SAVWRD ;SAVE ATT'N SUMMARY
6149 023412 000337 003202 SWAB SAVWRD ;GET ATT'N BITS IN BITS 0-7
6150 023416 112765 000177 000001 MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6151 023424 012737 041142 003034 MOV #ERRFRE,A.NORM ;RESTORE NORMAL DRIVER RETURN ADDRESS
6152 023432 004737 040716 JSR PC,DRVCAL ;CLEAR SUB-SYS
6153 023436 000137 041142 JMP ERRFRE ;TAKE NORMAL RETURN
6154
6155 023442 MULTI1:
6156
6157
6158
6159 .SBTTL END OF PASS ROUTINE
6160
6161 ;*****
6162 ;*INCREMENT THE PASS NUMBER ($PASS)
6163 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
6164 ;*IF THERES A MONITOR GO TO IT
6165 ;*IF THERE ISN'T JUMP TO NEWPAS
6166
6167 $EOP:
6168 023442 000004 SCOPE
6169 023444 005237 001330 INC $DEVCT ;INCREMENT DEVICE COUNT FOR APT
6170 023450 000137 017436 JMP NEWDRV ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
6171 023454 DUNPAS: ;THIS IS TRULY THE END OF A PASS
6172 023454 042777 000100 155462 BIC #BIT6,@$TKS ;DISABLE TTY KBD INTERRUPT
6173 023462 005037 001102 CLR $TSTNM ;ZERO THE TEST NUMBER
6174 023466 005037 001304 CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
6175 023472 005237 001326 INC $PASS ;INCREMENT THE PASS NUMBER
6176 023476 042737 100000 001326 BIC #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
6177 023504 005327 DEC (PC)+ ;LOOP?
6178 023506 000001 $EOPCT: .WORD 1
6179 023510 003022 BGT $DOAGN ;;YES
6180 023512 012737 MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
6181 023514 000001 $ENDCT: .WORD 1
6182 023516 023506 $EOPCT
6183 023520 104400 023565 TYPE $ENDMG ;;TYPE "END PASS #"
6184 023524 013746 001326 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
6185 023530 104404 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
6186 023532 104400 023562 TYPE $ENULL ;;TYPE A NULL CHARACTER
6187 023536 013700 000042 $GET42: MOV @#42,RO ;;GET MONITOR ADDRESS
6188 023542 001405 BEQ $DOAGN ;;BRANCH IF NO MONITOR
6189 023544 000005 RESET ;;CLEAR THE WORLD
6190 023546 004710 $ENDAD: JSR PC,(RO) ;;GO TO MONITOR
6191 023550 000240 NOP ;;SAVE ROOM
6192 023552 000240 NOP ;;FOR
6193 023554 000240 NOP ;;ACT11
6194 023556 $DOAGN:
6195 023556 000137 JMP @ (PC)+ ;;RETURN
6196 023560 017424 $RTNAD: .WORD NEWPAS
6197 023562 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING

```

M11

DZR6N-8 - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 143
DZR6NB.CMB END OF PASS ROUTINE

6198	023565	015	042412	042116	\$ENDMG: .ASCIZ <15><12>/END PASS #/
6199	023572	050040	051501	020123	
6200	023600	000043			
6201					
6202					.SBTTL RK06 HEAD ALIGNMENT AID


```

6203 023602 012737 000340 177776 ASTART: MOV #PR7, @#PS
6204 .SBTTL INITIALIZE THE COMMON TAGS
6205 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
6206 023610 012706 001100 MOV #CMTAG, R6 ;;FIRST LOCATION TO BE CLEARED
6207 023614 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
6208 023616 022706 001140 CMP #SWR, R6 ;;DONE?
6209 023622 001374 BNE -6 ;;LOOP BACK IF NO
6210 023624 012706 001100 MOV #STACK, SP ;;SETUP THE STACK POINTER
6211 ;;INITIALIZE A FEW VECTORS
6212 023630 012737 055426 000020 MOV #SCOPE, @#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
6213 023636 012737 000340 000022 MOV #340, @#IOTVEC+2 ;;LEVEL 7
6214 023644 012737 054722 000030 MOV #ERROR, @#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
6215 023652 012737 000340 000032 MOV #340, @#EMTVEC+2 ;;LEVEL 7
6216 023660 012737 056562 000034 MOV #TRAP, @#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
6217 023666 012737 000340 000036 MOV #340, @#TRAPVEC+2;LEVEL 7
6218 023674 012737 055272 000024 MOV #PWRDN, @#PWRVEC ;;POWER FAILURE VECTOR
6219 023702 012737 000340 000026 MOV #340, @#PWRVEC+2 ;;LEVEL 7
6220 023710 013737 023514 023506 MOV $ENDCT, $EOPCT ;;SETUP END-OF-PROGRAM COUNTER
6221 023716 005037 001304 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
6222 023722 005037 001 J6 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
6223 023726 112737 000J01 001115 MOVB #1, $ERMAX ;;ALLOW ONE ERROR PER TEST
6224 023734 012737 023734 001106 MOV #., $LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
6225 023742 012737 023742 001110 MOV #., $LPERR ;;SETUP THE ERROR LOOP ADDRESS
6226 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
6227 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
6228 023750 013746 000004 MOV @#ERRVEC, -(SP) ;;SAVE ERROR VECTOR
6229 023754 012737 024010 000004 MOV #64$, @#ERRVEC ;;SET UP ERROR VECTOR
6230 023762 012737 177570 001140 MOV #DSWR, SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
6231 023770 012737 177570 001142 MOV #DDISP, DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
6232 023776 022777 177777 155134 CMP #-1, @SWR ;;TRY TO REFERENCE HARDWARE SWR
6233 024004 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
6234 ;;AND THE HARDWARE SWR IS NOT = -1
6235 024006 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
6236 024010 012716 024016 64$: MOV #65$, (SP) ;;SET UP FOR TRAP RETURN
6237 024014 000002 RTI
6238 024016 012737 000176 001140 65$: MOV #SWREG, SWR ;;POINT TO SOFTWARE SWR
6239 024024 012737 000174 001142 MOV #DISPREG, DISPLAY
6240 024032 012637 000004 66$: MOV (SP)+, @#ERRVEC ;;RESTORE ERROR VECTOR
6241
6242 024036 005037 001326 CLR $PASS ;;CLEAR PASS COUNT
6243 024042 132737 000200 001341 BITB #APTSIZE, $ENVM ;;TEST USER SIZE UNDER APT
6244 024050 001403 BEQ 67$ ;;YES, USE NON-APT SWITCH
6245 024052 012737 001342 001140 MOV #SSWREG, SWR ;;NO, USE APT SWITCH REGISTER
6246 024060 67$:
6247 .SBTTL RK06 HEAD ALIGNMENT AID
6248 024060 000005 RESET ;;RESET UNIBUS
6249 024062 104400 007026 TYPE ,DZR6N ;;TYPE PROGRAM I.D. FOR PART 2
6250 024066 104400 007040 TYPE ,SUBVER
6251 024072 104400 007121 TYPE ,PART2
6252 024076 012737 000176 001140 MOV #SWREG, SWR ;;ADRS OF SOFTWARE SWR
6253 024104 005077 155030 CLR @SWR ;;MAKE SWR BITS ALL 0
6254 024110 012737 027110 000060 MOV #KBDHDL, @#TKVEC ;;LOAD VECTOR FOR TTY KBD
6255 024116 012737 000200 000062 MOV #PR4, @#TKVEC+2 ;;SET KBD PRIORITY = 4
6256 024124 013701 003030 MOV RKVEC, R1 ;;GET ADDRESS OF VECTOR STORAGE
6257 024130 012721 045542 MOV #I.INTR, (R1)+ ;;SET IT TO INTERRUPT HNDLR
6258 024134 012711 000340 MOV #PR7, (R1) ;;SET INTERRUPT HANDLER PR7

```

```

6259 024140 012737 000000 177776      MOV      #PRO,PS      :ALLOW ALL INTERRUPTS
6260 024146 012737 000000 005650      MOV      #C,FC       :SET FIRST CYL = 0
6261 024154 012737 000632 005652      MOV      #LSTCY,LC   :SET LAST CYL = 410(10)
6262 024162
6263 024162 012737 176543 053170      MOV      #176543,SHINUM :INITIALIZE PSEUDO-RANDOM NUMBERS
6264 024162 012737 123456 053172      MOV      #123456,SLONUM
6265 024170 105037 003106      CLRB     MFLAG       :CLEAR PARAM INP FLAG
6266 024176 105037 003110      CLRB     TSTING      :CLEAR "RUNNING TESTS" FLAG
6267 024202 005037 005504      CLR      STALLS      :DON'T STALL ON OPERATIONS
6268 024206 012701 005224      MOV      #PRVCMO,R1  :ZERO OUT PREV COMMAND
6269 024212 012700 000006      MOV      #6,RO
6270 024216 005021 425:      CLR      (R1)+
6271 024222 005200      DEC      RO
6272 024224 001375      BNE     425
6273 024226 105037 003116      CLRB     ERRCNT      :CLEAR ERROR COUNT FOR RESTARTS
6274 024230 104400 011431      TYPE    ,IDENT      :TYPE PROGRAM IDENTIFICATION
6275 024234 105737 003136      TSTB    #LPOVL      :SEE IF HELP FILE OVERLAID
6276 024240 104400 011476      BNE     ASKMOD      :BR IF YES
6277 024244 104400 011476      TYPE    ,FORHLP     :ASK IF HELP DESIRED
6278 024246 112501      RDCHR   (SP)+,R1    :READ RESPONSE
6279 024252 122701 000015      MOV     #C15,R1     :GET CHARACTER
6280 024254 104400 065213      CMP     ASKMOD      :SEE IF <CR> TYPED
6281 024256 104400 065213      BEQ     #HLPFIL     :BR IF NO HELP NEEDED
6282 024262
6283 024264
6284
6285
6286
6287 024270 104400 011533      :DETERMINE DESIRED OPERATIONAL MODE
6288 024274 004737 027250      ASKMOD: TYPE    TYPMOD :ASK FOR DESIRED OPERATIONAL MODE
6289 024300 005737 005524      1$:   JSR      PC,PREPKB :PREPARE FOR KBD INPUT
6290 024304 001775      TST     INTCHR      :CHECK FOR TTY INPUT
6291 024306 013701 005524      BEQ     1$         :BR IF NO INPUT YET
6292 024312 022701 000022      MOV     INTCHR,R1   :GET INPUT CHAR INTO R1
6293 024316 001721      CMP     #022,R1    :SEE IF (↑R) TYPED
6294 024320 022701 000115      BEQ     GIVEID     :BR IF (↑R) TO RESTART
6295 024324 001002      CMP     #'M,R1     :IS IT MANUAL MODE ?
6296 024326 000137 024352      BNE     2$         :BR IF NOT MANUAL
6297 024332 022701 000101      JMP     MANUAL     :JUMP TO MANUAL MODE ROUTINE
6298 024336 001002      CMP     #'A,R1     :IS IT AUTO MODE ?
6299 024340 000137 025300      BNE     3$         :BR IF NOT AUTO
6300 024344 004737 027270      JMP     AUTO       :JUMP TO AUTO MODE ROUTINE
6301 024350 000747      JSR     PC,ECOBAD  :ECHO BAD INPUT
6302
6303
6304
6305
6306
6307 024352 104400 011574      :MANUAL SELECT MODE ROUTINE
6308 024352 104400 011627      MANUAL: TYPE    ,TYPMAN :TYPE "MANUAL SELECT MODE"
6309 024356 104400 011627      ASKDRV: TYPE    ,ENTDRV  :ASK FOR DRIVE NUMBER
6310 024362 004737 027250      JSR     PC,PREPKB  :PREPARE FOR KBD INPUT
6311 024366 005737 005524      1$:   TST     INTCHR  :SEE IF ANY INPUT
6312 024372 001775      BEQ     1$         :BR IF NONE TO CHECK AGAIN
6313 024374 013701 005524      MOV     INTCHR,R1  :GET CHARACTER INTO R1
6314 024400 020127 000022      CMP     R1,#022    :TEST FOR (↑R) TYPED
6315 024404 001666      BEQ     GIVEID     :BR IF (↑R) TO RESTART

```

```

6315 024406 020127 000060      CMP      R1,#'0      ;COMPARE TO ASCII 0
6316 024412 002410      BLT      2$          ;BR IF <0 (BAD INPUT)
6317 024414 020127 000067      CMP      R1,#'7      ;COMPARE TO ASCII 7
6318 024420 003005      BGT      2$          ;BR IF >7 (BAD INPUT)
6319 024422 042701 000060      BIC      #'0,R1      ;STRIP ASCII BITS
6320 024426 010137 005502      MOV      R1,DRIVE    ;STORE DRIVE NUMBER
6321 024432 000403          BR       3$          ;PROCEED WITH DESIRED DRIVE
6322 024434 004737 027270      2$:     JSR      PC,ECOBAD ;ECHO BAD INPUT
6323 024440 000746          BR       ASKDRV     ;GO BACK TO ASK AGAIN
6324 024442 004737 030726      3$:     JSR      PC,INITSS ;INITIALIZE SUBSYSTEM
6325          ;CHECK DESIRED DRIVE FOR PROPER STATUS
6326 024446 004737 031460      JSR      PC,CHKDRV   ;CHECK DRIVE FOR ON-LINE,READY,WRITE LOCK
6327 024452 024356          ASKDRV          ;ADDRESS OF ERROR RETURN FOR CHKDRV
6328 024454 004737 033524      JSR      PC,DRVSR    ;TYPE "DRIVE SER. NO. XXX"
6329          ;SET UP DESIRED SUB-MODE OF OPERATION
6330          SUBMOD:
6331 024460 105037 011430      CLRB     VERIFY     ;CLEAR VERIFY MODE FLAG
6332 024464 104400 012412      TYPE     ENTMOD     ;ASK FOR ALIGN OR EXERCISE SUB-MODE
6333 024470 004737 027250      JSR      PC,PREPKB  ;PREPARE FOR KBD INPUT
6334 024474 005737 005524      1$:     TST      INTCHR ;SEE IF ANY INPUT
6335 024500 001775          BEQ      1$         ;BR IF NONE TO CHECK AGAIN
6336 024502 013701 005524      MOV      INTCHR,R1  ;GET CHAR INTO R1
6337 024506 020127 000022      CMP      R1,#022    ;SEE IF (↑R) TYPED
6338 024512 001002          BNE     2$         ;BR IF NOT (↑R)
6339 024514 000137 024162      JMP      GIVEID     ;JUMP TO RESTART
6340 024520 020127 000003      2$:     CMP      R1,#003 ;SEE IF (↑C) TYPED
6341 024524 001002          BNE     3$         ;BR IF NOT (↑C)
6342 024526 000137 024356      JMP      ASKDRV     ;JUMP TO ASK FOR DRIVE AGAIN
6343 024532 020127 000101      3$:     CMP      R1,#'A  ;SEE IF ALIGNMENT REQUESTED
6344 024536 001415          BEQ      MANAL     ;BR IF MANUAL ALIGNMENT REQUESTED
6345 024540 020127 000105      CMP      R1,#'E     ;SEE IF EXERCISES REQUESTED
6346 024544 001002          BNE     4$         ;BR IF NOT EXERCISES
6347 024546 000137 025044      JMP      MANEX     ;JUMP TO DO MANUAL EXERCISES
6348 024552 020127 000126      4$:     CMP      R1,#'V ;SEE IF VERIFY REQUESTED
6349 024556 001002          BNE     5$         ;BR IF NOT VERIFY
6350 024560 000137 025026      JMP      MANVR     ;JUMP TO VERIFY MODE
6351 024564 004737 027270      5$:     JSR      PC,ECOBAD ;ECHO BAD INPUT
6352 024570 000733          BR       SUBMOD    ;ASK FOR SUB-MODE AGAIN
6353
6354
6355
6356          ;MANUAL ALIGNMENT MODE
6357          MANAL:
6358 024572 104400 012552      TYPE     MANALN    ;REPORT MANUAL ALIGNMENT MODE
6359          ;REQUEST AND CHECK DESIRED HEAD
6360 024576 104400 012720      ASKHED: TYPE     ENTHED ;ASK FOR DESIRED HEAD
6361 024602 004737 027250      JSR      PC,PREPKB  ;PREPARE FOR KBD INPUT
6362 024606 005737 005524      1$:     TST      INTCHR ;SEE IF ANY INPUT
6363 024612 001775          BEQ      1$         ;BR IF NONE TO CHECK AGAIN
6364 024614 013701 005524      MOV      INTCHR,R1  ;GET CHAR INTO R1
6365 024620 020127 000022      CMP      R1,#022    ;SEE IF (↑R)
6366 024624 001007          BNE     2$         ;BR IF NOT (↑R)
6367 024626 112765 000113 000001      MOVB    #'RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6368 024634 004737 040716      JSR      PC,DRVCAL  ;RECALIBRATE DRIVE
6369 024640 000137 024162      JMP      GIVEID     ;JUMP TO RESTART
6370 024644 020127 000003      2$:     CMP      R1,#003 ;SEE IF (↑C)

```

```

6371 024650 001007      BNE      3$      ;BR IF NOT (1C)
6372 024652 112765 000113 000001      MOVB     #RECAL.P.CMND(R5) ;SET RECALIBRATE COMMAND
6373 024660 004737 040716      JSR     PC,DRVCAL      ;RECALIBRATE DRIVE
6374 024664 000137 024356      JMP     ASKDRV        ;JUMP TO ASK FOR DRIVE AGAIN
6375 024670 020127 000032      3$:     CMP     R1,#032    ;SEE IF (1Z)
6376 024674 001007      BNE      4$      ;BR IF NOT (1Z)
6377 024676 112765 000113 000001      MOVB     #RECAL.P.CMND(R5) ;SET RECALIBRATE COMMAND
6378 024704 004737 040716      JSR     PC,DRVCAL      ;RECALIBRATE DRIVE
6379 024710 000137 024460      JMP     SUBMOD        ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
6380 024714 020127 000060      4$:     CMP     R1,#'0    ;COMPARE TRACK TO ASCII 0
6381 024720 002410      BLT     5$      ;BR IF <0 (BAD INPUT)
6382 024722 020127 000062      CMP     R1,#'2    ;COMPARE TRACK TO ASCII 2
6383 024726 003005      BGT     5$      ;BR IF >2 (BAD INPUT)
6384 024730 042701 000060      BIC     #'0,R1    ;STRIP ASCII BITS
6385 024734 010137 005522      MOV     R1,TRACK    ;STORE TRACK (HEAD) NUMBER
6386 024740 000404      BR      6$      ;GO SELECT HEAD
6387 024742 004737 027270      5$:     JSR     PC,ECOBAD    ;ECHO BAD INPUT
6388 024746 000137 024576      JMP     ASKHED        ;ASK AGAIN FOR HEAD NUMBER
6389      ;UNLOAD HEADS,AND WHEN READY, START SPINDLE AND SEEK TO ALN CYL
6390 024752 105737 011430      6$:     TSTB     VERIFY    ;SEE IF VERIFY MODE
6391 024756 001002      BNE      8$      ;BR IF VERIFY
6392 024760 004737 031754      JSR     PC,WAIT4R    ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
6393 024764 113765 005522 000005 8$:     MOVB     TRACK,P.TRCK(R5) ;SET DESIRED HEAD NUMBER
6394 024772 004737 031720      JSR     PC,ALNSEK    ;SEEK TO ALIGNMENT CYLINDER
6395 024776 104400 012647      TYPE    HEDPOS      ;TYPE HEADS POSITIONED MSG
6396 025002 013701 005522      MOV     TRACK,R1    ;GET BINARY TRACK NO.
6397 025006 152701 000060      BISB     #'0,R1    ;CONVERT TO ASCII
6398 025012 110137 013006      MOVB     R1,HEADNO   ;GET HEAD NO. INTO MSG BUFF.
6399 025016 104400 013001      TYPE    HEDSEL      ;TYPE HEAD SELECTED MSG
6400 025022 000137 024576      JMP     ASKHED        ;GO BACK TO ASK FOR NEW HEAD SELECT
6401
6402
6403
6404      ;MANUAL SELECT VERIFY MODE
6405 025026 112737 000001 011430  MANVR:  MOVB     #'1,VERIFY    ;SET VERIFY MODE FLAG
6406 025034 104400 012612      TYPE    MANVRF      ;REPORT MANUAL VERIFY MODE
6407 025040 000137 024576      JMP     ASKHED        ;GO ASK FOR DESIRED HEAD
6408
6409
6410
6411      ;MANUAL RANDOM SEEK EXERCISE ROUTINE
6412 025044  MANEX:
6413 025044 004737 031754      JSR     PC,WAIT4R    ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
6414 025050 013701 005502      MOV     DRIVE,R1    ;GET DRIVE NUMBER
6415 025054 052701 000060      BISB     #'0,R1    ;CONVERT TO ASCII
6416 025060 110137 012544      MOVB     R1,DR1EXR   ;PUT DRIVE NUMBER INTO OUTPUT BUFFER
6417 025064 104400 012465      TYPE    MANEXR      ;TYPE RANDOM SEEKS MSG
6418 025070 012700 016514      MOV     #RNDLNG,RO  ;INITIALIZE RANDOM SEEK COUNTER
6419 025074 112765 000117 000001 8$:     MOVB     #SEEK.P.CMND(R5) ;SET SEEK COMMAND
6420 025102 004737 027250      1$:     JSR     PC,PREPKB   ;PREPARE FOR KBD INPUT
6421 025106 004737 034044      2$:     JSR     PC,RNDADR   ;SELECT RANDOM CYLINDER ADDRESS
6422 025112 013765 005506 000002 8$:     MOV     CYLNDR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
6423 025120 004737 040716      JSR     PC,DRVCAL    ;DO A RANDOM SEEK
6424      ;CHECK FOR TTY INPUT DURING EXERCISES
6425 025124 005737 005524      TST     INTCHR      ;SEE IF ANY INPUT
6426 025130 001016      BNE      3$      ;BR IF A CHAR WAS TYPED

```

6427	025132	005300		DEC	RO	: DECREMENT SEEK COUNTER
6428	025134	001364		BNE	2\$: BR IF NOT DONE SEEKING YET
6429	025136	042777	000100 154000	BIC	#BIT6,2\$TKS	: DISABLE KBD INTERRUPT
6430	025144	112765	000113 000001	MOVB	#RECAL.P.CMND(R5)	: SET RECALIBRATE COMMAND
6431	025152	004737	040716	JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6432	025156	104400	001310	TYPE	\$BELL	: RING BELL TO SIGNIFY END OF EXERCISES
6433	025162	000137	024460	JMP	SUBMOD	: GO ASK FOR NEW MANUAL SUB-MODE
6434	025166	013701	005524	3\$: MOV	INTCHR,R1	: GET CHARACTER INTO R1
6435	025172	020127	000022	CMP	R1,#022	: SEE IF (↑R)
6436	025176	001007		BNE	4\$: BR IF NOT (↑R)
6437	025200	112765	000113 000001	MOVB	#RECAL.P.CMND(R5)	: SET RECALIBRATE COMMAND
6438	025206	004737	040716	JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6439	025212	000137	024162	JMP	GIVEID	: JUMP TO RESTART
6440	025216	020127	000003	4\$: CMP	R1,#003	: SEE IF (↑C)
6441	025222	001007		BNE	5\$: BR IF NOT (↑C)
6442	025224	112765	000113 000001	MOVB	#RECAL.P.CMND(R5)	: SET RECALIBRATE COMMAND
6443	025232	004737	040716	JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6444	025236	000137	024356	JMP	ASKDRV	: JUMP TO ASK FOR NEW DRIVE NUMBER
6445	025242	020127	000032	5\$: CMP	R1,#032	: SEE IF (↑Z)
6446	025246	001007		BNE	6\$: BR IF NOT (↑Z)
6447	025250	112765	000113 000001	MOVB	#RECAL.P.CMND(R5)	: SET RECALIBRATE COMMAND
6448	025256	004737	040716	JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6449	025262	000137	024460	JMP	SUBMOD	: JUMP TO ASK FOR NEW MANUAL SUB-MODE
6450	025266	004737	027270	6\$: JSR	PC,ECOBAD	: ECHO BAD INPUT
6451	025272	104400	012465	TYPE	\$MANEXR	: VERIFY STILL IN MANUAL EXERCISES
6452	025276	000701		BR	1\$: BR TO CONTINUE EXERCISES

6453						
6454						
6455						
6456						
6457	025300					: AUTO SELECT MODE ROUTINE
6458	025300	004737	030726	JSR	PC,INITSS	: INITIALIZE SUBSYSTEM
6459	025304	104400	013023	TYPE	\$TYPALT	: TYPE "AUTO SELECT MODE"
6460	025310	105037	011430	1\$: CLRB	\$VERIFY	: CLEAR VERIFY MODE FLAG
6461	025314	104400	012412	TYPE	\$ENTMOD	: ASK FOR ALIGN,VERIFY, OR EXERCISE
6462	025320	004737	027250	JSR	PC,PREPKB	: PREPARE FOR KEYBOARD INPUT
6463	025324	005737	005524	2\$: TST	INTCHR	: SEE IF ANY INPUT
6464	025330	001775		BEQ	2\$: BR IF NONE, TO CHECK AGAIN
6465	025332	013701	005524	MOV	INTCHR,R1	: GET CHAR INTO R1
6466	025336	020127	000022	CMP	R1,#022	: SEE IF (↑R) TYPED
6467	025342	001002		BNE	3\$: BR IF NOT (↑R)
6468	025344	000137	024162	JMP	GIVEID	: JUMP TO RESTART
6469	025350	020127	000101	3\$: CMP	R1,#'A	: SEE IF AUTO ALIGNMENT REQUESTED
6470	025354	001417		BEQ	AUTAL	: BR IF AUTO ALIGNMENT REQUESTED
6471	025356	020127	000105	CMP	R1,#'E	: SEE IF AUTO EXERCISES REQUESTED
6472	025362	001002		BNE	4\$: BR IF EXERCISES NOT REQUESTED
6473	025364	000137	026410	JMP	AUTEX	: JUMP TO DO AUTO EXERCISES
6474	025370	020127	000126	4\$: CMP	R1,#'V	: SEE IF VERIFY REQUESTED
6475	025374	001004		BNE	6\$: BR IF NOT VERIFY
6476	025376	112737	000001 011430	MOVB	#1,VERIFY	: SET VERIFY MODE FLAG
6477	025404	000403		BR	AUTAL	: PROCEED IN VERIFY MODE
6478	025406	004737	027270	6\$: JSR	PC,ECOBAD	: ECHO BAD INPUT
6479	025412	000736		BR	1\$: BR TO ASK FOR A OR E AGAIN
6480						
6481						
6482						

```

6483                                     :AUTO ALIGNMENT MODE
6484 025414                                AUTAL:
6485 025414 105737 011430                TSTB   VERIFY           ;SEE IF AUTO VERIFY MODE
6486 025420 001403                        BEQ    3$              ;BR IF NOT
6487 025422 104400 013112                TYPE   ,AUTVRF        ;REPORT AUTO VERIFY MODE
6488 025426 000402                        BR     4$              ;
6489 025430 104400 013054                3$:   TYPE   ,AUTALN   ;TYPE ALIGNMENT MESSAGE
6490 025434 004737 027250                4$:   JSR    PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
6491 025440 005037 003014                CLR    T.MR2          ;CLEAR MR2 STORAGE WORD
6492 025444 005037 005526                CLR    SELECT         ;INITIALIZE AUTO-SELECTED DRIVE INDICATOR
6493 025450 012737 000377 005530        MOV    #377,ONLINE    ;INIT. OLD ONLINE DRIVE INDICATOR
6494 025456 012737 000377 005532        MOV    #377,NEWON     ;INIT. NEW ONLINE DRIVE INDICATOR
6495 025464 012737 000377 005534        MOV    #377,SCRACH    ;INITIALIZE LAST DRIVE INDICATOR
6496 025472 005737 005524                DRVLUP: TST   INTCHR   ;SEE IF ANY KBD INPUT
6497 025476 001445                        BEQ    6$              ;BR IF NO INPUT
6498 025500 113765 005534 000000        MOVB   SCRACH,P.DRVN(R5) ;DRIVE NO. FOR POSSIBLE RECALIBRATE
6499 025506 123727 005524 000032        CMPB   INTCHR,#032    ;SEE IF (↑Z) TYPED
6500 025514 001013                        BNE    2$              ;BR IF NOT (↑Z)
6501 025516 022737 000377 005534        CMP    #377,SCRACH    ;SEE IF ANY DRIVE SELECTED (ET
6502 025524 001405                        BEQ    1$              ;BR IF NONE
6503 025526 112765 000113 000031        MOVB   #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6504 025534 004737 040716                JSR    PC,DRVCAL      ;RECALIBRATE DRIVE
6505 025540 000137 025300                1$:   JMP    AUTO        ;RESTART AUTO MODE
6506 025544 123727 005524 000022        2$:   CMPB   INTCHR,#022 ;SEE IF (↑R) TYPED
6507 025552 001013                        BNE    4$              ;BR IF NOT (↑R)
6508 025554 022737 000377 005534        CMP    #377,SCRACH    ;SEE IF ANY DRIVE SELECTED YET
6509 025562 001405                        BEQ    3$              ;BR IF NONE
6510 025564 112765 000113 000001        MOVB   #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6511 025572 004737 040716                JSR    PC,DRVCAL      ;RECALIBRATE DRIVE
6512 025576 000137 024162                3$:   JMP    GIVEID       ;RESTART ALIGNMENT AID
6513 025602 004737 027270                4$:   JSR    PC,ECOBAD    ;ECHO BAD INPUT
6514 025606 004737 027250                JSR    PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT
6515 025612 005037 005502                6$:   CLR    DRIVE        ;ZERO THE DRIVE NUMBER
6516 025616 012700 000001                MOV    #1,R0          ;INITIALIZE DRIVE BIT MASK
6517 025622 012737 027032 003036        8$:   MOV    #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
6518 025630 050037 005532                BIS    R0,NEWON       ;SET ON-LINE BIT FOR THIS DRIVE
6519                                     ;SELECT CURRENT DRIVE, CHECK FOR NON-EXISTENT DRIVE (NED) INDICATION
6520 025634 113765 005502 000000        MOVB   DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
6521 025642 112765 000101 000001        MOVB   #SELDIV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
6522 025650 004737 040716                JSR    PC,DRVCAL      ;SELECT THIS DRIVE
6523 025654 022737 000007 005502        CMP    #7,DRIVE       ;SEE IF WE JUST CHECKED DRIVE 7
6524 025662 001405                        BEQ    10$             ;BR IF IT WAS DRIVE 7
6525 025664 005237 005502                INC    DRIVE          ;ADD 1 TO DRIVE NUMBER
6526 025670 000241                        CLC                                ;CLEAR CARRY BEFORE ROTATE
6527 025672 006100                        ROL                                ;SHIFT BIT POINTER
6528 025674 000752                        BR     8$              ;BR TO SELECT NEXT DRIVE
6529 025676 013701 005532                10$:  MOV    NEWON,R1
6530                                     ;***** TO BE REMOVED *****
6531 025702 023737 005532 005530        CMP    NEWON,ONLINE
6532 025710 001670                        BEQ    DRVLUP
6533 025712 012703 000001                MOV    #1,R3          ;NUMBER OF 200 MILLI-SEC STALLS
6534 025716 012700 177777                LOOP1: MOV    #177777,R0
6535 025722 005300                        LOOP2: DEC    R0
6536 025724 005700                        TST   R0
6537 025726 001375                        BNE   LOOP2
6538 025730 005303                        DEC   R3
    
```



```

6539 025732 001371          BNE      LOOP1
6540                      ;*****
6541 025734 043701 005530  BIC      ONLINE,R1      ;GET CHANGED ONLINE BITS
6542 025740 013737 005532 005530  MOV      NEWON,ONLINE   ;UPDATE ONLINE DRIVE BITS
6543 025746 005701          TST      R1              ;SEE IF ANY DRIVE JUST WENT ON-LINE
6544 025750 001650          BEQ      DRVLUP          ;BR IF NO DRIVE JUST WENT ONLINE
6545                      ;SERVICE THE DRIVE WHICH WAS JUST SELECTED
6546 025752 012737 042410 003036  MOV      #ERRHDL,A.ABNL ;RESTORE USUAL ERROR HANDLER ADDRESS
6547 025760 005037 005502          CLR      DRIVE          ;INITIALIZE DRIVE NO.
6548 025764 010103          MOV      R1,R3          ;COPY NEW SELECTED DRIVE BITS
6549 025766 012700 000001          MOV      #1,R0          ;INITIALIZE BIT POINTER
6550 025772 030003 12$:      BIT      R0,R3          ;SEE IF THIS BIT IS SET
6551 025774 001005          BNE      14$           ;BR IF THIS BIT SET
6552 025776 005237 005502          INC      DRIVE          ;INCREMENT DRIVE NO.
6553 026002 000241          CLC                      ;CLEAR CARRY BEFORE ROTATE
6554 026004 006100          ROL      R0              ;SHIFT BIT POINTER
6555 026006 000771          BR      12$           ;TRY AGAIN
6556 026010 040003 14$:      BIC      R0,R3          ;CLEAR OUT THIS BIT
6557 026012 001415          BEQ      16$           ;BR IF ONLY ONE DRIVE SELECTED
6558                      ;ERROR - MORE THAN ONE DRIVE SELECTED SIMULTANEOUSLY
6559 026014 104400 012313          TYPE     ,MULSEL       ;REPORT ERROR
6560 026020 042762 000100 000000  BIC      #IE,RKCS1(R2) ;INHIBIT RK06 INTERRUPT
6561 026026 000000          HALT                    ;HALT FOR MANUAL INTERVENTION
6562                      ;PRESS 'CONT' TO PROCEED
6563 025030 112765 000177 000001  MOVVB   #SUBCLR.P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6564 025036 004737 040716          JSR     PC,DRVCAL      ;CLEAR THE SUBSYSTEM
6565 026042 000137 025414          JMP     AUTAL          ;RESTART AUTO ALIGNMENT
6566                      ;CONTINUE WITH SELECTED DRIVE
6567 026046 020137 005526 16$:      CMP      R1,SELECT     ;SEE IF STILL ALIGNING SAME DRIVE
6568 026052 001506          BEQ      26$           ;BR IF SAME DRIVE
6569                      ;PROCEED WITH NEWLY-SELECTED DRIVE
6570 026054 022737 000377 005534  CMP      #377,SCRACH    ;SEE IF THIS IS FIRST DRIVE SELECTED
6571 026062 001421          BEQ      23$           ;BR IF FIRST DRIVE
6572 026064 113765 005534 000000  MOVVB   SCRACH,P.DRVN(R5) ;GET LAST DRIVE NUMBER
6573 026072 112765 000141 000001  MOVVB   #RDSTAT.P.CMND(R5) ;SET READ STATUS COMMAND
6574 026100 004737 040716          JSR     PC,DRVCAL      ;READ STATUS OF PREVIOUS DRIVE
6575 026104 032765 000040 000044  BIT      #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLOADED
6576 026112 001405          BEQ      23$           ;BR IF HEADS NOT UNLOADED
6577 026114 112765 000111 000001  MOVVB   #SRTSPL.P.CMND(R5) ;SET START SPINDLE COMMAND
6578 026122 004737 040716          JSR     PC,DRVCAL      ;START SPINDLE ON PREVIOUS DRIVE
6579 026126 113737 005502 005534 23$:      MOVVB   DRIVE,SCRACH    ;STORE DRIVE NO. FOR LATER CLEANUP STRT SPL
6580 026134 010137 005526          MOV      R1,SELECT     ;UPDATE SELECTED DRIVE NUMBER
6581 026140 013701 005502          MOV      DRIVE,R1      ;GET DRIVE NUMBER
6582 026144 052701 000060          BIS      #'0,R1        ;CONVERT TO ASCII
6583 026150 110137 013154          MOVVB   R1,DRVSEL      ;GET DRIVE NUMBER INTO OUTPUT BUFFER
6584 026154 104400 013145          TYPE     ,DRSEL        ;TYPE "DRIVE X SELECTED"
6585 026160 004737 031460          JSR     PC,CHKDRV      ;CHECK DRIVE FOR RDY, WRITE PROT.
6586 026164 025414          AUTAL                    ;ERROR RETURN ADDRESS
6587 026166 112737 000377 011427  MOVVB   #377,UNLOD      ;SET DRIVE UNLOADED INDICATOR
6588 026174 012737 000002 005522  MOV      #2,TRACK       ;SET TRACK = 2
6589 026202 113765 005502 000000  MOVVB   DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
6590 026210 105737 011430 24$:      TSTB   VERIFY         ;SEE IF AUTO VERIFY
6591 026214 001016          BNE      32$           ;SKIP UNLOAD IF VERIFY
6592 026216 112765 000107 000001  MOVVB   #UNLOAD.P.CMND(R5) ;SET UNLOAD COMMAND
6593 026224 004737 040716          JSR     PC,DRVCAL      ;UNLOAD THE DRIVE
6594 026230 112765 000141 000001  MOVVB   #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND

```

```

6595 026236 004737 040716 25$: JSR PC,DRVCAL ;READ STATUS OF DRIVE
6596 026242 032765 000040 000044 BIT #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLOADED YET
6597 026250 001772 BEQ 25$ ;BR IF NOT YET
6598 026252 112765 000177 000001 32$: MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6599 026260 004737 040716 JSR PC,DRVCAL ;CLEAR THE S.S.
6600 026264 000137 025472 JMP DRVLUP ;GO BACK TO SCAN DRIVES AGAIN
6601 ;PROCEED WITH SAME DRIVE
6602 026270 113765 005502 000000 26$: MOVB DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6603 026276 105137 011427 COMB UNLOD ;COMPLEMENT UNLOAD INDICATOR
6604 026302 001342 BNE 24$ ;BR IF DRIVE SHOULD BE UNLOADED
6605 026304 105737 011430 TSTB VERIFY ;SEE IF AUTO VERIFY
6606 026310 001005 BNE 33$ ;SKIP START SPL IF VERIFY
6607 026312 112765 000111 000001 MOVB #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
6608 026320 004737 040716 JSR PC,DRVCAL ;START SPINDLE ON THIS DRIVE
6609 026324 022737 000002 005522 33$: CMP #2,TRACK ;SEE IF LAST TRACK WAS 2
6610 026332 001003 BNE 28$ ;BR IF IT WAS NOT 2
6611 026334 005037 005522 CLR TRACK ;SET TRACK = 0
6612 026340 000402 BR 30$ ;GO SEEK TO ALIGNMENT CYLINDER
6613 026342 005237 005522 28$: INC TRACK ;INCREMENT TRACK NUMBER
6614 026346 113765 005522 000005 30$: MOVB TRACK,P.TRCK(R5) ;SET TRACK NO.
6615 026354 004737 031720 JSR PC,ALNSEK ;SEEK IN INCREMENTS TO ALIGN. CYL
6616 026360 104400 012647 TYPE #HEDPOS ;TYPE HEADS POSITIONED MSG
6617 026364 013701 005522 MOV #TRACK,R1 ;GET CURRENT TRACK NO.
6618 026370 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
6619 026374 110137 013006 MOVB R1,HEADNO ;GET TRACK INTO MSG BUFFER
6620 026400 104400 013001 TYPE #HEDSEL ;TYPE "HEAD X SELECTED"
6621 026404 000137 025472 JMP DRVLUP ;GO BACK TO SCAN DRIVES AGAIN
6622
6623
6624 ;AUTO RANDOM SEEK EXERCISE ROUTINE
6625 AUTEX:
6626 026410 004737 030726 JSR PC,INITSS ;INITIALIZE SUBSYSTEM
6627 026414 004737 027250 JSR PC,PREPKB ;PREPARE FOR KBD INPUT
6628 026420 104400 013172 TYPE #AUTEXR ;ASK FOR SHORT OR LONG SEEK EXERCISES
6629 026424 005737 005524 1$: TST INTCHR ;SEE IF ANY INPUT
6630 026430 001775 BEQ 1$ ;BR IF NO INPUT
6631 026432 023727 005524 000022 CMP INTCHR,#022 ;SEE IF (↑R) TYPED
6632 026440 001002 BNE 2$ ;BR IF NOT (↑R)
6633 026442 000137 024162 JMP GIVEID ;JUMP TO RESTART ALIGNMENT AID
6634 026446 023727 005524 000032 2$: CMP INTCHR,#032 ;SEE IF (↑Z) TYPED
6635 026454 001002 BNE 3$ ;BR IF NOT (↑Z)
6636 026456 000137 025300 JMP AUTO ;JUMP TO RESTART AUTO MODE
6637 026462 023727 005524 000123 3$: CMP INTCHR,#'S ;SEE IF SHORT EXERCISES DESIRED
6638 026470 001004 BNE 4$ ;BR IF NOT SHORT
6639 026472 012737 002734 005534 MOV #RNDSH,SCRACH ;SET 1 MINUTE SEEK COUNT
6640 026500 000413 BR 6$ ;PROCEED WITH SHORT EXERCISES
6641 026502 023727 005524 000114 4$: CMP INTCHR,#'L ;SEE IF LONG EXERCISES DESIRED
6642 026510 001004 BNE 5$ ;BR IF NOT LONG
6643 026512 012737 016514 005534 MOV #RNDLNG,SCRACH ;SET 5 MINUTE SEEK COUNT
6644 026520 000403 BR 6$ ;PROCEED WITH LONG EXERCISES
6645 026522 004737 027270 5$: JSR PC,ECOBAD ;ECHO BAD INPUT
6646 026526 000730 BR AUTEX ;BR TO ASK AGAIN
6647 026530 005037 005502 6$: CLR DRIVE ;SET DRIVE = 0
6648 ;SELECT CURRENT DRIVE AND CHECK FOR NON-EXISTENT DRIVE INDICATION
6649 026534 012737 027032 003036 8$: MOV #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
6650 026542 113765 005502 000000 MOVB DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
    
```



```

6651 026550 112765 000101 000001      MOVVB  #SELDIV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
6652 026556 012737 000377 005532      MOV    #377,NEWON      ;INITIALIZE ON-LINE INDICATOR
6653 026564 004737 040716                JSR    PC,DRVVAL      ;SELECT THIS DRIVE
6654 026570 012737 042410 003036      MOV    #ERRHDL,A.ABNL ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
6655 026576 022737 000377 005532      CMP    #377,NEWON     ;SEE IF NED INDICATION ON THIS DRIVE
6656 026604 001075                BNE    18$            ;BR IF NED, TO SKIP THIS DRIVE
6657                                ;CHECK DRIVE FOR RDY, WRITE PROTECT
6658 026606 004737 031460                JSR    PC,CHKDRV     ;CHECK THIS DRIVE
6659 026612 026410                AJTEX                    ;ERROR RETURN ADDRESS FOR CHKDRV
6660                                ;PROCEED WITH SEEK EXERCISES ON THIS DRIVE
6661 026614 013701 005502      MOV    DRIVE,R1      ;GET DRIVE NUMBER
6662 026620 052701 000060      BIS    #'0,R1        ;CONVERT TO ASCII
6663 026624 110137 013304      MOVVB  R1,AUTODR     ;GET DRIVE NUMBER INTO OUTPUT BUFFER
6664 026630 104400 013263      TYPE  ,AUTOEX       ;TYPE "EXERCISING DRIVE X"
6665 026634 112765 000117 000001      MOVVB  #SEEK,P.CMND(R5) ;SET SEEK COMMAND
6666 026642 013700 005534      MOV    SCRACH,RO     ;SEEK COUNT
6667 026646 004737 027250      10$: JSR    PC,PREPKB   ;PREPARE FOR POSSIBLE KBD INPUT
6668 026652 004737 034044      JSR    PC,RNDADR     ;SELECT RANDOM CYLINDER ADDRESS
6669 026656 013765 005506 000002      MOV    CYLNR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
6670 026664 004737 040716      JSR    PC,DRVVAL     ;DO A RANDOM SEEK
6671 026670 005737 005524      TST   INTCHR         ;SEE IF ANY KBD INPUT
6672 026674 001432                BEQ    16$            ;BR IF NO INPUT
6673 026676 023727 005524 000022      CMP    INTCHR,#022   ;SEE IF (↑R) TYPED
6674 026704 001007                BNE    12$            ;BR IF NOT (↑R)
6675 026706 112765 000113 000001      MOVVB  #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6676 026714 004737 040716      JSR    PC,DRVVAL     ;RECALIBRATE DRIVE
6677 026720 000137 024162      JMP    GIVEID        ;JUMP TO RESTART ALIGNMENT AID
6678 026724 023727 005524 000032 12$: CMP    INTCHR,#032   ;SEE IF (↑Z) TYPED
6679 026732 001007                BNE    14$            ;BR IF NOT (↑Z)
6680 026734 112765 000113 000001      MOVVB  #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6681 026742 004737 040716      JSR    PC,DRVVAL     ;RECALIBRATE DRIVE
6682 026746 000137 025300      JMP    AUTO          ;JUMP TO RESTART AUTO MODE
6683 026752 004737 027270      14$: JSR    PC,EC,BAD   ;ECHO BAD INPUT
6684 026756 004737 027250      JSR    PC,PREPKB     ;ENABLE KBD INPUT AGAIN
6685 026762 005300      16$: DEC    RO        ;DECREMENT SEEK COUNT
6686 026764 001330                BNE    10$            ;BR IF NOT DONE WITH THIS DRIVE
6687 026766 112765 000113 000001      MOVVB  #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6688 026774 004737 040716      JSR    PC,DRVVAL     ;RECALIBRATE DRIVE
6689 027000 005237 005502      18$: INC    DRIVE     ;INCREMENT DRIVE NUMBER
6690 027004 022737 000010 005502      CMP    #10,DRIVE    ;SEE IF DONE WITH ALL DRIVES
6691 027012 001250                BNE    8$             ;BR IF MORE DRIVES TO EXERCISE YET
6692 027014 042777 000100 152122      BIC    #BIT6,#STKS  ;DISABLE KBD INTERRUPT
6693 027022 104400 001310      TYPE  ,SBELL        ;RING BELL AT END OF AUTO-EXERCISES
6694 027026 000137 025300      JMP    AUTO          ;RESTART AUTO MODE
6695
6696
6697                                ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
6698                                ; WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
6699                                ; (NED = NON-EXISTENT DRIVE)
6700 027032 032765 010000 000020 NEDHDL: BIT  #NED,P.CS2(R5) ;SEE IF NED ON DRIVE SELECT
6701 027040 001002                BNE    1$             ;BR IF NED SET
6702 027042 000137 042410      1$: JMP    ERRHDL      ;GO HANDLE OTHER ERROR
6703 027046 010046                MOV    RO,-(SP)      ;SAVE RO,R1
6704 027050 010146                MOV    R1,-(SP)
6705 027052 012700 000001      MOV    #1,RO        ;SET BIT POINTER
6706 027056 005001      CLR    R1           ;CLEAR COUNTER

```

```

6707 027060 120137 005502 2$: CMPB R1,DRIVE ;SEE IF R1 = CURRENT DRIVE NUMBER
6708 027064 001403 BEQ 3$ ;BR IF =
6709 027066 005201 INC R1 ;INCREMENT COUNTER
6710 027070 006300 ASL R0 ;SHIFT BIT POINTER
6711 027072 000772 BR 2$ ;TRY AGAIN
6712 027074 040037 005532 3$: BIC R0,NEWON ;CLEAR ONLINE BIT FOR THIS DRIVE
6713 027100 012601 MOV (SP)+,R1 ;RESTORE R0,R1
6714 027102 012600 MOV (SP)+,R0
6715 027104 000137 044624 JMP RETNML ;TAKE NORMAL RETURN
6716
6717
6718
6719
6720
6721

```

```

;*****
;SBTTL KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
;*****

```

```

6722 027110 010146 KBDHDL: MOV R1, -(SP) ;SAVE R1 ON STACK
6723 027112 017701 152030 MOV @STKB,R1 ;READ A CHAR FROM KBD BUFFER
6724 027116 042701 177600 BIC #177600,R1 ;MASK OUT UNUSED BITS
6725 027122 120127 000172 CMPB R1,#172 ;SEE IF LOWER CASE TYPED
6726 027126 003005 BGT 20$ ;BR IF NOT
6727 027130 120127 000141 CMPB R1,#141
6728 027134 002402 BLT 20$ ;BR IF NOT
6729 027136 042701 000040 BIC #BITS,R1 ;MAKE IT UPPER CASE
6730 027142 010137 005524 20$: MOV R1,INTCHR ;SAVE INPUT CHARACTER
6731 027146 122701 000003 CMPB #003,R1 ;SEE IF (↑C) TYPED
6732 027152 001007 BNE 2$ ;BR IF NOT (↑C)
6733 027154 104400 013354 TYPE ,CNTRLC ;ECHO (↑C)
6734 027160 042777 000100 151756 1$: BIC #BIT6,@STKS ;CLEAR KBD INTERRUPT ENABLE BIT
6735 027166 012601 MOV (SP)+,R1 ;RESTORE R1
6736 027170 000002 RTI ;RETURN
6737 027172 122701 000032 2$: CMPB #032,R1 ;SEE IF (↑Z) TYPED
6738 027176 001003 BNE 3$ ;BR IF NOT (↑Z)
6739 027200 104400 013361 TYPE ,CNTRLZ ;ECHO (↑Z)
6740 027204 000765 BR 1$ ;RETURN
6741 027206 122701 000022 3$: CMPB #022,R1 ;SEE IF (↑R) TYPED
6742 027212 001003 BNE 4$ ;BR IF NOT (↑R)
6743 027214 104400 013366 TYPE ,CNTRLR ;ECHO (↑R)
6744 027220 000757 BR 1$ ;RETURN
6745 027222 122701 000007 4$: CMPB #007,R1 ;SEE IF (↑G) TYPED
6746 027226 001003 BNE 5$ ;BR IF NOT (↑G)
6747 027230 104400 013400 TYPE ,CNTRLG ;ECHO (↑G)
6748 027234 000751 BR 1$ ;RETURN
6749 027236 104400 005524 5$: TYPE ,INTCHR ;ECHO INPUT
6750 027242 104400 001315 TYPE ,$CRLF ;DO <CR> AND <LF>
6751 027246 000744 BR 1$ ;RETURN
6752
6753
6754
6755
6756
6757
6758
6759
6760
6761
6762

```

```

;*****
;SBTTL PREPKB - PREPARE FOR KEYBOARD INPUT
;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
;*ENABLES KBD INTERRUPT.
;* CALL:
;* JSR PC,PREPKB

```

```

6763      ;*****
6764
6765 027250 005077 151672      PREPKB: CLR      Q$TKB      ;CLEAR KBD BUFFER AND DONE BIT
6766 027254 005037 005524      CLR      INTCHR     ;CLEAR TTY INPUT WORD
6767 027260 052777 000100 151656  BIS      #BIT6,Q$TKS ;ENABLE KBD INTERRUPT
6768 027266 000207      RTS      PC          ;RETURN
6769
6770
6771
6772      ;*****
6773      ;SBTTL  ECOBAD - ECHO BAD TTY INPUT
6774      ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
6775      ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
6776      ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
6777      ;*AND <LF> ARE DONE.
6778      ;* CALL -      JSR      PC,ECOBAD
6779      ;*****
6780
6781 027270 104400 005524      ECOBAD: TYPE     ,INTCHR   ;ECHO BAD CHARACTER
6782 027274 104400 001314      TYPE     $QUES     ;TYPE (?) AND <CR>, <LF>
6783 027300 000207      RTS      PC          ;RETURN
6784
6785
6786
6787      ;*****
6788      ;*SIZMEM - SIZE MEMORY, SET LIMITS
6789      ;*THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE, AND
6790      ;*IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
6791      ;*IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D),
6792      ;*AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70).
6793      ;*****
6794 027302 104406      SIZMEM: SAVREG    ;SAVE R0-R5
6795 027304 012737 000200 056216  MOV      #200,$KT11 ;SET MEM MGT KEY FOR $SIZE
6796 027312 004737 056132      JSR      PC,$SIZE  ;SIZE MEMORY
6797 027316 005737 056216      TST      $KT11     ;SEE IF MEM MGT PRESENT
6798 027322 100406      BMI      8$        ;BR IF MEM MGT PRESENT
6799 027324 013737 056462 005574  MOV      $LSTAD,MAHILM ;SET MEM LIMIT LO BITS
6800 027332 005037 005576      CLR      MAHILM+2  ;CLEAR MEM LIMIT HI BITS
6801 027336 000427      BR       16$       ;GO TYPE LAST ADDRESS
6802
6803 027340 013700 056464      ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
6804 027344 005001      8$:  MOV      $LSTBK,R0 ;LO BITS
6805 027346 012702 000006      CLR      R1         ;HI BITS
6806 027352 000241      MOV      #6,R2     ;SET LOOP COUNT = 6
6807 027354 006100      12$: CLC          ;ROTATE LOOP
6808 027356 006101      ROL      R0
6809 027360 005302      ROL      R1
6810 027362 001373      DEC      R2
6811
6812 027364 063700 056462      ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
6813 027370 005501      ADD      $LSTAD,R0 ;ADD LO BITS
6814 027372 010037 005574      ADC      R1         ;HI BITS
6815 027376 010137 005576      MOV      R0,MAHILM ;SET LO BITS OF MEM LIMIT
6816 027402 042701 000003      MOV      R1,MAHILM+2 ;SET HI BITS OF MEM LIMIT
6817 027406 001403      BIC      #3,R1     ;CLEAR ADRS BITS 16,17
6818 027410 112737 000001 003134  BEQ      16$       ;BR IF NOT 22-BIT ADDRESSES
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839
6840
6841
6842
6843
6844
6845
6846
6847
6848
6849
6850
6851
6852
6853
6854
6855
6856
6857
6858
6859
6860
6861
6862
6863
6864
6865
6866
6867
6868
6869
6870
6871
6872
6873
6874
6875
6876
6877
6878
6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912
6913
6914
6915
6916
6917
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943
6944
6945
6946
6947
6948
6949
6950
6951
6952
6953
6954
6955
6956
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967
6968
6969
6970
6971
6972
6973
6974
6975
6976
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991
6992
6993
6994
6995
6996
6997
6998
6999
7000

```

6819
6820 027416 104400 007125
6821 027422 012746 005574
6822 027426 004737 053674
6823 027432 004737 054210
6824 027436 104400 001315
6825 027442 104400 001315
6826 027446 104407
6827 027450 000207

```
;TYPE "LAST PHYS MEM ADR = XXXXXXXX"  
16$: TYPE ,LSTMEM ;TYPE "LAST PHYS MEM ADR ="  
MOV #MAHILM,-(SP) ;PUT POINTER ON STACK  
JSR PC,2#$DB20 ;CONVERT BINARY TO OCTAL ASCII  
JSR PC,2#$SUPRS ;TYPE "XXXXXXXXY"  
TYPE ,$CRLF ;TYPE <CR>,<LF>  
TYPE , $CRLF  
RESREG ;RESTORE R0-R5  
RTS PC ;RETURN
```

6828
6829
6830
6831
6832
6833
6834
6835
6836

```
*****  
* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD  
*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE  
*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON  
*TOP OF STACK.  
*****
```

6837 027452 016646 000002
6838 027456 104402
6839 027460 006
6840 027461 000
6841 027462 104400 007333
6842 027466 004737 032214
6843 027472 027524
6844 027474 027524
6845 027476 027524
6846 027500 005700
6847 027502 001001
6848 027504 000207
6849 027506 020027 000006
6850 027512 003407
6851 027514 104400 005264
6852 027520 104400 001314
6853 027524 162716 000010
6854 027530 000207
6855 027532 012746 005264
6856 027536 004737 052736
6857 027542 027514
6858 027544 012600
6859 027546 005737 053070
6860 027552 001360
6861 027554 010066 000002
6862 027560 000207

```
GETPRM: MOV 2(SP),-(SP) ;GET OLD VALUE  
TYPOS ;TYPE IT  
.BYTE 6 ;SIX DIGITS  
.BYTE 0 ;SUPPRESS LEADING ZEROS  
TYPE NEWMSG ;TYPE " NEW = "  
JSR PC,RDCHRS ;READ NEW VALUE FROM KBD  
7$: ;(↑C) RETURN ADDRESS  
7$: ;(↑Z) RETURN ADDRESS  
7$: ;(↑U) OR ERROR RETURN ADDRESS  
TST R0 ;SEE IF ANY CHARS TYPED  
BNE 4$ ;BR IF YES  
RTS PC ;RETURN - OLD VALUE UNCHANGED  
4$: CMP R0,#6 ;SEE IF > 6 CHARS TYPED  
BLE 8$ ;BR IF NOT BAD  
6$: TYPE ,BUFFD ;ECHO BAD INPUT  
TYPE , $QUES  
7$: SUB #10,(SP) ;FIX ERROR RETURN PC  
RTS PC ;ERROR RETURN  
8$: MOV #BUFFD,-(SP) ;PUT POINTER TO CHARS ON STACK  
JSR PC,OCTBIN ;CONVERT DIGITS TO BINARY  
6$: ;ERROR RETURN ADDRESS  
MOV (SP)+,R0 ;GET NEW BINARY VALUE  
TST $HI OCT ;SEE IF HI BITS ARE 0  
BNE 6$ ;BR IF NOT  
MOV R0,2(SP) ;PUT NEW VALUE ON STACK  
RTS PC ;RETURN
```

6863
6864
6865
6866
6867
6868
6869
6870
6871
6872

```
*****  
;SBTTL GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION  
*THIS SUBROUTINE ALLOWS THE CONTENTS OF THE SOFTWARE SWITCH  
*REGISTER TO BE MODIFIED BY TTY INPUT. THE SUBROUTINE TYPES  
* "SWR = XXXXX NEW = " AND WAITS FOR A NEW OCTAL VALUE  
*OF UP TO SIX DIGITS TO BE TYPED.  
*****
```

6873 027562 022737 000176 001140
6874 027570 001010

```
GTSWRG: CMP #SWREG,SWR ;SEE IF SOFTWARE SWR SELECTED  
BNE 6$ ;BR IF NOT
```

```

6875 027572 013746 000176      MOV      SWREG, -(SP)      ;PUT OLD VALUE ON STACK
6876 027576 104400 007324      TYPE    SWRMSG           ;TYPE "SWR = "
6877 027602 004737 027452      JSR     PC, GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
6878 027606 012637 000176      MOV     (SP)+, SWREG     ;STORE NEW VALUE
6879 027612 000207                RTS      PC              ;RETURN

```

6\$:

```

6880
6881
6882
6883 ;*****
6884 ;*INITMM - INITIALIZE MEMORY MANAGEMENT REGISTERS
6885 ;*THIS SUBROUTINE INITIALIZES KIPAR'S AND KIPDR'S, FOR
6886 ;*CONTIGUOUS 4K PAGES, STARTING AT PHYSICAL ADDRESS 0.
6887 ;*KIPAR7 IS LOADED WITH 177600, TO PRESERVE THE I/O PAGE.
6888 ;*22-BIT MODE AND THE UNIBUS MAP ARE ENABLED, IF PRESENT,
6889 ;*BUT MEM MGT IS NOT TURNED ON IN THIS SUBROUTINE. HOWEVER,
6890 ;*MEM MGT TRAPS ARE ENABLED.

```

```

6891 ;*****
6892 ;*****
6893 027614 104406      INITMM: SAVREG           ;SAVE R0-R5
6894 027616 005001      CLR      R1             ;INIT FOR PAR LOADING
6895 027620 012702 172340      MOV     #KIPAR0, R2     ;ADDR OF FIRST PAR
6896 027624 012703 000010      MOV     #8, R3          ;LOAD 8 PAR'S AND 8 PDR'S
6897 027630 012762 077406 177740 4$: MOV     #77406, -40(R2)  ;PDR = 4K UP, READ/WRITE
6898 027636 010122      MOV     R1, (R2)+       ;LOAD A PAR
6899 027640 062701 000200      ADD     #200, R1        ;UPDATE FOR NEXT PAR
6900 027644 005303      DEC     R3             ;DECREMENT LOOP COUNTER
6901 027646 001370      BNE     4$             ;LOOP UNTIL ALL 8 ARE LOADED
6902 027650 012742 177600      MOV     #177600, -(R2)  ;SET UP KIPAR7 FOR I/O PAGE
6903 027654 105737 003134      TSTB   UBMPRS         ;SEE IF 22-BIT ADDRESSES
6904 027660 001403      BEQ     10$           ;BR IF NOT
6905 027662 012737 000060 172516      MOV     #60, @#SR3     ;ENABLE 22-BIT MODE, AND UNIBUS MAP
6906 027670 012737 001000 177572 10$: MOV     #BIT9, @#SRO   ;ENABLE KT11 TRAPS
6907 027676 104407      RESREG                    ;RESTORE R0-R5
6908 027700 000207      RTS      PC          ;RETURN

```

```

6909
6910
6911 ;*****
6912 ;*SERVICE ROUTINE FOR MEM MGT TRAPS
6913 ;*****

```

```

6914 ;*****
6915 027702 004737 042130      KTRHD: JSR     PC, REPSUP ;GATHER STATUS FOR PRINTOUT
6916 027706 013737 177572 001174      MOV     @#SRO, $REG5   ;GET SRO
6917 027714 013737 177574 001176      MOV     @#SR1, $REG6   ;GET SR1
6918 027722 013737 177576 001200      MOV     @#SR2, $REG7   ;GET SR2
6919 027730 012737 027754 000004      MOV     #8$, @#ERRVEC  ;SET TIME-OUT VECTOR
6920 027736 012737 000340 000006      MOV     #PR7, @#ERRVEC+2
6921 027744 013737 172516 001202      MOV     @#SR3, $REG10  ;GET SR3
6922 027752 000406      BR      10$
6923 027754 022626      8$: CMP     (SP)+, (SP)+  ;CLEAN UP STACK
6924 027756 112737 000003 064546      MOVB   #3, DF30+18.   ;FIX PRINTOUT FOR NO SR3
6925 027764 105037 063207      CLRB   DH704+11.
6926 027770 104121      10$: ERROR 121 ;KT11 FAILURE
6927 027772 000137 044420      JMP     HLTPRG        ;ABORT !!!

```

6928
6929
6930

```

6931      ;:*****
6932      ;ENBCSR - ENABLE MEMORY CSR'S FOR PARITY ERRORS
6933      ;:*****
6934 027776 010146      ENBCSR: MOV R1, -(SP) ;SAVE R1
6935 030000 013746 000004      MOV @#ERRVEC, -(SP) ;SAVE OLD VECTORS
6936 030004 013746 000006      MOV @#ERRVEC+2, -(SP)
6937 030010 012737 030034 000004      MOV #8$, @#ERRVEC ;SET TIME-OUT VECTOR
6938 030016 012701 172100      MOV #MEMCSR, R1 ;ADRS OF MEMORY CSR'S
6939 030022 005021      4$: CLR (R1)+ ;ZERO THIS CSR
6940 030024 012761 000001 177776      MOV #BIT0, -2(R1) ;SET ENABLE IN THIS CSR
6941 030032 000401      BR 10$
6942 030034 022626      8$: CMP (SP)+, (SP)+ ;CLEAN UP THE STACK
6943 030036 020127 172140      10$: CMP R1, #MEMCSR+40 ;SEE IF DONE CHECKING YET
6944 030042 001367      BNE 4$ ;BR IF NOT
6945 030044 012637 000006      MOV (SP)+, @#ERRVEC+2 ;RESTORE OLD VECTORS
6946 030050 012637 000004      MOV (SP)+, @#ERRVEC
6947 030054 012601      MOV (SP)+, R1 ;RESTORE R1
6948 030056 000207      RTS PC ;RETURN
6949
6950
6951
6952      ;:*****
6953      ;*SERVICE ROUTINE FOR MEM PARITY ERRORS
6954      ;:*****
6955 030060 010146      MPERHD: MOV R1, -(SP) ;SAVE R1
6956 030062 013746 000004      MOV @#ERRVEC, -(SP) ;SAVE OLD VECTORS
6957 030066 013746 000006      MOV @#ERRVEC+2, -(SP)
6958 030072 004737 042130      JSR PC, REPSUP ;GATHER STATUS FOR PRINTOUT
6959 030076 016637 000006 001174      MOV 6(SP), $REG5 ;GET PC OF ERROR
6960 030104 012737 030230 000004      MOV #10$, @#ERRVEC ;SET T.O. VECTOR
6961 030112 012737 000340 000006      MOV #PR7, @#ERRVEC+2
6962      ;HANDLE 11/70 MEMORY PARITY ERROR
6963 030120 013737 177740 001176      MOV @#LOERAD, $REG6 ;LOW ERROR ADRS REG
6964 030126 013737 177742 001200      MOV @#HIERAD, $REG7 ;HI ERROR ADRS REG
6965 030134 013737 177744 001202      MOV @#MEMSYS, $REG10 ;MEMORY SYSTEM REG
6966 030142 012737 063243 064570      MOV #DH707, DF31+16. ;FIX ERROR MSG FOR 11/70
6967 030150 112737 000004 064572      MOVB #4, DF31+18.
6968 030156 104122      ERROR 122 ;11/70 MEM PARITY ERROR
6969 030160 005737 001200      TST $REG7 ;SEE IF BAD MEMORY IS IN PROGRAM AREA
6970 030164 001011      BNE 6$ ;BR IF NOT
6971 030166 023727 001176 072574      CMP $REG6, #RWBUF+6000 ;SEE IF BAD MEM IS IN PROG. AREA
6972 030174 103005      BHS 6$ ;BR IF NOT
6973 030176 105737 003135      4$: TSTB MEMABT ;SEE IF ABORT DESIRED
6974 030202 001002      BNE 6$ ;BR IF NOT
6975 030204 000137 044420      JMP HLTPRG ;ABORT !!!
6976 030210 012637 000006      6$: MOV (SP)+, @#ERRVEC+2 ;RESTORE T.O. VECTOR
6977 030214 012637 000004      MOV (SP)+, @#ERRVEC
6978 030220 012601      MOV (SP)+, R1 ;RESTORE R1
6979 030222 004737 027776      JSR PC, ENBCSR ;GO CLEAR AND ENABLE CSR'S
6980 030226 000002      RTI ;RETURN
6981      ;HANDLE ALL OTHER MEMORY PARITY ERRORS (NON-11/70)
6982 030230 022626      10$: CMP (SP)+, (SP)+ ;CLEAN UP THE STACK
6983 030232 012737 030262 000004      MOV #14$, @#ERRVEC ;SET T.O. VECTOR
6984 030240 012701 172100      MOV #MEMCSR, R1 ;GET FIRST CSR ADDRESS
6985 030244 011137 001200      12$: MOV (R1), $REG7 ;CHECK FOR A MEMORY CSR
6986 030250 100005      BPL 16$ ;BR IF NO ERROR SET HERE

```

6997 030252 010137 001176
6998 030256 104122
6999 030260 000746
6999.5 030262 022626
6999.7 030264 062701 000002
6999.9 030270 020127 172140
6999.95 030274 103753
6999.99 030276 020744

MOV R1,SREG6 ;GET CSR ADDRESS
ERRPOR 122 ;MEMORY PARITY ERRPOR (NON-11/70)
BR 45 ;GO SEE IF SHOULD ABORT
148: CMP (SP)+,(SP)+ ;CLEAN UP STACK
158: ADD #2,R1 ;INCR R1 TO POINT TO NEXT CSR
CMP R1,MEMCSR+40 ;SEE IF DONE CHECKING
BLC 125 ;BR IF NOT
BR 65 ;RETURN

;PREPAR - PREPARE MEM MGT FOR RELOCATION
;THIS SUBROUTINE CALLS INITMM, AND THEN SETS UP CONSTANT
;FOR KIPAR6 IN SAVPAR, AND LOADS THE UNIBUS MAP REGISTERS.
;IF PRESENT.

7004 030300 104406
7005 030302 004737 027614
7006 030306 013700 005604
7007 030312 042700 017777
7008 030316 013701 005606
7009 030322 010003
7010 030324 010104
7011 030326 006100
7012 030330 006101
7013 030332 006100
7014 030334 006101
7015 030336 000301
7016 030340 006100
7017 030342 106001
7018 030344 010137 003200
7019 030350 042701 007777
7020 030354 001420
7022 030356 012700 170200
7023 030362 012701 000037
7024 030366 010320
7025 030370 010420
7026 030372 062703 020000
7027 030376 005504
7028 030400 077106
7029 030402 042765 160000 000010
7030 030410 142765 000003 000007
7031 030416 104407
7032 030420 000207

PREPAR: SAVREG ;SAVE R0-R5
JSR PC,INITMM ;INIT MEM MGT REGISTERS
MOV PMA,R0 ;LO BITS OF MA
BIC #17777,R0 ;MASK FOR BITS 13-15
MOV PMA+2,R1 ;HI BITS OF MA
MOV R0,R3 ;SAVE THESE BITS
MOV R1,R4 ;
ROL R0 ;GET MA BITS 13-21 INTO R1 BITS 7-15
ROL R1
ROL R0
ROL R1
SWAB R1
ROL R0
PORB R1
MOV R1,SAVPAR ;CONSTANT FOR LOADING PAR6 LATER
BIC #7777,R1 ;SEE IF 22-BIT ADDRESSES USED (PDP 11/70)
BEQ 95 ;BR IF NOT (NO UNIBUS MAPPING)
;PREPARE FOR 22-BIT ADDRESSING - LOAD UNIBUS MAP REGISTERS
MOV #MAPLOO,R0 ;STARTING ADDR OF MAP REGISTERS
MOV #31,R1 ;SET REGISTER COUNTER
45: MOV R3,(R0)+ ;LOAD A MAP REGISTER
MOV R4,(R0)+
ADD #20000,R3 ;ADD 4K WORDS
ADC R4
SOB R1,45 ;LOOP UNTIL 31(DEC) ARE LOADED
BIC #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
BICB #3,P.BAHI(R5)
95: RESREG ;RESTORE R0-R5
RTS PC ;RETURN

;FNDXDP - FIND STARTING ADR OF XXDP LOADER, AND STORE IT
;IN XXDPAD.

7040 030422 005737 005576
7041 030426 001404
7042 030430 012737 160000 005540

FNDXDP: TST MAHILM+2 ;TEST HI BITS OF UPPER MEMORY LIMIT
BEQ 65 ;BR IF HI BITS ARE 0
45: MOV #160000,XXDPAD ;START OF 28K IS END OF XXDP


```

7043 030436 000412          BR          B5
7044 030440 023727* 005574 157776 65:  CMP          MAHILM,#157776 ;SEE IF MEM LIMIT > OR = 157776
7045 030446 103370          BHS          45 ;BR IF YES
7046 030450 013737 005574 005540  MOV          MAHILM,XXDPAD ;HI MEM LIMIT IS END OF XXDP
7047 030456 062737 000002 005540  ADD          #2,XXDPAD ;ADD 2 BYTES
7048 030464 162737 006000 005540 85:  SUB          #6000,XXDPAD ;COMPUTE START OF XXDP
7049 030472 000207          RTS          PC ;RETURN

```

```

7050
7051
7052
7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064 030474 104406
7065 030476 005004
7066 030500 000410
7067 030502 104406
7068 030504 105037 003124
7069 030510 105737 003125
7070 030514 001427
7071 030516 012704 000001
7072 030522 105737 000041
7073 030526 001422
7074 030530 012702 003000
7075 030534 013703 005540
7076 030540 013700 005542
7077 030544 013701 005544
7078 030550 005737 056216
7079 030554 100411
7080 030556 005704
7081 030560 001002
7082 030562 012320
7083 030564 000401
7084 030566 012023
7085 030570 005302
7086 030572 001371
7087 030574 104407
7088 030576 000207
7089
7090 030600 004737 027614
7091 030604 006100
7092 030606 006101
7093 030610 006100
7094 030612 006101
7095 030614 000301
7096 030616 006100
7097 030620 106001
7098 030622 010137 172354

```

```

*****
* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
* THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
* PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
* STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
*
* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
* THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
* STORED IN XDPSAV, XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
* ADDRESS).
*****
SAVXDP: SAVREG ;SAVE RO-R5
        CLR          R4 ;SET INDICATOR TO SAVE YXDP
        BR          XDP1
GETXDP: SAVREG ;SAVE RO-R5
        CLR          XOVLD ;CLEAR XXDP OVERLAID INDICATOR
        TST          XDPSVD ;SEE IF XXDP WAS SAVED
        BEQ          XDP2 ;BR IF NOT SAVED
        MOV          #1,R4 ;SET INDICATOR TO GET XXDP
        TST          XDP1 ;SEE IF LOADED BY XXDP
        BEQ          XDP2 ;BR IF NOT
        MOV          #1536,R2 ;GET SET TO MOVE 1536(DEC) WORDS
        MOV          XXDPAD,R3 ;GET ORIG ADR OF START OF XXDP
        MOV          XDPSAV,R0 ;GET SAVE ADR LO BITS
        MOV          XDPSAV+2,R1 ;HI BITS
        TST          $KT11 ;SEE IF MEM MGT PRESENT
        BMI          XDP3 ;BR IF PRESENT
65:     TST          R4 ;SEE IF WANT TO SAVE OR GET XXDP
        BNE          B5 ;BR IF WANT TO GET XXDP
        MOV          (R3)+,(R0)+ ;SAVE A WORD
        BR          10$
85:     MOV          (R0)+,(R3)+ ;GET A WORD
10$:    DEC          R2 ;SEE IF 1536(DEC) WORDS YET
        BNE          65 ;BR IF NOT YET
XDP2:  RESREG ;RESTORE RO-R5
        RTS          PC ;RETURN
;COME HERE IF MEM MGT
XDP3:  JSR          PC,INITMM ;INIT MEM MGT REGISTERS
        ROL          R0 ;GET ADR BITS 13-21 INTO R1 BITS 7-15
        ROL          R1
        ROL          R0
        ROL          R1
        SWAB        R1
        ROL          R0
        RORB        R1
        MOV          R1,#KIPAR6 ;SET UP PAR6

```



```

7:099 030626 013701 005542      MOV      XDPSAV,R1      ;GET LO ADRS BITS
7:100 030632 042701 160000      BIC      #160000,R1    ;FIX UP R1 TO REFERENCE PAR6
7:101 030636 052701 140000      BIS      #140000,R1
7:102 030642 005704      16$: TST      R4          ;SEE IF WANT TO SAVE OR GET XXDP
7:103 030644 001007      BNE      18$          ;BR IF WANT TO GET XXDP
7:104 030646 012305      MOV      (R3)+,R5     ;SAVE A WORD
7:105 030650 005237 177572      INC      @#SRO        ;TURN ON MEMORY MANAGEMENT
7:106 030654 010521      MOV      R5,(R1)+
7:107 030656 005337 177572      DEC      @#SRO        ;TURN OFF MEMORY MANAGEMENT
7:108 030662 000406      BR       20$
7:109 030664 005237 177572      18$: INC      @#SRO        ;TURN ON MEMORY MANAGEMENT
7:110 030670 012105      MOV      (R1)+,R5     ;GET A WORD
7:111 030672 005337 177572      DEC      @#SRO        ;TURN OFF MEMORY MANAGEMENT
7:112 030676 010523      MOV      R5,(R3)+
7:113 030700 032701 020000      20$: BIT      #BIT13,R1 ;SEE IF OVERFLOW TO PAGE 7
7:114 030704 001405      BEQ      22$          ;BR IF NOT
7:115 030706 042701 020000      BIC      #BIT13,R1    ;SET PAGE = 6 AGAIN
7:116 030712 062737 000200 172354      ADD      #200,@#KIPAR6 ;UPDATE PAR BY 4K
7:117 030720 005302      22$: DEC      R2          ;SEE IF 1536 WORDS YET
7:118 030722 001347      BNE      16$          ;BR IF NOT YET
7:119 030724 000723      BR       XDP2        ;BR TO RETURN

```

```

7:120
7:121
7:122
7:123
7:124
7:125
7:126
7:127
7:128
7:129
7:130
7:131
7:132

```

```

;*****
.SBTTL  INITSS - INITIALIZE SUBSYSTEM
;*
;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
;*AND DOES A SUBSYSTEM CLEAR.
;* USES - R2,R5
;* CALL:
;*          JSR    PC,INITSS
;*****

```

```

7:133 030726 012737 042410 003036 INITSS: MOV      #ERRHDL,A.ABNL ;SET UP ABNORMAL ERROR RETURN ADDRESS
7:134 030734 012737 041142 003034      MOV      #ERRFRE,A.NORM ;SET UP NORMAL RETURN ADDRESS
7:135 030742 013702 003026      MOV      RKBAS,R2      ;GET ADDRESS OF RK611 REGISTERS
7:136 030746 012705 002620      MOV      #PARM0,R5     ;GET ADDRESS OF PARAMETER BLOCK
7:137 030752 105037 003133      CLR     NORTRY        ;CLEAR "NO-RETRY" FLAG
7:138 030756 004737 031020      JSR     PC,CLRPRM     ;CLEAR DRIVER INPUT PARAMETERS
7:139 030762 112765 000177 000001      MOV     #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
7:140 030770 004737 040716      JSR     PC,DRVCAL     ;DO SUBSYSTEM CLEAR
7:141 030774 113765 005502 000000      MOV     DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
7:142 031002 113737 005502 002704      MOV     DRIVE,PARM1   ;SET DRIVE NO. IN ALTERNATE P.B.
7:143 031010 113765 003115 000007      MOV     FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
7:144 031016 000207      RTS      PC          ;SUBROUTINE EXIT

```

```

7:145
7:146
7:147
7:148
7:149
7:150
7:151
7:152
7:153
7:154

```

```

;*****
.SBTTL  CLRPRM - CLEAR DRIVER INPUT PARAMETERS
;*THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
;*PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
;* CALL - JSR    PC,CLRPRM
;*****

```

```

7155 031020 010046          CLRPRM: MOV      RD,-(SP)      ;SAVE RD
7156 031022 010546          MOV      RS,-(SP)      ;SAVE RS
7157 031024 010500          MOV      RS,RD        ;GET PARAMETER BLOCK ADDRESS
7158 031026 062705 000016  ADD      #P.CS1,RS     ;COMPUTE LIMIT ADDRESS
7159 031032 005020          IS:   CLR      (RD)+    ;CLEAR A WORD IN PARAMETER BLOCK
7160 031034 020005          CMP      RD,RS        ;SEE IF DONE YET
7161 031036 001375          SNE     IS           ;BR IF NOT DONE YET
7162 031040 012635          MOV      (SP)+,RS     ;RESTORE RS
7163 031042 012600          MOV      (SP)+,RD     ;RESTORE RD
7164 031044 000237          RTS      PC          ;RETURN

```

```

7165
7166
7167
7168 ;*****
7169 ;SBTTL SCNDRV - SCAN DRIVE FOR STATUS
7170 ;*THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
7171 ;*THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
7172 ;*AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE, IF ANY
7173 ;*OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
7174 ;*MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
7175 ;*A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7176 ;*AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
7177 ;*MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
7178 ;*ALSO, IF RUNNING FROM ADRS 200 AND RK06 IS LOAD MEDIUM,
7179 ;*DRIVE 0 WILL BE REJECTED FOR USE.
7180 ;* CALL - JSR PC,SCNDRV
7181 ;*          <ERROR RETURN ADDRESS>
7182 ;*
7183 ;*****
7184

```

```

7185 031046 004737 030726  SCNDRV: JSR      PC,INITSS ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
7186 031052 113765 005502 000000  MOVB     DRIVE,P.DRVN(RS) ;GET DRIVE NUMBER
7187 ;SEE IF DRIVE 0 IS XXDP LOAD MEDIUM
7188 031060 001012          BNE     3$           ;BR IF NOT DRIVE 0
7189 031062 122737 000013 000041  CMPB     #13,2#41     ;SEE IF RK06 IS XXDP MEDIUM
7190 031070 001006          BNE     3$           ;BR IF NOT
7191 031072 105737 003106          TSTB     MDFLAG      ;SEE IF 200 START
7192 031076 001003          BNE     3$           ;BR IF NOT
7193 031100 104400 007667          TYPE     ,DROXDP     ;TYPE "DRIVE 0 IS LOAD MEDIUM"
7194 031104 000434          BR      2$           ;TAKE ERROR EXIT
7195 031106 112765 000141 000001 3$:  MOVB     #RDSTAT,P.CMND(RS) ;SET READ DRIVE STATUS COMMAND
7196 031114 012737 027032 003036  MOV      #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
7197 031122 012737 000377 005532  MOV      #377,NEWON    ;INIT. ON-LINE INDICATOR
7198 031130 004737 040716          JSR      PC,DRVCAL   ;READ ALL DRIVE STATUS
7199 ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
7200 031134 012737 042410 003036  MOV      #ERRHDL,A.ABNL ;RESTORE ABNL RETURN ADDRESS
7201 031142 022737 000377 005532  CMP      #377,NEWON   ;SEE IF NED INDICATION ON THIS DRIVE
7202 031150 001420          BEQ     4$           ;BR IF NED NOT SET
7203 031152 113737 005502 007370  MOVB     DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
7204 031160 152737 000060 007370  BISB     #'0,BADDRV+6 ;CONVERT TO ASCII
7205 031166 104400 007362          TYPE     ,BADDRV    ;TYPE "DRIVE X"
7206 031172 104400 007373          TYPE     ,NXDRIV    ;TYPE "NON-EXISTENT"
7207 ;SERVICE ERRORS HERE
7208 031176 042762 000100 000000 2$:  BIC      #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT
7209 031204 017616 000000          MOV      2(SP),(SP)  ;SET UP ERROR RETURN ADDRESS
7210 031210 000207          RTS      PC          ;ERROR RETURN

```

F13

```

7211          ;SEE IF DRIVE IS READY
7212 031212 032765 000200 000040 4$: BIT #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
7213 031220 001013          BEQ 6$ ;BR IF DRIVE IS READY
7214 031222 113737 005502 007370 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
7215 031230 152737 000060 007370 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7216 031236 104400 007362          TYPE ,BADDRV ;TYPE "DRIVE X"
7217 031242 104400 007412          TYPE ,NTREDY ;TYPE "NOT READY"
7218 031246 000753          BR 2$ ;TAKE ERROR EXIT
7219          ;SEE IF DRIVE IS WRITE ENABLED
7220 031250 032765 004000 000040 6$: BIT #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
7221 031256 001413          BEQ 8$ ;BR IF WRITE LOCK NOT SET
7222 031260 113737 005502 007370 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
7223 031266 152737 000060 007370 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7224 031274 104400 007362          TYPE ,BADDRV ;TYPE "DRIVE X"
7225 031300 104400 007426          TYPE ,WRTLOK ;TYPE "WRITE-LOCKED"
7226 031304 000734          BR 2$ ;TAKE ERROR EXIT
7227          ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
7228 031306 112765 000103 000001 8$: MOVB #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7229 031314 004737 040716          JSR PC,DRVCAL ;SET VOLUME VALID
7230 031320 112765 000121 000001 MOVB #RDATA,P.CMND(R5) ;SET READ COMMAND
7231 031326 012765 000632 000002 MOV #LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)
7232 031334 112765 000002 000005 MOVB #LSTRK,P.TRCK(R5) ;SET TRACK = 2
7233 031342 012765 064574 000010 MOV #RWBUF,P.BALO(R5) ;BUS ADDRESS
7234 031350 012765 177774 000012 MOV #-4,P.WC(R5) ;READ 4 WORDS
7235 031356 105065 000007          CLRB P.CS1H(R5) ;SET 22-SECTOR FORMAT
7236 031362 004737 040716          JSR PC,DRVCAL ;READ 4 WORDS OF BAD SECTOR FILE
7237 031366 032737 100000 005476 BIT #ANYDER,RECODE ;SEE IF DATA ERROR
7238 031374 001402          BEQ 10$ ;BR IF OK
7239 031376 104400 013310          TYPE ,BAD632 ;TYPE READ ERROR MESSAGE
7240 031402 022737 177777 064602 10$: CMP #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
7241 031410 001013          BNE 12$ ;BR IF NOT ALL 1'S
7242 031412 113737 005502 007370 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
7243 031420 152737 000060 007370 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7244 031426 104400 007362          TYPE ,BADDRV ;TYPE "DRIVE X"
7245 031432 104400 007445          TYPE ,ALNPAK ;TYPE "LOADED WITH ALIGN PACK"
7246 031436 000657          BR 2$ ;TAKE ERROR EXIT
7247          ;ERROR FREE RETURN
7248 031440 112765 000113 000001 12$: MOVB #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
7249 031446 004737 040716          JSR PC,DRVCAL ;RECALIBRATE THIS DRIVE
7250 031452 062716 000002          ADD #2,(SP) ;FIX UP RETURN PC
7251 031456 000207          RTS PC ;RETURN
7252
7253
7254
7255 ;*****
7256 ;SBTTL CHKDRV - CHECK STATUS OF DRIVE
7257 ;*THIS SUBROUTINE CHECKS THE DESIRED DRIVE TO DETERMINE
7258 ;*IF THE DRIVE IS ON-LINE,READY,WRITE-PROTECTED. IF NOT, THE
7259 ;*APPROPRIATE ERROR MESSAGE IS TYPED,AND THE CPU HALTS
7260 ;*WHILE MANUAL INTERVENTION TAKES PLACE TO CORRECT THE
7261 ;*PROBLEM. THE OPERATOR THEN DEPRESSES 'CONT' TO PROCEED,
7262 ;*AND A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7263 ;*AFTER THE CALL. IF THERE ARE NO PROBLEMS WITH THIS
7264 ;*DRIVE, A NORMAL RETURN IS MADE.
7265 ;*R5 MUST CONTAIN THE ADDRESS OF THE PARAMETER BLOCK,
7266 ;*AND R2 MUST CONTAIN THE RK06 REGISTER BASE ADDRESS.
  
```

```

7267
7268
7269
7270
7271
7272
7273 031460 113765 005502 000000 CHKDRV: MOVB DRIVE,P.DRVN(R5) ;SET DESIRED DRIVE NUMBER
7274 031466 112765 000141 000001 MOVB #R0STAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
7275 031474 012737 027032 003036 MOV #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
7276 031502 012737 000377 005532 MOV #377,NEWON ;INITIALIZE ON-LINE INDICATOR
7277 031510 004737 040716 JSR PC,DRVCL ;READ ALL DRIVE STATUS
7278 ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
7279 031514 012737 042410 003036 MOV #ERRHDL,A.ABNL ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
7280 031522 022737 000377 005532 CMP #377,NEWON ;SEE IF NED INDICATION ON THIS DRIVE
7281 031530 001411 BEQ 1$ ;BR IF NED NOT SET
7282 031532 113701 005502 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7283 031536 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7284 031542 110137 012037 MOVB R1,DRVNO ;PUT DRIVE NO. INTO MSG BUFFER
7285 031546 104400 012025 TYPE ,NONEXD ;TYPE NON-EXISTENT DRIVE MESSAGE
7286 031552 000414 BR 3$ ;SERVICE THE ERROR
7287 ;SEE IF DESIRED DRIVE IS READY
7288 031554 032765 000200 000040 1$: BIT #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
7289 031562 001024 BNE 4$ ;BR IF DRIVE READY
7290 031564 113701 005502 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7291 031570 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7292 031574 110137 011673 MOVB R1,DRNRDY ;PUT DRIVE NUMBER IN MESSAGE BUFFER
7293 031600 104400 011661 TYPE ,NOTRDY ;TYPE DRIVE NOT READY MESSAGE
7294 ;SERVICE ALL ERRORS HERE
7295 031604 042762 000100 000000 3$: BIC #IE,RKCS1(R2) ;CLEAR RK06 INTERRUPT ENABLE BIT
7296 031612 000000 HALT ;HALT IS NECESSARY DURING MANUAL
7297 ; INTERVENTIONS, TO CHANGE PACK,
7298 ; WRITE LOCK, START DRIVE, ETC.
7299 ; PRESS 'CONT' TO PROCEED !
7300 031614 112765 000177 000001 MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
7301 031622 004737 040716 JSR PC,DRVCL ;CLEAR THE SUBSYSTEM
7302 031626 017616 000000 MOV 2(SP),(SP) ;SET UP ERROR RETURN ADDRESS
7303 031632 000207 RTS PC ;ERROR RETURN
7304 ;SEE IF DRIVE IS WRITE-LOCKED
7305 031634 032765 004000 000040 4$: BIT #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
7306 031642 001011 BNE 5$ ;BR IF WRITE-PROTECTED
7307 031644 113701 005502 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7308 031650 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7309 031654 110137 012166 MOVB R1,NOORLK ;PUT DRIVE NUMBER IN MSG BUFFER
7310 031660 104400 012154 TYPE ,NOTLOK ;TYPE "DRIVE NOT WRITE-LOCKED"
7311 031664 000747 BR 3$ ;SERVICE THE ERROR
7312 ;RECALIBRATE DESIRED DRIVE, SET VOLUME VALID
7313 031666 112765 000113 000001 5$: MOVB #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
7314 031674 004737 040716 JSR PC,DRVCL ;RECALIBRATE DRIVE
7315 031700 112765 000103 000001 MOVB #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7316 031706 004737 040716 JSR PC,DRVCL ;DO PACK ACK. (SETS VOLUME VALID)
7317 ;RETURN HERE
7318 031712 062716 000002 7$: ADD #2,(SP) ;FIX UP RETURN ADDRESS ON STACK
7319 031716 000207 RTS PC ;ERROR-FREE RETURN
7320
7321
7322

```

H13

```
7323 ;*****
7324 ;* ALNSEK - SEEK IN SINGLE INCREMENTS FROM CYL 0 TO ALIGNMENT CYL (365 OCT).
7325 ;*****
7326 031720 112765 000117 000001 ALNSEK: MOVB #SEEK.P.CMND(R5) ;SET SEEK COMMAND
7327 031726 005065 000002 CLR P.CYLN(R5) ;INIT CYL TO 0
7328 031732 004737 040716 2$: JSR PC,DRVCAL ;SEEK TO THIS CYL
7329 031736 005265 000002 INC P.CYLN(R5) ;INCR CYL NO.
7330 031742 022765 000366 000002 CMP #ALNCYL+1,P.CYLN(R5) ;SEE IF 365 REACHED YET
7331 031750 001370 BNE 2$ ;BR IF NOT YET
7332 031752 000207 RTS PC ;RETURN
7333
7334
7335
7336 ;*****
7337 ;* WAIT4R - THIS SUBROUTINE UNLOADS HEADS ON THE CURRENT DRIVE, TYPES
7338 ;* "TYPE <R> WHEN READY", AND THEN LOADS THE HEADS WHEN <R> IS TYPED.
7339 ;*****
7340 031754 112765 000107 000001 WAIT4R: MOVB #UNLOAD.P.CMND(R5) ;SET UNLOAD COMMAND
7341 031762 004737 040716 JSR PC,DRVCAL ;UNLOAD HEADS ON THIS DRIVE
7342 031766 112765 000141 000001 MOVB #RDSTAT.P.CMND(R5) ;SET READ STATUS COMMAND
7343 031774 004737 040716 1$: JSR PC,DRVCAL ;READ STATUS OF DRIVE
7344 032000 032765 000040 000044 BIT #S.HOVM,P.A01(R5) ;SEE IF HEADS UNLOADED YET
7345 032006 001772 BEQ 1$ ;BR IF NOT YET
7346 032010 104400 012751 2$: TYPE ,RWNRDY ;TYPE "TYPE <R> WHEN READY"
7347 032014 004737 027250 JSR PC,PREPKB ;PREPARE FOR KBD INPUT
7348 032020 005737 005524 4$: TST INTCHR ;SEE IF ANY INPUT YET
7349 032024 001775 BEQ 4$ ;BR IF NOT YET
7350 032026 022737 000122 005524 CMP #R,INTCHR ;SEE IF <R> TYPED
7351 032034 001403 BEQ 6$ ;BR IF <R> TYPED
7352 032036 004737 027270 JSR PC,ECOBAD ;ECHO BAD INPUT
7353 032042 000762 BR 2$ ;GO ASK AGAIN
7354 032044 112765 000111 000001 6$: MOVB #SRTSPL.P.CMND(R5) ;SET START SPINDLE COMMAND
7355 032052 004737 040716 JSR PC,DRVCAL ;START SPINDLE AND LOAD HEADS
7356 032056 000207 RTS PC ;RETURN
7357
7358
7359
7360 ;*****
7361 ;*SETUP - SET UP FOR LOOP ON ERROR
7362 ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
7363 ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
7364 ;*****
7365 032060 011637 001076 SETUP: MOV (SP),2#STACK-2 ;MOVE RETURN PC ON STACK
7366 032064 012706 001076 MOV #STACK-2,SP ;RE-INIT THE STACK POINTER
7367 032070 005037 005476 CLR RECODE ;CLEAR ERROR RECOVERY FLAGS
7368 032074 105037 003116 CLR ERRCNT ;CLEAR RETRY ERROR COUNT
7369 032100 012705 002620 MOV #PARM,R5 ;SET PARAM BLK ADRS
7370 032104 112765 000177 000001 MOVB #SUBCLR.P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
7371 032112 004737 040716 JSR PC,DRVCAL ;CLEAR THE SUBSYSTEM
7372 032116 012737 000000 177776 MOV #PRO,2#PS ;RE-ESTABLISH PRIORITY 0
7373 032124 000207 RTS PC ;RETURN
7374
7375
7376
7377 ;*****
7378 ;SBTTL CHKITR - CHECK CURRENT TEST ITERATION NUMBER
```

```

7379 ;*THIS SUBROUTINE LOADS THE ITERATION NUMBER FOR THE CURRENT
7380 ;*TEST INTO $TIMES, AND CHECKS IT TO DETERMINE IF THIS TEST
7381 ;*SHOULD BE RUN. IF ITERATION NO. = 0, THE SUBROUTINE RETURNS
7382 ;*TO THE STORED PC (AND THE TEST WILL BE SKIPPED). IF IT IS
7383 ;*NOT 0, THE RETURN IS MADE TO PC+2, AND THE TEST WILL
7384 ;*BE RUN. HOWEVER, ON THE FIRST PASS (QUICK VERIFY) OF A RUN
7385 ;*WHICH STARTED AT ADRS 200, ALL TESTS WILL BE RUN ONCE.
7386
7387 032126 010146          CHKITR: MOV     R1,-(SP)      ;SAVE R1
7388 032130 105737 003126  TSTB   DULACS      ;SEE IF DUAL-ACCESS FLAG SET
7389 032134 001006          BNE     3$           ;BR IF YES
7390 032136 105737 003106  TSTB   MDFLAG      ;SEE IF 200 START
7391 032142 001007          BNE     4$           ;BR IF NOT
7392 032144 005737 001326  TST    $PASS       ;SEE IF FIRST PASS
7393 032150 001004          BNE     4$           ;BR IF NOT
7394 032152 012737 000001 001304 3$:  MOV    #1,$TIMES    ;SET UP FOR 1 ITERATION
7395 032160 000410          BR     1$           ;GO RUN DUAL-ACCESS TEST
7396 032162 013701 001102 4$:  MOV    $TSTN,R1     ;GET CURRENT TEST NUMBER
7397 032166 005301          DEC    R1           ;DECREMENT BY 1
7398 032170 006301          ASL    R1           ;DOUBLE IT, TO GET TEST LIST INDEX
7399 032172 016137 005620 001304  MOV    TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
7400 032200 001403          BEQ    2$           ;BR IF TEST SHOULD BE SKIPPED
7401 032202 062766 000004 000002 1$:  ADD    #4,2(SP)     ;ADJUST RETURN PC TO RUN THIS TEST
7402 032210 012601          2$:  MOV    (SP)+,R1     ;RESTORE R1
7403 032212 000207          RTS     PC         ;RETURN
7404
7405
7406
7407
7408 ;*****
7409 ;SBTTL RDCHRS - READ A STRING OF KBD INPUT CHARS
7410 ;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
7411 ;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
7412 ;*THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
7413 ;*RUB-OUT AND (↑) FEATURES ARE PROVIDED.
7414 ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
7415 ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
7416 ;*TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
7417 ;*TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
7418 ;*IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
7419 ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
7420 ;*IS TAKEN.
7421 ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
7422 ;*IN RD.
7423 ;*
7424 ;* CALL - JSR PC,RDCHRS
7425 ;*          <CONTROL-C RETURN ADDRESS>
7426 ;*          <CONTROL-Z RETURN ADDRESS>
7427 ;*          <CONTROL-U OR ERROR RETURN ADDRESS>
7428 ;*          RETURN
7429 ;*****
7430
7431 032214          RDCHRS: MOV    R1,-(SP)      ;SAVE R1
7432 032216 010146          MOV    R2,-(SP)     ;SAVE R2
7433 032220 005000          CLR    R0           ;INITIALIZE CHARACTER COUNT
7434 032222 005001          CLR    R1           ;INITIALIZE RUB-OUT INDICATOR
    
```

```

7435          ;READ A CHARACTER
7436 032224 104405 2$: RDCHR          ;READ A CHARACTER
7437 032226 112602          ;GET CHARACTER INTO R2
7438          ;CHECK FOR (↑C)
7439 032230 122702 0C0003  CMPB #003,R2      ;SEE IF (↑C) TYPED
7440 032234 001006          BNE 4$          ;BR IF NOT (↑C)
7441 032236 104400 013354  TYPE ,CNTRLC      ;ECHO (↑C)
7442 032242 017666 000004 000004 3$: MOV 24(SP),4(SP) ;PUT RETURN ADDRESS ON STACK
7443 032250 000523          BR 24$         ;BR TO TAKE EXIT
7444          ;CHECK FOR (↑Z)
7445 032252 122702 000032 4$:  CMPB #032,R2      ;SEE IF (↑Z) TYPED
7446 032256 001006          BNE 6$          ;BR IF NOT (↑Z)
7447 032260 104400 013361  TYPE ,CNTRLZ      ;ECHO (↑Z)
7448 032264 062766 000002 000004  ADD #2,4(SP)    ;MAKE OLD PC POINT TO NEXT RETURN ADR.
7449 032272 000763          BR 3$          ;BR TO TAKE (↑Z) EXIT
7450          ;CHECK FOR (↑U)
7451 032274 122702 000025 6$:  CMPB #025,R2      ;SEE IF (↑U) TYPED
7452 032300 001006          BNE 8$          ;BR IF NOT (↑U)
7453 032302 104400 013373  TYPE ,CNTRLU      ;ECHO (↑U)
7454 032306 062766 000004 000004 7$:  ADD #4,4(SP)    ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
7455 032314 000752          BR 3$          ;BR TO TAKE (↑U) EXIT
7456          ;CHECK FOR (↑G)
7457 032316 122702 000007 8$:  CMPB #007,R2      ;SEE IF (↑G) TYPED
7458 032322 001005          BNE 9$          ;BR IF NOT (↑G)
7459 032324 104400 013400  TYPE ,CNTRLG      ;ECHO (↑G)
7460 032330 004737 027562  JSR PC,GTSWRG  ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
7461 032334 000764          BR 7$          ;TAKE (↑U) RETURN
7462          ;CHECK FOR RUB-OUT (DELETE)
7463 032336 122702 000177 9$:  CMPB #177,R2      ;SEE IF RUB-OUT (DEL) TYPED
7464 032342 001020          BNE 14$        ;BR IF NOT RUB-OUT
7465 032344 005700          TST R0         ;CHECK THE CHARACTER COUNT
7466 032346 001726          BEQ 2$          ;BR IF COUNT = 0
7467 032350 005701          TST R1         ;CHECK THE RUB-OUT INDICATOR
7468 032352 001003          BNE 11$        ;BR IF WE HAD A PREVIOUS RUB-OUT
7469 032354 005201          INC R1         ;SET RUB-OUT INDICATOR
7470 032356 104400 013411  TYPE ,BKSLSH    ;TYPE A BACK-SLASH (\)
7471 032362 005037 005534 11$: CLR SCRACH     ;USE SCRATCH WORD FOR TEMP. BUFFER
7472 032366 005300          DEC R0         ;DECREMENT COUNT
7473 032370 116037 005264 005534  MOVB BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
7474 032376 104400 005534  TYPE ,SCRACH     ;ECHO CHARACTER TO BE DELETED
7475 032402 000710          BR 2$          ;GO READ ANOTHER CHARACTER
7476 032404 005701          TST R1         ;CHECK THE RUB-OUT INDICATOR
7477 032406 001403          BEQ 16$        ;BR IF INDICATOR IS NOT SET
7478 032410 104400 013411  TYPE ,BKSLSH    ;TYPE A BACK-SLASH
7479 032414 005001          CLR R1         ;CLEAR THE RUB-OUT INDICATOR
7480          ;CHECK FOR CARRIAGE RETURN
7481 032416 122702 000015 16$: CMPB #015,R2      ;SEE IF <CR> TYPED
7482 032422 001426          BEQ 19$        ;BR IF <CR>
7483          ;HANDLE POSSIBLE DIGIT
7484 032424 005037 005534  CLR SCRACH     ;USE SCRATCH WORD FOR TEMP. BUFFER
7485 032430 110237 005534  MOVB R2,SCRACH ;GET THIS CHARACTER INTO BUFFER
7486 032434 104400 005534  TYPE ,SCRACH     ;ECHO THE CHARACTER TYPED
7487 032440 110260 005264  MOVB #2,BUFFO(R0) ;PUT CHARACTER INTO BUFFER
7488 032444 005200          INC R0         ;INCREMENT CHARACTER COUNTER
7489 032446 022700 000120  CMP #80.,R0    ;SEE IF TOO MANY CHARACTERS TYPED
7490 032452 001264          BNE 2$          ;BR IF NOT TOO MANY
    
```



```

7491 032454 104400 001315          TYPE      $CRLF          ;TYPE <CR> AND <LF>
7492 032460 112760 000000 005264  MOVB     #0,BUFF0(R0) ;PUT TERMINATING NULL INTO BUFFER
7493 032466 104400 005264          TYPE      ,BUFF0        ;ECHO INPUT STRING
7494 032472 104400 001314          TYPE      $QUES        ;TYPE <?>, <CR>, <LF>
7495 032476 000703          BR       ?$            ;TAKE ERROR EXIT FROM RDCHRS
7496 032500 104400 001315          TYPE      $CRLF          ;TYPE <CR>, <LF>
7497 032504 112760 000000 005264  MOVB     #0,BUFF0(R0) ;PUT TERMINATING NULL INTO BUFFER
7498 032512 062766 000006 000004  ADD      #6,4(SP)     ;FIX UP RETURN ADDRESS
7499 032520 012602          MOVB     (SP)+,R2     ;RESTORE R2
7500 032522 012601          MOV      (SP)+,R1     ;RESTORE R1
7501 032524 000207          RTS       PC          ;SUBROUTINE EXIT

```

7502
7503
7504
7505

```

;*****
;SBTTL TYPTST - TYPE CURRENT TEST AND ITERATION NUMBER
;*THIS SUBROUTINE TYPES : "XX YYYYYY" WITH NO <CR>
;*OR <LF>. XX IS THE NUMBER OF THE CURRENT TEST AND
;*YYYYYY IS THE NUMBER OF ITERATIONS FOR THAT TEST.
;*R1 MUST CONTAIN THE INDEX INTO THE TEST LIST FOR
;*THE CURRENT TEST.
;* CALL - JSR PC, TYPTST
;*****

```

```

7515 032526 104400 013422  TYPTST: TYPE      ,SPACE1      ;TYPE A SPACE
7516 032532 010146          MOV      R1,-(SP)     ;PUT INDEX ONTO STACK
7517 032534 006216          ASR      (SP)        ;DIVIDE BY 2
7518 032536 005216          INC      (SP)        ;INCREMENT TO GET TEST NO.
7519 032540 104402          TYPOS     ;TYPE TEST NO. IN OCTAL
7520 032542 002          .BYTE    2          ;TYPE 2 DIGITS
7521 032543 000          .BYTE    0          ;SUPPRESS LEADING ZEROS
7522 032544 104400 013415          TYPE      ,SPACE6      ;TYPE 6 SPACES
7523 032550 016146 005620          MOV      TSTLST(R1),-(SP) ;PUT CURRENT ITERATION NO. ONTO STACK
7524 032554 104402          TYPOS     ;TYPE ITERATION NO. IN OCTAL
7525 032556 006          .BYTE    6          ;TYPE 6 DIGITS
7526 032557 000          .BYTE    0          ;SUPPRESS LEADING ZEROS
7527 032560 000207          RTS       PC          ;RETURN

```

7528
7529
7530

```

;*****
;SBTTL TYPPRM - TYPE CURRENT PARAMETER
;*THIS SUBROUTINE TYPES THE LINE : XX=YYYYYYYY
;*WHERE XX IS THE CURRENT PARAMETER MNEMONIC, AND
;*YYYYYYYY IS ITS VALUE IN OCTAL DIGITS, WITH LEADING
;*ZEROS SUPPRESSED. NOTE: <CR> AND <LF> ARE NOT DONE.
;*ON ENTRY, R1 MUST CONTAIN THE INDEX INTO THE
;*PARAMETER TABLES, FOR THE CURRENT PARAMETER.
;* CALL - JSR PC, TYPPRM
;*****

```

```

7541 032562 016137 006000 013424  TYPPRM: MOV      PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
7542 032570 104400 013424          TYPE      ,PRMBUF     ;TYPE "XX="
7543 032574 016137 005650 005470  MOV      PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
7544 032602 005037 005472          CLR      HIGOCT      ;CLEAR HIGH BINARY BITS
7545 032606 022761 040515 006000  CMP      #*MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
7546 032614 001003          BNE     IS          ;BR IF NOT (MA)

```



```

7547 032616 016137 005652 005472      MOV      PRMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
7548 032624 012746 005470      15:     MOV      #LOWOCT,-(SP) ;PUT ADDRESS OF BINARY VALUE ON STACK
7549 032630 004737 053674      JSR      PC,@#SD820 ;CONVERT BINARY TO OCTAL ASCII
7550 032634 004737 054210      JSR      PC,@#SSUPRS ;TYPE YYYYYYYY, SUPPRESSING LEADING 0'S
7551 032640 000207      RTS      PC ;RETURN

```

```

7552
7553
7554
7555

```

```

;*****
;SBTTL TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15
;THIS SUBROUTINE TYPES THE LINE : WORD XX = YYYYYY,
;WHERE XX IS THE NUMBER OF THE CURRENT WORD OF
;USER-DEFINED PATTERN 15, AND YYYYYY IS ITS 16-BIT
;VALUE, IN OCTAL DIGITS. <CR> AND <LF> ARE NOT TYPED.
;ON ENTRY, R1 MUST CONTAIN THE INDEX OF THE CURRENT
;WORD IN THE PATTERN 15 TABLE.
;* CALL - JSR PC,TYPPAT
;*****

```

```

7565
7566 032642
7567 032642 104400 013430
7568 032646 010146
7569 032650 006216
7570 032652 104402
7571 032654 002
7572 032655 001
7573 032656 104400 013436
7574 032662 016146 006766
7575 032666 104402
7576 032670 006
7577 032671 001
7578 032672 000207
7579
7580
7581

```

```

TYPPAT:
TYPE      WORDSP      ;TYPE "WORD "
MOV      R1,-(SP)      ;PUT INDEX ONTO STACK
ASR      (SP)          ;DIVIDE BY TWO FOR WORD NO.
TYPOS     ;GO TYPE WORD NUMBER
.BYTE    2             ;DIGIT COUNT = 2 FOR TYPOC
.BYTE    1             ;TELL TYPOS TO TYPE LEADING ZEROS
TYPE     EQUALS        ;TYPE " = "
MOV      PAT15(R1),-(SP) ;GET BINARY VALUE OF THIS WORD
TYPOS     ;TYPE VALUE IN OCTAL
.BYTE    6             ;TYPE 6 DIGITS
.BYTE    1             ;TYPE LEADING ZEROS
RTS      PC            ;RETURN

```

```

7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592
7593
7594
7595

```

```

;*****
;SBTTL MODP15 - MODIFY USER-DEFINED PATTERN 15
;THIS SUBROUTINE DETERMINES WHETHER OR NOT USER-DEFINED
;DATA PATTERN 15 SHOULD BE OPENED FOR MODIFICATION, AND
;IF SO, IT REQUESTS THE NEW VALUES AND CONTROLS THEIR
;INPUT ON THE TTY. IT CHECKS VALUES FOR VALIDITY, AND
;LOADS THE 16 WORDS INTO THE PATTERN 15 TABLE (PAT15).
;IF THE OPERATOR ANSWERS THE REQUEST FOR A PATTERN WORD
;WITH EXCLAMATION POINT (!) (CR), THE SUBROUTINE
;PROPAGATES THE LAST WORD TYPED INTO ALL REMAINING
;WORDS OF THE PATTERN 15 TABLE.
;* CALL - JSR PC,MODP15
;*****

```

```

7596 032674
7597 032674 010046
7598 032676 010146
7599 032700 010246
7600
7601 032702 022761 052120 006000
7602 032710 001127

```

```

MODP15:
MOV      R0,-(SP)      ;SAVE R0
MOV      R1,-(SP)      ;SAVE R1
MOV      R2,-(SP)      ;SAVE R2
;SEE IF PARAMETER IS PT
CMP      #PT,PRMNEM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
BNE     22$           ;BR IF NOT (PT)

```

M13

```

7603          :SEE IF PATTERN 15 IS SPECIFIED
7604 032712 032737 100000 005670 BIT #BIT15,PT ;SEE IF PATTERN 15 SPECIFIED
7605 032720 001523 BEQ 22$ ;BR IF NOT SPECIFIED
7606          :SEE IF PATTERN 15 SHOULD BE MODIFIED
7607 032722 104400 010750 4$: TYPE MDY15 ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
7608 032726 004737 032214 JSR PC, RDCHRS ;READ RESPONSE
7609 032732 033200 24$ ;(+C) RETURN ADDRESS
7610 032734 033210 26$ ;(+Z) RETURN ADDRESS
7611 032736 032774 8$ ;(+U) OR ERROR RETURN ADDRESS
7612 032740 005700 TST R0 ;SEE IF NULL INPUT
7613 032742 001512 BEQ 22$ ;BR IF MODIFICATION NOT REQUESTED
7614 032744 022737 000115 005264 CMP #'M, BUFF0 ;SEE IF (M) TYPED
7615 032752 001405 BEQ 6$ ;BR IF MODIFICATION REQUESTED
7616 032754 104400 005264 TYPE ,BUFF0 ;ECHO BAD INPUT
7617 032760 104400 001314 TYPE $QUES
7618 032764 000756 BR 4$ ;GO ASK AGAIN
7619          :MODIFY PATTERN 15
7620 032766 104400 010720 6$: TYPE SELP15 ;TYPE "MODIFY PATTERN 15"
7621 032772 005001 CLR R1 ;INITIALIZE WORD INDEX
7622 032774 004737 032642 8$: JSR PC, TYPPAT ;TYPE CURRENT WORD AND VALUE
7623 033000 104400 013422 TYPE ,SPACE1 ;TYPE A SPACE
7624 033004 104400 013442 TYPE ,PROMPT ;TYPE ASTERISK AND SPACE
7625          :READ AND CHECK INPUT, IF ANY
7626 033010 004737 032214 JSR PC, RDCHRS ;READ NEW DATA PATTERN WORD
7627 033014 033200 24$ ;(+C) RETURN ADDRESS FOR RDCHRS
7628 033016 033210 26$ ;(+Z) RETURN ADDRESS FOR RDCHRS
7629 033020 032774 8$ ;(+U) OR ERROR RETURN ADDR. FOR RDCHRS
7630 033022 005700 TST R0 ;SEE IF ANY INPUT
7631 033024 001006 BNE 12$ ;BR IF ANY INPUT
7632 033026 062701 000002 10$: ADD #2, R1 ;INCREMENT WORD INDEX
7633 033032 022701 000040 CMP #32., R1 ;SEE IF ALL DONE
7634 033036 001454 BEQ 22$ ;BR IF DONE, TO RETURN
7635 033040 000755 BR 8$ ;CONTINUE WITH NEXT WORD
7636 033042 022737 000041 005264 12$: CMP #'!, BUFF0 ;SEE IF (!) TYPED
7637 033050 001431 BEQ 16$ ;BR TO PROPAGATE CURRENT VALUE
7638 033052 005002 CLR R2 ;INIT. (!) INDICATOR
7639 033054 122760 000041 005263 CMPB #'!, BUFF0-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
7640 033062 001004 BNE 14$ ;BR IF NOT (!)
7641 033064 105060 005263 CLRB BUFF0-1(R0) ;INSERT TERMINATOR BYTE
7642 033070 005300 DEC R0 ;DECREMENT CHAR COUNT
7643 033072 005202 INC R2 ;SET (!) INDICATOR
7644 033074 022700 000006 14$: CMP #6, R0 ;SEE HOW MANY CHARS NOW
7645 033100 002426 BLT 20$ ;BR IF TOO MANY
7646 033102 012746 005264 MOV #BUFF0, -(SP) ;GET BUF. ADDR. ON STACK FOR OCTBIN
7647 033106 004737 052736 JSR PC, OCTBIN ;CHECK DIGITS AND CONVERT TO BINARY
7648 033112 033156 20$ ;ERROR RETURN ADDRESS FOR OCTBIN
7649 033114 012600 MOV (SP)+, R0 ;GET BINARY VALUE
7650 033116 005737 053070 TST $HI OCT ;SEE IF VALUE EXCEEDS 16 BITS
7651 033122 001015 BNE 20$ ;BR TO ECHO BAD INPUT
7652 033124 010061 006766 MOV R0, PAT15(R1) ;PUT NEW WORD VALUE INTO TABLE
7653 033130 005702 TST R2 ;SEE IF (!) WAS TYPED
7654 033132 001735 BEQ 10$ ;BR IF (!) WAS NOT TYPED
7655          :PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE
7656 033134 022701 000036 16$: CMP #30., R1 ;SEE IF ALL DONE YET
7657 033140 001413 BEQ 22$ ;BR IF DONE
7658 033142 016161 006766 006770 MOV PAT15(R1), PAT15+2(R1) ;PROPAGATE TO NEXT WORD
  
```

```

7659 033150 062701 000002          ADD      #2,R1          ;INCREMENT WORD INDEX
7660 033154 000767          BR       16$          ;LOOP UNTIL DONE
7661          ;ECHO BAD INPUT
7662 033156 104400 005264          20$:    TYPE      ,BUFFO          ;ECHO BAD INPUT
7663 033162 104400 001314          TYPE      $QUES          ;TYPE (?) AND <CR>, <LF>
7664 033166 000702          BR       8$          ;BR TO ASK AGAIN
7665          ;NORMAL RETURN
7666 033170 012602          22$:    MOV      (SP)+,R2          ;RESTORE R2
7667 033172 012601          MOV      (SP)+,R1          ;RESTORE R1
7668 033174 012600          MOV      (SP)+,R0          ;RESTORE R0
7669 033176 000207          RTS      PC          ;RETURN
7670          ;((1C) RETURN
7671 033200 012766 014520 000006          24$:    MOV      #DRVTST,6(SP)      ;PC FOR (1C) RETURN
7672 033206 000770          BR       22$          ;BR TO RETURN
7673          ;((1Z) RETURN
7674 033210 012766 016124 000006          26$:    MOV      #ASKMODE,6(SP)      ;PC FOR (1Z) RETURN
7675 033216 000764          BR       22$          ;BR TO RETURN
7676
7677
7678
7679          ;*****
7680          ;SBTTL  CHKPRM - CHECK VALUE OF INPUT PARAMETER
7681          ;*THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
7682          ;*HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
7683          ;*LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
7684          ;*INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
7685          ;*TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
7686          ;*IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
7687          ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
7688          ;*THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
7689          ;*VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
7690          ;* CALL -      JSR      PC,CHKPRM
7691          ;*              <ERROR RETURN ADDRESS>
7692          ;*              RETURN
7693          ;*****
7694
7695 033220 104406          CHKPRM: SAVREG          ;SAVE R0-R5
7696 033222 010102          MOV      R1,R2          ;GET COPY OF INDEX
7697 033224 006302          ASL     R2              ;DOUBLE IT
7698 033226 022761 040515 006000          CMP     #MA,PRMNEM(R1)  ;SEE IF PARAMETER IS (MA)
7699 033234 001422          BEQ     20$            ;BR IF (MA)
7700 033236 005737 005472          TST     HIGOCT          ;SEE IF HIGH BITS = 0
7701 033242 001403          BEQ     18$            ;BR IF ZERO
7702 033244 017616 000000          16$:    MOV      2(SP),(SP)      ;GET ERROR RETURN PC
7703 033250 000475          BR       30$            ;GO TO RETURN
7704          ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
7705 033252 023762 005470 005724          18$:    CMP     LOWOCT,PRMLIM(R2)  ;SEE IF INPUT VALUE IS TOO SMALL
7706 033260 103771          BLO     16$            ;BR IF INPUT VALUE TOO SMALL
7707 033262 023762 005470 005726          CMP     LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
7708 033270 101365          BHI     16$            ;BR IF INPUT VALUE IS TOO LARGE
7709          ;UPDATE 16-BIT PARAMETER VALUE IN LIST
7710 033272 013761 005470 005650          MOV     LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
7711 033300 000457          BR       28$            ;BR TO RETURN
7712          ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
7713 033302 032737 000001 005470          20$:    BIT     #BIT0,LOWOCT      ;SEE IF MA IS ODD
7714 033310 001355          BNE     16$            ;BR IF MA IS ODD
    
```

```

7715 033312 023762 005472 005726      CMP      HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
7716 033320 103751      BLO      16$ ;BR IF HIGH WORD IS TOO SMALL
7717 033322 001004      BNE      24$ ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
7718 033324 023762 005470 005724      CMP      LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
7719 033330 103744      BLO      16$ ;BR IF LOW WORD IS TOO SMALL
7720 033334 023762 005472 005732 24$:  CMP      HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
7721 033342 101340      BHI      16$ ;BR IF HIGH WORD TOO BIG
7722 033344 001004      BNE      26$ ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
7723 033346 023762 005470 005730      CMP      LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
7724 033354 101333      BHI      16$ ;BR IF LOW WORD IS TOO LARGE
7725      ;UPDATE 32-BIT PARAMETER VALUE IN LIST
7726 033356 013761 005470 005650 26$:  MOV      LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
7727 033364 013761 005472 005652      MOV      HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
7728      ;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
7729 033372 004737 033450      JSR      PC,MXWRDC ;GET WORD COUNT IN R1-RO
7730 033376 104400 010451      TYPE     MAWRDC ;TYPE "MAX WORD COUNT = "
7731 033402 010037 003170      MOV      RO,SUMLO1
7732 033406 010137 003172      MOV      R1,SUMHI1
7733 033412 012746 003170      MOV      #SUMLO1,-(SP) ;PUT POINTER ON STACK
7734 033416 004737 053E74      JSR      PC,$DB20 ;CONVERT TO OCTAL
7735 033422 004737 054210      JSR      PC,$SUPRS ;TYPE IT
7736 033426 104400 001315      TYPE     $CRLF ;TYPE (CR),(LF)
7737 033432 062766 000002 000014      ADD      #2,14(SP) ;INCREMENT R1 INDEX
7738 033440 062716 000002 28$:  ADD      #2,(SP) ;GET NORMAL RETURN PC
7739 033444 104407 30$:  RESREG ;RESTORE RO-R5
7740 033446 000207      RTS      PC ;RETURN

```

```

*****
*MXWRDC - GET MAX WORD COUNT FOR CURRENT MEM LIMITS AND CURRENT MA.
*AND LEAVE THE WORD COUNT IN R1-RO. NO REGISTERS ARE SAVED.
*****

```

```

7748 033450 013700 005574      MXWRDC: MOV      MAHILM,RO ;GET LO BITS OF MEM LIM
7749 033454 013701 005576      MOV      MAHILM+2,R1 ;HI BITS OF MEM LIM
7750 033460 163700 005600      SUB      MA,RO ;SUBTRACT MA FROM MAHILM
7751 033464 005601      SBC      R1
7752 033466 163701 005602      SJB      MA+2,R1
7753 033472 100413      BMI      27$ ;IF NEGATIVE, RETURN
7754 033474 000241      CLC ;DIVIDE BY 2 TO GET WORDS
7755 033476 006001      ROR      R1
7756 033500 006000      ROR      RO
7757 033502 062700 000001      ADD      #1,RO ;INCREMENT BY 1 WORD
7758 033506 005501      ADC      R1
7759 033510 005701      TST      R1
7760 033512 001403      BEQ      27$ ;BR IF WORD COUNT < 65,536 DEC
7761 033514 005000      CLR      RO ;MAKE WORD COUNT = 65,536
7762 033516 012701 000001      MOV      #1,R1
7763 033522 000207 27$:  RTS      PC ;RETURN

```

```

*****
$BTTL DRVSR - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
*THIS SUBROUTINE TYPES "DRIVE SER. NO. XXX" (IN DECIMAL), WITH LEADING
*ZEROS SUPPRESSED.

```

```

*****
DRYSER: SAVREG          ;SAVE RD-R5
JSR PC,INITSS          ;CLEAR S.S. AND PARAMETERS
MOVB #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
JSR PC,DRVCAL          ;READ STATUS OF THIS DRIVE
TYPE ,DRIV            ;TYPE "DRIVE"
TYPE ,SERNM           ;TYPE "SER. NO."
MOV P,A11(R5),R1      ;GET "A" STATUS BYTE 11
MOV #SOCTVL,R4        ;GET ADDR OF CHAR BUFFER
MOV R4,-(SP)          ;STORE IT ON STACK FOR $SUPRS
MOV #3,R3              ;INIT CHAR COUNT
RCL R1                 ;INITIALIZE BIT POSITIONS
4S: ROL R1              ;GET NEXT 4 BITS
    ROL R1
    ROL R1
    ROL R1
    MOV R1,R0          ;GET A WORKING COPY
    BIC #177760,R0     ;CLEAR ALL BUT LOW 4 BITS
    BIS #0,R0          ;CONVERT A DIGIT TO ASCII
    MOVB R0,(R4)+      ;PUT ASCII DIGIT INTO CHAR BUFFER
    DEC R3              ;DECREMENT CHAR COUNT
    BNE 4S             ;BR IF NOT 3 CHARS YET
    CLRB (R4)          ;INSERT NULL TERMINATOR
    JSR PC,2*$SUPRS    ;TYPE DRIVE SER NUMBER
    TYPE ,$CRLF        ;TYPE <CR> AND <LF>
RESREG                 ;RESTORE RD-R5
RTS PC                 ;RETURN

```

7799
7800
7801
7802
7803
7804
7805
7806

```

*****
$BTTL CRTSER - TYPE CARTRIDGE SERIAL NUMBER
; *THIS SUBROUTINE TYPES "CART. SER. NO. XXXXXXXXXX" (IN OCTAL).
; *WITH LEADING ZEROS SUPPRESSED.
*****

```

```

*****
CRTSER: JSR PC,INITSS          ;CLEAR S.S. AND PARAMETERS
        CLRB P,CS1H(R5)       ;SET 22-SECTOR FORMAT
        TSTB FORMAT           ;CHECK THE ACTUAL FORMAT
        BEQ 4S                ;BR IF 22 SECTORS
        INCB P,SECT(R5)        ;IF 20 SECTORS, READ SECTOR 1
4S: MOVB #RDDATA,P.CMND(R5)    ;SET READ COMMAND
    MOV #LSTCYL,P.CYLN(R5)    ;SET CYL = 632(OCT)
    MOVB #LSTTRK,P.TRCK(R5)   ;SET TRACK = 2
    MOV #RWBUF,P.BALO(R5)     ;SET READ BUFFER ADDRESS
    MOV #-2,P.WC(R5)          ;SET WORD COUNT TO READ 2 WORDS
    JSR PC,DRVCAL             ;READ SERIAL NO. IN BSF
    TYPE ,CART                ;TYPE "CART."
    TYPE ,SERNM               ;TYPE "SER. NO."
    MOV #RWBUF,-(SP)          ;GET POINTER FOR $DB20
    JSR PC,2*$DB20            ;CONVERT BINARY TO OCTAL
    JSR PC,2*$SUPRS           ;TYPE CART. SERIAL NO. IN OCTAL
    TYPE ,CRLF                ;TYPE <CR> AND <LF>
RTS PC                         ;RETURN

```

7825
7826

```

7827
7828
7829
7830
7831
7832
7833
7834
7835
7836 033762 010046
7837 033764 010146
7838 033766 032777 000400 145144
7839 033774 001407
7840 033776 004737 053072
7841 034002 13700 053172
7842 034006 006200
7843 034010 006200
7844 034012 000403
7845 034014 013700 005504
7846 034020 001406
7847 034022 012701 000016
7848 034026 005301
7849 034030 001376
7850 034032 005300
7851 034034 001372
7852 034036 012601
7853 034040 012600
7854 034042 000207
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865 034044 004737 053072
7866 034050 013737 053172 005506
7867 034056 042737 177000 005506
7868 034064 022737 000632 005506
7869 034072 002003
7870 034074 042737 000400 005506
7871 034102 000207
7872
7873
7874
7875
7876
7877
7878
7879 034104 104406
7880 034106 012701 064574
7881 034112 010021
7882 034114 020127 065574

```

```

*****
.SBTTL STALL - STALL FOR ST UNIT STALL TIMES
;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES.
;*WHERE A STALL TIME = 40 US. FOR AN "AVERAGE" CPU. IF BIT 8
;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
* CALL - JSR PC,STALL
*****

```

```

STALL: MOV RO, -(SP) ;SAVE RO
MOV RI, -(SP) ;SAVE RI
BIT #BIT08, SWR ;APPLY RANDOM STALL ?
SEQ 15 ;BR IF NOT RANDOM
JSR PC, $RAND ;GENERATE PSEUDO-RANDOM NUMBER
MOV $LONUM, RO ;GET IT INTO RO
ASR RO ;SCALE IT DOWN
ASR RO
BR 2$ ;GO STALL WITH RANDOM NO.
1$: MOV STALLS, RO ;GET REQUESTED NO. OF STALLS
BEQ 6$ ;RETURN IF NO STALL REQUIRED
2$: MOV #14., R1 ;SET CONSTANT FOR 40 US
4$: DEC R1 ;INNER LOOP COUNTER
BNE 4$ ;INNER LOOP BR
DEC RO ;OUTER LOOP COUNTER
BNE 2$ ;OUTER LOOP BR
6$: MOV (SP)+, R1 ;RESTORE R1
MOV (SP)+, RO ;RESTORE RO
RTS PC ;RETURN

```

```

*****
.SBTTL RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
;*THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
;*ADDRESS, AND LEAVES IT IN "CYLNDR". IT REQUIRES THE SYSMAC
;*SUBROUTINE $RAND.
* CALL - JSR PC, RNDADR
*****

```

```

RNDADR: JSR PC, $RAND ;GENERATE 2 16-BIT RANDOM NUMBERS
MOV $LONUM, CYLNDR ;GET A RANDOM NUMBER
BIC #177000, CYLNDR ;SCALE IT TO 9 BITS
CMP #632, CYLNDR ;SEE IF CYL IS TOO BIG
BGE 2$ ;BR IF CYL IS OK
BIC #BIT08, CYLNDR ;SCALE DOWN TO A VALID CYLINDER
2$: RTS PC ;RETURN

```

```

*****
* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF RO INTO ALL
*256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
*****

```

```

LODSEC: SAVREG ;SAVE RO-R5
MOV #RWBUF, R1 ;GET ADDRESS OF DATA BUF INTO R1
2$: MOV RO, (R1)+ ;PUT WORD INTO BUFFER
CMP R1, #RWBUF+512. ;SEE IF 256 WORDS WRITTEN YET

```

7883 034120 001374
7884 034122 104407
7885 034124 000207

BNE 25
RESREG
RTS PC

;BR IF NOT DONE YET
;RESTORE R0-R5
;RETURN

7886
7887
7888
7889
7890
7891
7892
7893
7894

* TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
*FC FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
*TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.

7895 034126 104406
7896 034130 015556 050004
7897 034134 105065 000705
7898
7899 034140 116500 000005
7900 034144 062700 000100
7901 034150 004737 034104
7902 034154 112765 000131 000001
7903 034162 004737 040716
7904 034166 105265 000005
7905 034172 122765 000003 000005
7906 034200 001357
7907 034202 012665 000004
7908 034206 104407
7909 034210 000207

TRKCHK: SAVREG ;SAVE R0-R5
MOV P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
CLR B P.TRCK(R5) ;CLEAR THE TRACK NO.
;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
25: MOV B P.TRCK(R5),R0 ;GET TRACK NO. INTO R0
ADD #100,R0 ;ADD 100(OCT) TO TRACK NO.
JSR PC,LODSEC ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
MOV B #WATCHK,P.COMND(R5) ;SET WRITE CHECK COMMAND
JSR PC,DRVCAL ;PERFORM THE WRITE CHECK
INCB P.TRCK(R5) ;INCREMENT THE TRACK NO.
CMP B #3,P.TRCK(R5) ;SEE IF DONE WITH ALL TRACKS
BNE 25 ;BR IF NOT DONE YET
MOV (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

7910
7911
7912
7913
7914
7915
7916

*LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
*INTO THE PATTERN 14 TABLE.

7917 034212 104406
7918 034214 012701 000010
7919 034220 012700 006726
7920 034224 004737 053072
7921 034230 013720 053172
7922 034234 013720 053170
7923 034240 005301
7924 034242 001370
7925 034244 104407
7926 034246 000207

LODP14: SAVREG ;SAVE R0-R5
MOV #8,R1 ;INIT LOOP COUNTER TO 8.
MOV #PAT14,R0 ;GET ADDRESS OF PATTERN 14 BUFFER
45: JSR PC,\$RAND ;GENERATE 2 16-BIT RANDOM NUMBERS
MOV \$L0NUM,(R0)+ ;PUT ONE NUMBER INTO PATTERN
MOV \$H1NUM,(R0)+ ;PUT OTHER NO. INTO PATTERN
DEC R1 ;SEE IF 16 WORDS LOADED YET
BNE 45 ;BR IF NOT YET
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938

*SBTTL FINADR - COMPUTE FINAL PACK ADDRESS
*THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
*COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
*WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
*P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
*PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
*THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
*DATA MISCOMPARE.


```

7939
7940 034250 104406
7941 034252 013746 005572
7942 034256 005416
7943 034260 005046
7944 034262 116616 000003
7945 034266 005066 000002
7946 034272 116566 000004 000002
7947 034300 066616 000002
7948 034304 005066 000002
7949 034310 012700 000026
7950 034314 105737 003115
7951 034320 001402
7952 034322 012700 000024
7953 034326 020016
7954 034330 101004
7955 034332 160016
7956 034334 005266 000002
7957 034340 000772
7958 034342 112637 005571
7959 034346 005046
7960 034350 116516 000005
7961 034354 066616 000002
7962 034360 005066 000002
7963 034364 122716 000003
7964 034370 101005
7965 034372 162716 000003
7966 034376 005266 000002
7967 034402 000770
7968 034404 112637 005570
7969 034410 066516 000002
7970 034414 011637 005566
7971 034420 005726
7972 034422 104407
7973 034424 000207
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988 034426 104406
7989 034430 013702 005564
7990 034434 005737 005536
7991 034440 001007
7992 034442 012700 006026
7993 034446 012746 000400
7994 034452 012701 064574

```

```

*****
FINADR: SAVREG ;SAVE R0-R5
MOV LASTWC,-(SP) ;STORE WORD COUNT
NEG (SP) ;MAKE IT POSITIVE
18$: CLR -(SP) ;MAKE ROOM ON STACK
MOV 3(SP),(SP) ;STORE NO. OF SECTORS TRANSFERRED
CLR 2(SP) ;CLEAR LOCATION ON STACK
MOV P.SECT(R5),2(SP) ;STORE STARTING SECTOR
ADD 2(SP),(SP) ;DETERMINE FINAL SECTOR ADDRESS
CLR 2(SP) ;CLEAR NO. OF TRACKS TRANSFERRED
MOV #22,R0 ;SET FOR 22 SECTORS
TSTB FORMAT ;DETERMINE THE FORMAT
SEQ 19$ ;BR IF 22 SECTORS
MOV #20,R0 ;SET FOR 20 SECTORS
19$: CMP R0,(SP) ;CHECK FOR SECTOR OVERFLOW
BHI 20$ ;NO, CHECK IF SECTOR CORRECT
SUB R0,(SP) ;DECREMENT SECTOR COUNT BY 20 OR 22
INC 2(SP) ;INCREMENT TRACKS TRANSFERRED
BR 19$ ;CHECK FOR SECTOR OVERFLOW
20$: MOV (SP)+,FINSEC ;STORE FINAL SECTOR
CLR -(SP) ;MAKE ROOM FOR TRACKS TRANSFERRED
MOV P.TRACK(R5),(SP) ;STORE STARTING TRACK
ADD 2(SP),(SP) ;DETERMINE FINAL TRACK ADDRESS
CLR 2(SP) ;CLEAR FINAL CYLINDER
21$: CMPB #3,(SP) ;CHECK FOR TRACK OVERFLOW
BHI 22$ ;NO, CHECK FINAL TRACK
SUB #3,(SP) ;DECREMENT TRACK COUNT BY 3
INC 2(SP) ;INCR CYL COUNT
BR 21$ ;CHECK FOR TRACK OVERFLOW
22$: MOV (SP)+,FINTRK ;STORE FINAL TRACK
ADD P.CYL(R5),(SP) ;CALCULATE FINAL CYLINDER
MOV (SP),FENCYL ;STORE FINAL CYLINDER
TST (SP)+ ;CLEAN OFF STACK
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

```

```

*****
;SBTTL LODBUF - LOAD THE READ/WRITE DATA BUFFER
;THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
;PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
;PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
;OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
;IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
;OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
;ENTRY, ARE LOADED INTO THE BUFFER.
*****

```

```

LODBUF: SAVREG ;SAVE R0-R5
MOV WDSXFR,R2 ;GET NO. OF WORDS
TST PATRN ;SEE IF QUICK VERIFY DATA TEST DESIRED
BNE 3$ ;BR IF NOT QUICK VERIFY
MOV #PAT00,R0 ;SET DATA PATTERN STARTING ADDRESS
MOV #256,-(SP) ;SET PATTERN WORD COUNT
MOV #RWBUR,R1 ;SET BUFFER ADDRESS

```



```

7995 034456 000463          BR      30$      ;PROCEED
7996 034460 005000          CLR     RO      ;INIT PATTERN NUMBER
7997 034462 032701 000001  4$:    BIT     #BIT0,R1 ;SEE IF THIS BIT IS SET
7998 034466 001003          BNE     6$      ;BR IF THIS BIT IS SET
7999 034470 005200          INC     RO      ;INCREMENT PATTERN NO.
8000 034472 006201          ASR     R1      ;SHIFT TO EXAMINE NEXT BIT
8001 034474 000772          SR     4$      ;BR TO CHECK NEXT BIT
8002 034476 006300          6$:    ASL     RO      ;MULTIPLY PATTERN NO. BY 32(DEC)
8003 034500 006300          ASL     RO
8004 034502 006300          ASL     RO
8005 034504 006300          ASL     RO
8006 034506 006300          ASL     RO
8007 034510 062700 006026      ADD     #PAT00,RO ;GET ADDRESS OF DESIRED PATTERN
8008 034514 012746 000020      MOV     #16,-(SP) ;SET PATTERN WORD COUNT
8009 034520 013701 005604      MOV     PMA,R1  ;SET BUFFER ADDRESS
8010 034524 005737 056216      TST     $KT11  ;SEE IF MEM MGT PRESENT
8011 034530 100036          BPL     30$      ;BR IF NOT PRESENT
8012          ;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
8013 034532 013737 003200 172354  MOV     SAVPAR,#KIPAR6 ;SET UP WORKING PAR
8014 034540 052737 000001 177572  BIS     #BIT0,#SRO ;TURN ON MEMORY MANAGEMENT
8015 034546 042701 160000      BIC     #160000,R1 ;FORCE RELOCATION THRU KIPAR6
8016 034552 052701 140000      BIS     #140000,R1
8017 034556 010003          22$:    MOV     RO,R3  ;GET A COPY OF PATTERN ADDRESS
8018 034560 011604          MOV     (SP),R4 ;INIT PATTERN WORD COUNT
8019 034562 012321          24$:    MOV     (R3)+,(R1)+ ;LOAD A DATA WORD INTO BUFFER
8020 034564 032701 020000      BIT     #BIT13,R1 ;SEE IF OVERFLOW TO NEXT PAGE
8021 034570 001405          BEQ     26$      ;BR IF NO OVERFLOW
8022 034572 062737 000200 172354  ADD     #200,#KIPAR6 ;INCREMENT PAR BY 4K FOR NEW PAGE
8023 034600 042701 020000      BIC     #BIT13,R1 ;SET PAGE = 6 AGAIN
8024 034604 005302          26$:    DEC     R2      ;DECREMENT WORD COUNTER
8025 034606 001403          BEQ     28$      ;BR IF ALL DONE
8026 034610 005304          DEC     R4      ;DECREMENT PATTERN WORD COUNT
8027 034612 001363          BNE     24$      ;BR IF NOT DONE WITH PATTERN YET
8028 034614 000760          BR     22$      ;BR TO REPEAT THE PATTERN
8029 034616 042737 000001 177572  28$:    BIC     #BIT0,#SRO ;DISABLE MEMORY MANAGEMENT
8030 034624 000410          BR     44$      ;GO TO RETURN
8031          ;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE
8032 034626 010003          30$:    MOV     RO,R3  ;GET A COPY OF PATTERN ADDRESS
8033 034630 011604          MOV     (SP),R4 ;INIT PATTERN WORD COUNT
8034 034632 012321          34$:    MOV     (R3)+,(R1)+ ;LOAD A DATA WORD INTO BUFFER
8035 034634 005302          DEC     R2      ;DECREMENT WORD COUNTER
8036 034636 001403          BEQ     44$      ;BR IF ALL DONE
8037 034640 005304          DEC     R4      ;DECREMENT PATTERN WORD COUNT
8038 034642 001373          BNE     34$      ;BR IF NOT DONE WITH PATTERN YET
8039 034644 000770          BR     30$      ;BR TO REPEAT THE PATTERN
8040 034646 005726          44$:    TST     (SP)+ ;POP THE STACK
8041 034650 104407          RESREG ;RESTORE RO-R5
8042 034652 000207          RTS     PC      ;RETURN
    
```

```

;*****
;SBTTL CMPBUF - SOFTWARE COMPARE DATA
;THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
;TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
;PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
    
```

```

8051      ;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
8052      ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
8053      ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
8054      ;*****
8055      CMPBUF: SAVREG      ;SAVE R0-R5
8056      034654 104406      MOV      #40,DH701+38.  ;RESTORE ERROR MSG PARAMS
8057      034656 112737 000040 063113      MOV      #7,DF25+2
8058      034664 012737 000007 064466      MOV      WDSXFR,R2      ;SET NO. OF WORDS
8059      034672 013702 005564      CLR      SCRACH      ;CLEAR COMPARE ERROR COUNT
8060      034676 005037 005534      JSR      PC,KEPSUP    ;STORE PREV CMND FOR POSS. PRINT
8061      034702 004737 042130      TST     PATRN      ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
8062      034706 005737 005536      BNE     3$      ;BR IF NOT
8063      034712 001007      MOV      #PAT00,R0    ;GET DATA PATTERN STARTING ADDR
8064      034714 012700 006026      MOV      #256,-(SP)  ;SET PATTERN WORD COUNT
8065      034720 012746 000400      MOV      #RWBUF,R1   ;SET BUFFER ADDRESS
8066      034724 012701 064574      BR      30$      ;PROCEED
8067      034730 000466      BR      30$
8068      034732 005000      3$: CLR      RO      ;INIT PATTERN NO.
8069      034734 005000      4$: BIT      #BIT0,R1 ;SEE IF THIS BIT IS SET
8070      034734 032701 000001      BNE     6$      ;BR IF THIS BIT IS SET
8071      034740 001003      INC     RO      ;INCREMENT PATTERN NUMBER
8072      034742 005200      ASR     R1      ;SHIFT TO EXAMINE NEXT BIT
8073      034744 006201      BR      4$      ;BR TO CHECK NEXT BIT
8074      034746 000772      6$: ASL     RO      ;MULTIPLY PATTERN NO. BY 32(DEC)
8075      034750 006300      ASL     RO
8076      034752 006300      ASL     RO
8077      034754 006300      ASL     RO
8078      034756 006300      ASL     RO
8079      034760 006300      ADD     #PAT00,RO    ;GET ADDRESS OF DESIRED PATTERN
8080      034762 062700 006026      MOV     #16,-(SP)   ;PATTERN WORD COUNT
8081      034766 012746 000020      MOV     PMA,R1     ;SET BUFFER ADDRESS
8082      034772 013701 005604      TST     $KT11     ;SEE IF MEM MGT PRESENT
8083      034776 005737 056216      BPL     30$      ;BR IF NOT PRESENT
8084      035002 100041      ;COMPARE LOOP FOR MEM MGT STARTS HERE
8085      035004 013737 003200 172354      MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
8086      035012 052737 000001 177572      BIS     #BIT0,@#SRO  ;TURN ON MEM MGT
8087      035020 042701 160000      BIC     #160000,R1  ;FORCE RELOCATION THRU KIPAR6
8088      035024 052701 140000      BIS     #140000,R1
8089      035030 010003      22$: MOV     RO,R3     ;GET A COPY OF PATTERN ADDRESS
8090      035032 011604      MOV     (SP),R4     ;INIT PATTERN WORD COUNT
8091      035034 022321      24$: CMP     (R3)+,(R1)+ ;COMPARE DATA WORD TO PATTERN WORD
8092      035036 001404      BEQ     25$      ;BR IF NO ERROR
8093      035040 013737 172354 001216      MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
8094      035046 000432      BR      35$      ;GO HANDLE ERROR
8095      035050 032701 020000      25$: BIT     #BIT13,R1 ;SEE IF OVERFLOW TO NEXT PAGE
8096      035054 001405      BEQ     26$      ;BR IF NO OVERFLOW
8097      035056 062737 000200 172354      ADD     #200,@#KIPAR6 ;INCR PAR BY 4K FOR NEW PAGE
8098      035064 042701 020000      BIC     #BIT13,R1  ;SET PAGE = 6 AGAIN
8099      035070 005302      26$: DEC     R2      ;DECREMENT WORD COUNTER
8100      035072 001002      BNE     27$      ;BR IF NOT ALL DONE YET
8101      035074 000137 035540      JMP     54$      ;ALL DONE - GET OUT
8102      035100 005304      27$: DEC     R4      ;DECREMENT PATTERN WORD COUNT
8103      035102 001354      BNE     24$      ;BR IF NOT DONE WITH PATTERN YET
8104      035104 000751      BR      22$      ;BR TO REPEAT THE PATTERN
8105      ;COMPARE LOOP FOR NO MEM MGT STARTS HERE
8106      30$: MOV     RO,R3     ;GET COPY OF PATTERN ADDRESS
           MOV     (SP),R4 ;INIT PATTERN WORD COUNT

```

8107	035112	022321			34\$:	CMP	(R3)+,(R1)+	;COMPARE DATA WORD TO PATTERN WORD
8109	035114	001002				BNE	31\$;BR IF COMPARE ERROR
8109	035116	000137	035520			JMP	40\$;JUMP IF DATA COMPARES OK
8110	035122	105037	063113		31\$:	CLRB	DH701+38.	;ADJUST DATA HEADER FOR MSG
8111	035126	012737	DC0005	064466		MOV	#5,DF25+2	;ADJUST ERROR DATA WORD COUNT
8112					;COMMON	COMPARE	ERROR HANDLER	
8113	035134	010237	001202		35\$:	MOV	R2,\$REG10	;GET WORD NO.
8114	035140	163737	005564	001202		SUB	WDSXFR,\$REG10	
8115	035146	013737	001202	005572		MOV	\$REG10, LASTWC	;GET 2'S COMP OF WORD NO.
8116	035154	004737	034250			JSR	PC,FINADR	;COMPUTE ACTUAL PACK ADRS
8117	035160	104406				SAVREG		;SAVE R0-R5
8118	035162	013700	005566			MOV	FINCYL,R0	;GET CYL
8119	035166	113701	005570			MOVB	FINTRK,R1	;GET TRACK
8120	035172	113702	005571			MOVB	FINSEC,R2	;GET SECTOR
8121	035176	004737	040612			JSR	PC,BDSRCK	;SEE IF THIS SECTOR LISTED BAD
8122	035202	104407				RESREG		;RESTORE R0-R5
8123	035204	032737	001000	005476		BIT	#BADSEC,RECODE	
8124	035212	001132				BNE	50\$;BR IF LISTED- DON'T REPORT ERROR
8125	035214	005737	005534			TST	SCRACH	;CHECK THE ERROR COUNT
8126	035220	001024				BNE	36\$;BR IF THIS IS NOT FIRST ERROR
8127	035222	105737	003134			TSTB	UBMPRS	;SEE IF UNIBUS MAP PRESENT
8128	035226	001411				BEQ	46\$;BR IF NOT
8129	035230	013737	005262	001174		MOV	CRMPHO,\$REG5	;GET CURRENT MAP REG 0
8130	035236	013737	005260	001176		MOV	CRMPLO,\$REG6	
8131	035244	104116				ERROR	116	;DATA MISCOMPARE (11/70)
8132	035246	104117				ERROR	117	
8133	035250	000401				BR	48\$	
8134	035252	104034			46\$:	ERROR	34	;TYPE HEADING FOR ERROR MSG
8135	035254	012737	177777	001174	48\$:	MOV	#-1,\$REG5	;INIT CYL NO.
8136	035262	005037	001176			CLR	\$REG6	
8137	035266	005037	001200			CLR	\$REG7	
8138	035272	005237	005534		36\$:	INC	SCRACH	;INCREMENT THE ERROR COUNT
8139	035276	032777	000001	143634		BIT	#BIT0,DSWR	;SEE IF ALL ERRORS SHOULD BE REPORTED
8140	035304	001004				BNE	38\$;BR TO REPORT ALL ERRORS
8141	035306	022737	000012	005534		CMP	#10.,SCRACH	;SEE IF 10(DEC) ERRORS YET
8142	035314	002511				BLT	54\$;BR IF ERROR LIMIT EXCEEDED
8143	035316	023737	001174	005566	38\$:	CMP	\$REG5,FINCYL	;SEE IF DIFFERENT CYL
8144	035324	001010				BNE	42\$;BR IF YES
8145	035326	123737	001176	005570		CMPB	\$REG6,FINTRK	;SEE IF DIFFERENT TRACK
8146	035334	001004				BNE	42\$;BR IF YES
8147	035336	123737	001200	005571		CMPB	\$REG7,FINSEC	;SEE IF DIFFERENT SECTOR
8148	035344	001412				BEQ	44\$;BR IF SAME PACK ADDRESS
8149	035346	013737	005566	001174	42\$:	MOV	FINCYL,\$REG5	;SET NEW PACK ADRS FOR PRINTOUT
8150	035354	113737	005570	001176		MOVB	FINTRK,\$REG6	
8151	035362	113737	005571	001200		MOVB	FINSEC,\$REG7	
8152	035370	104115				ERROR	115	;TYPE NEW PACK ADDRESS
8153	035372	005437	001202		44\$:	NEG	\$REG10	;GET WORD NO.
8154	035376	016337	177776	001204		MOV	-2(R3),\$REG11	;GET GOOD DATA
8155	035404	016137	177776	001206		MOV	-2(R1),\$REG12	;GET BAD DATA
8156	035412	013737	001202	001212		MOV	\$REG10,\$REG14	;COMPUTE PHYSICAL ADDRESS
8157	035420	005037	001210			CLR	\$REG13	
8158	035424	006137	001212			ROL	\$REG14	
8159	035430	006137	001210			ROL	\$REG13	
8160	035434	063737	005604	001212		ADD	PMA,\$REG14	
8161	035442	005537	001210			ADC	\$REG13	
8162	035446	063737	005606	001210		ADD	PMA+2,\$REG13	

```

8163 035454 010137 001214      MOV      R1,$REG15      ;GET VIRT. ADRS FOR PRINTOUT
8164 035460 162737 000002 001214  SUB      #2,$REG15
8165 035466 104063      ERROR      63          ;TYPE GOOD AND BAD DATA, MEM. ADRS.
8166 035470 032777 000100 143442  BIT      #BIT6,$SWR     ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8167 035476 001020      BNE      54$          ;BR IF JUST 1 ERROR SHOULD BE REPORTED
8168 035500 005737 005536      50$:    TST      PATRN     ;SEE IF DEFAULT DATA TEST
8169 035504 001405      BEQ      40$          ;BR IF YES
8170 035506 005737 056216      TST      $KT11        ;SEE IF MEM MGT PRESENT
8171 035512 100002      BPL      40$          ;BR IF NOT PRESENT
8172 035514 000137 035050      JMP      25$          ;GO HANDLE MEM MGT
8173 035520 005302      40$:    DEC      R2        ;DECREMENT WORD COUNTER
8174 035522 001406      BEQ      54$          ;BR IF ALL DONE
8175 035524 005304      DEC      R4          ;DECREMENT PATTERN WORD COUNT
8176 035526 001002      BNE      53$          ;BR IF NOT DONE WITH PATTERN YET
8177 035530 000137 035106      JMP      30$          ;JUMP TO REPEAT THE PATTERN
8178 035534 000137 035112      53$:    JMP      34$
8179 035540 005737 056216      54$:    TST      $KT11        ;SEE IF MEM MGT PRESENT
8180 035544 100003      BPL      56$          ;BR IF NOT PRESENT
8181 035546 042737 000001 177572  BIC      #BIT0,$SRO    ;DISABLE MEM MGT
8182 035554 005726      55$:    TST      (SP)+       ;POP THE STACK
8183 035556 104407      RESREG
8184 035560 000207      RTS      PC           ;RESTORE R0-R5
                        ;RETURN

```

```

8185
8186
8187
8188
8189
8190
8191
8192
8193
8194
8195
8196
8197

```

;*****
;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
;* IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (↑Z)
;* WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
;* A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
;* IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
;* NO ACTION IS TAKEN.
;* CALL - JSR PC,CTLOUT
;* OR - CKEXIT
;*****

```

8198 035562 122737 000001 003110  CTLOUT: CMPB     #1,TSTING     ;SEE IF CURRENTLY RUNNING TESTS
8199 035570 001077      BNE     10$          ;BR IF NOT RUNNING TESTS
8200 035572 005737 005524      TST     INTCHR      ;SEE IF ANY TTY INPUT
8201 035576 001474      BEQ     10$          ;BR IF NO INPUT
8202 035580 105737 003106      TSTB   MDFLAG      ;SEE IF DEFAULT MODE RUN
8203 035604 001446      BEQ     12$          ;BR IF YES
8204 035606 122737 000003 005524  CMPB   #003,INTCHR  ;SEE IF (↑C) TYPED
8205 035614 001033      BNE     4$          ;BR IF NOT (↑C)
8206 035616 012700 014520      MOV     #DRVTST,R0   ;SET RETURN ADDR = DRVTST
8207 035622 105737 003124      2$:    TSTB   XOVLAD     ;SEE IF XXDP CURRENTLY OVERLAID
8208 035626 001402      BEQ     6$          ;BR IF NOT
8209 035630 004737 030502      JSR     PC,GETXDP    ;RESTORE SAVED XXDP, IF NECESSARY
8210 035634 000005      6$:    RESET          ;RESET ALL DEVICES
8211 035636 005037 005524      CLR     INTCHR      ;CLEAR TTY CHAR BUFFER WORD
8212 035642 005037 001304      CLR     $TIMES      ;CLEAR THE ITER. COUNT
8213 035646 105037 003125      CLRB   XDPSVD      ;CLEAR THE XXDP SAVED FLAG
8214 035652 012737 042410 003036  MOV     #ERRHDL,A,ABNL ;RESTORE ERROR HANDLER ADDRESS
8215 035660 012706 001100      MOV     #STACK,SP    ;RESET THE STACK
8216 035664 112765 000113 000001  MOVB   #RECAL,P,CMND(R5) ;SET RECAL COMMAND
8217 035672 004737 040716      JSR     PC,DRVCL     ;DO CLEANUP RECALIBRATE
8218 035676 005037 001102      CLR     $STNM       ;CLEAR THE TEST NO.

```

```

8219 035702 000110          JMP      JRO          ;EXIT FROM TESTS
8220 035704 122737 000032 005524 4$:  CMPB    #032,INTCHR ;SEE IF (↑Z) TYPED
8221 035712 001016          BNE     7$          ;BR IF NOT (↑Z)
8222 035714 012700 016120          MOV     #INPUTP,RO  ;SET RETURN ADDR = INPUTP
8223 035720 000740          BR      2$          ;TAKE EXIT
8224 035722 122737 000003 005524 12$: CMPB    #003,INTCHR ;SEE IF (↑C) TYPED
8225 035730 001007          BNE     7$          ;BR IF NOT
8226 035732 104400 007646          TYPE   ,HLTRQD     ;TYPE "HALT REQUESTED"
8227 035736 104400 007616          TYPE   ,CNTRDY     ;TYPE "PRESS CONT WHEN RDY"
8228 035742 012700 044420          MOV     #HLTPRG,RO ;SET HALT ADDRESS
8229 035746 000725          BR      2$          ;TAKE EXIT
8230 035750 122737 000007 005524 7$:  CMPB    #007,INTCHR ;SEE IF (↑G) TYPED
8231 035756 001002          BNE     8$          ;BR IF NOT (↑G)
8232 035760 004737 027562          JSR    PC,GTSWRG   ;OPEN SOFTWARE SWR FOR MODIFICATION
8233 035764 004737 027250          JSR    PC,PREPKB   ;ENABLE KBD INPUT AGAIN
8234 035770 000207          RTS     PC          ;RETURN
8235
8236
8237

```

```

9238 ;*****
8239 ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO DISK AND PERFORM
8240 ;*A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
8241 ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
8242 ;*USED.
8243 ;*****

```

```

8244 035772 104406          WRTSEC: SAVREG      ;SAVE R0-R5
8245 ;LOAD THE R/W BUFFER WITH DATA
8246 035774 012700 064574          MOV     #RWBUF,RO  ;BUFFER ADDRESS
8247 036000 012701 000400          MOV     #400,R1    ;400(OCT) WORDS
8248 036004 010320 4$:  MOV     R3,(R0)+   ;LOAD A BUFFER WORD
8249 036006 005301          DEC     R1         ;DECR COUNTER
8250 036010 001375          BNE     4$         ;BR IF NOT DONE YET
8251 ;SET UP PARAMETERS
8252 036012 012765 064574 000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
8253 036020 012765 177400 000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
8254 ;WRITE THE DATA
8255 036026 112765 000123 000001          MOVB   #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
8256 036034 032777 000002 143076          BIT    #BIT1,JSWR   ;SEE IF WRITES INHIBITED
8257 036042 001002          BNE     6$         ;BR IF YES
8258 036044 004737 040716          JSR    PC,DRVCAL   ;WRITE THE DATA
8259 ;PERFORM WRITE CHECK
8260 036050 112765 000131 000001 6$:  MOVB   #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
8261 036056 004737 040716          JSR    PC,DRVCAL   ;PERFORM WRITE CHECK
8262 036062 104407          RESREG             ;RESTORE R0-R5
8263 036064 000207          RTS     PC          ;RETURN
8264
8265
8266

```

```

8267 ;*****
8268 ;*LDMEM1 - LOAD MEMORY WITH INCREASING NUMBERS, STARTING WITH 1.
8269 ;*LDMEM2 - LOAD MEMORY WITH REPEATED WORD, IN R3 ON ENTRY.
8270 ;*BOTH SUBROUTINES REQUIRE MEMORY ADDRESS IN PMA, PMA+2, AND WORD COUNT
8271 ;*MUST BE IN WDSXFR.
8272 ;*****
8273 036066 104406          LDMEM1: SAVREG     ;SAVE R0-R5
8274 036070 005004          CLR     R4         ;R4=0 INDICATES LDMEM1

```

```

8275 036072 012703 000001      MOV      #1,R3      ;INIT DATA WORD TO 1
8276 036076 000403      BR       LDMMO     ;PROCEED
8277 036100 104406      LDMMO2: SAVREG    ;SAVE R0-R5
8278 036102 012704 177777      MOV      #-1,R4    ;R4=177777 INDICATES LDMMO2
8279 036106 013702 005564      LDMMO:  MOV      WDSXFR,R2 ;GET NO. OF WORDS
8280 036112 013701 005604      MOV      PMA,R1    ;GET MEM ADRS
8281 036116 005737 056216      TST     $KT11     ;SEE IF MEM MGT
8282 036122 100012      BPL     6$        ;BR IF NOT
8283 036124 042701 160000      BIC     #160000,R1 ;FORCE RELOCATION THRU PAR6
8284 036130 052701 140000      BIS     #140000,R1
8285 036134 013737 003200 172354      MOV      SAVPAR,2#KIPAR6 ;INIT PAR6
8286 036142 052737 000001 177572      BIS     #BIT0,2#SRO ;TURN ON MEM MGT
8287 036150 010321      6$:  MOV      R3,(R1)+ ;LOAD A WORD INTO BUFFER
8288 036152 005704      TST     R4        ;SEE WHICH DATA DESIRED
8289 036154 001001      BNE     8$        ;BR IF NOT INCREMENTING DATA
8290 036156 005203      INC     R3        ;INCREMENT THE DATA
8291 036160 005737 056216      8$:  TST     $KT11     ;SEE IF MEM MGT
8292 036164 100010      BPL     10$       ;BR IF NOT
8293 036166 032701 020000      BIT     #BIT13,R1 ;SEE IF OVERFLOW TO NEXT PAGE
8294 036172 001405      BEQ     10$       ;BR IF NO OVERFLOW
8295 036174 062737 000200 172354      ADD     #200,2#KIPAR6 ;INCR PAR BY 4K FOR NEW PAGE
8296 036202 042701 020000      BIC     #BIT13,R1 ;SET PAGE = 6 AGAIN
8297 036206 005302      10$: DEC     R2        ;DECREMENT COUNTER
8298 036210 001357      BNE     6$        ;BR IF NOT DONE LOADING YET
8299 036212 005737 056216      TST     $KT11     ;SEE IF MEM MGT
8300 036216 100003      BPL     12$       ;BR IF NOT
8301 036220 042737 000001 177572      BIC     #BIT0,2#SRO ;TURN OFF MEM MGT
8302 036226 104407      12$: RESREG    ;RESTORE R0-R5
8303 036230 000207      RTS     PC        ;RETURN

```

```

8304
8305
8306
8307 ;:*****
8308 ;*CKMEM1 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH INCREASING
8309 ;*NUMBERS, STARTING WITH 1.
8310 ;*CKMEM2 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH REPEATED DATA WORD
8311 ;*IN R3 ON ENTRY.
8312 ;*BOTH SUBROUTINES REQUIRE MEM ADRS IN PMA, PMA+2, AND WORD COUNT MUST BE
8313 ;*IN WDSXFR.
8314 ;:*****

```

```

8315 036232 104406      CKMEM1: SAVREG    ;SAVE R0-R5
8316 036234 005004      CLR     R4        ;R4=0 INDICATES CKMEM1
8317 036236 012703 000001      MOV     #1,R3     ;INIT DATA TO 1
8318 036242 000403      BR     CKMMO     ;PROCEED
8319 036244 104406      CKMEM2: SAVREG    ;SAVE R0-R5
8320 036246 012704 177777      MOV     #-1,R4    ;R4=177777 INDICATES CKMEM2
8321 036252 013702 005564      CKMO:  MOV     WDSXFR,R2 ;GET NO. OF WORDS
8322 036256 013701 005604      MOV     PMA,R1    ;GET MEM ADRS
8323 036262 005037 005534      CLR     SCRACH
8324 036266 112737 000040 063113      MOV     #40,DH701+38. ;RESTORE ERROR MSG PARAMS
8325 036274 012737 000007 064466      MOV     #7,DF25+2
8326 036302 004737 042130      JSR    PC,REPSUP ;STORE PREV CMND FOR POSS. PRINT
8327 036306 005737 056216      TST    $KT11     ;SEE IF MEM MGT
8328 036312 100012      BPL    6$        ;BR IF NOT
8329 036314 042701 160000      BIC    #160000,R1 ;FORCE RELOCATION THRU PAR6
8330 036320 052701 140000      BIS    #140000,R1

```

8331	036324	013737	003200	172354		MOV	SAVPAR,2#KIPAR6	; INIT PAR5
8332	036332	052737	000001	177572		BIS	#BIT0,2#SR0	; TURN ON MEM MGT
8333	036340	020321			6\$:	CMP	R3,(R1)+	; COMPARE A DATA WORD
8334	036342	001575				BEQ	16\$; BR IF DATA COMPARES OK
8335								; COMPARE ERROR HANDLER
8336	036344	010237	001202			MOV	R2,\$REG10	; GET WORD NO.
8337	036350	163737	005564	001202		SUB	WDSXFR,\$REG10	
8338	036356	013737	001202	005572		MOV	\$REG10, LASTWC	; GET 2'S COMP OF WORD NO.
8339	036364	004737	034250			JSR	PC, FINADR	; COMPUTE ACTUAL PACK ADRS
8340	036370	104406				SAVREG		; SAVE R0-R5
8341	036372	013700	005566			MOV	FINCYL,R0	; GET CYL
8342	036376	113701	005570			MOVB	FINTRK,R1	; GET TRACK
8343	036402	113702	005571			MOVB	FINSEC,R2	; GET SECTOR
8344	036406	004737	040612			JSR	PC, BDSRCK	; SEE IF THIS SECTOR LISTED BAD
8345	036412	104407				RESREG		; RESTORE R0-R5
8346	036414	032737	001000	005476		BIT	#BADSEC, RECODE	
8347	036422	001145				BNE	16\$; BR IF LISTED- DON'T REPORT ERROR
8348	036424	005737	056216			TST	\$KT11	; SEE IF MEM MGT
8349	036430	100406				BMI	8\$; BR IF YES
8350	036432	105037	063113			CLRB	DH701+38.	; ADJUST DATA HEADER FOR MSG
8351	036436	012737	000005	064466		MOV	#5, DF25+2	; ADJ. ERROR DATA WORD COUNT
8352	036444	000403				BR	9\$	
8353	036446	013737	172354	001216	8\$:	MOV	2#KIPAR6,\$REG16	; SET PAR FOR PRINTOUT
8354	036454	005737	005534		9\$:	TST	SCRACH	; SEE IF FIRST ERROR IN THIS BLOCK
8355	036460	001024				BNE	10\$; BR IF NOT FIRST ERROR
8356	036462	105737	003134			TSTB	UBMPRS	; SEE IF UNIBUS MAP PRESENT
8357	036466	001411				BEQ	17\$; BR IF NOT
8358	036470	013737	005262	001174		MOV	CRMPHO,\$REG5	; CURRENT UB MAP REG 0
8359	036476	013737	005260	001176		MOV	CRMPLO,\$REG6	
8360	036504	104116				ERROR	116	; DATA MISCOMPARE (11/70)
8361	036506	104117				ERROR	117	
8362	036510	000401				BR	11\$	
8363	036512	104034			17\$:	ERROR	34	; TYPE HEADING FOR ERROR MSG
8364	036514	012737	177777	001174	11\$:	MOV	#-1,\$REG5	; INIT CYL NO.
8365	036522	005037	001176			CLR	\$REG6	
8366	036526	005037	001200			CLR	\$REG7	
8367	036532	005237	005534		10\$:	INC	SCRACH	; INCREMENT THE ERROR COUNT
8368	036536	032777	000001	142374		BIT	#BIT0,2SWR	; SEE IF ALL ERRORS SHOULD BE REPORTED
8369	036544	001004				BNE	12\$; BR TO REPORT ALL ERRORS
8370	036546	022737	000012	005534		CMP	#10., SCRACH	; SEE IF 10(DEC) ERRORS YET
8371	036554	002512				BLT	21\$; BR IF ERROR LIMIT EXCEEDED
8372	036556	023737	001174	005566	12\$:	CMP	\$REG5, FINCYL	; SEE IF DIFFERENT CYL
8373	036564	001010				BNE	14\$; BR IF YES
8374	036566	123737	001176	005570		CMPB	\$REG6, FINTRK	; SEE IF DIFFERENT TRACK
8375	036574	001004				BNE	14\$; BR IF YES
8376	036576	123737	001200	005571		CMPB	\$REG7, FINSEC	; SEE IF DIFFERENT SECTOR
8377	036604	001412				BEQ	15\$; BR IF SAME PACK ADDRESS
8378	036606	013737	005566	001174	14\$:	MOV	FINCYL,\$REG5	; SET NEW PACK ADR FOR PRINTOUT
8379	036614	113737	005570	001176		MOVB	FINTRK,\$REG6	
8380	036622	113737	005571	001200		MOVB	FINSEC,\$REG7	
8381	036630	104115				ERROR	115	; TYPE NEW PACK ADRS
8382	036632	005437	001202		15\$:	NEG	\$REG10	; GET WORD NO.
8383	036636	010337	001204			MOV	R3,\$REG11	; GOOD DATA
8384	036642	016137	177776	001206		MOV	-2(R1),\$REG12	; BAD DATA
8385	036650	013737	001202	001212		MOV	\$REG10,\$REG14	; COMPUTE PHYSICAL ADDRESS
8386	036656	005037	001210			CLR	\$REG13	


```

8387 036662 006137 001212          ROL    $REG14
8388 036666 006137 001210          ROL    $REG13
8389 036672 063737 005604 001212  ADD    PMA,$REG14
8390 036700 005537 001210          ADC    $REG13
8391 036704 063737 005606 001210  ADD    PMA+2,$REG13
8392 036712 010137 001214          MOV    R1,$REG15      ;GET VIRT ADRS FOR PRINTOUT
8393 036716 162737 000002 001214  SUB    #2,$REG15
8394 036724 104063          ERROR  63             ;TYPE GOOD AND BAD DATA
8395 036726 032777 000100 142204  BIT    #BIT6,$SWR     ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8396 036734 001022          BNE    21$           ;BR IF JUST 1
8397                                ;PROCEED WITH COMPARISONS
8398 036736 005704          16$:  TST    R4             ;SEE WHICH DATA DESIRED
8399 036740 001001          BNE    18$           ;BR IF NOT INCREASING DATA
8400 036742 005203          INC    R3             ;INCREMENT THE DATA WORD
8401 036744 005737 056216          18$:  TST    $KT11        ;SEE IF MEM MGT
8402 036750 100010          BPL    20$           ;BR IF NOT
8403 036752 032701 020000          BIT    #BIT13,R1     ;SEE IF OVERFLOW TO NEXT PAGE
8404 036756 001405          BEQ    20$           ;BR IF NO OVERFLOW
8405 036760 062737 000200 172354  ADD    #200,$#KIPAR6 ;INCR. PAR BY 4K FOR NEW PAGE
8406 036766 042701 020000          BIC    #BIT13,R1     ;SET PAGE = 6 AGAIN
8407 036772 005302          20$:  DEC    R2             ;DECR COUNTER
8408 036774 001402          BEQ    21$           ;BR IF DONE COMPARING
8409 036776 000137 036340          JMP    6$             ;CONTINUE COMPARING WORDS
8410 037002 005737 056216          21$:  TST    $KT11        ;SEE IF MEM MGT PRESENT
8411 037006 100003          BPL    22$           ;BR IF NOT
8412 037010 042737 000001 177572  BIC    #BIT0,$#SRO   ;TURN OFF MEM MGT
8413 037016 104407          22$:  RESREG  ;RESTORE R0-R5
8414 037020 000207          RTS    PC             ;RETURN
8415
8416
8417
8418
8419
8420
8421 037022 013765 005650 000002  SETUPM: MOV    FC,P.CYLN(R5) ;SET CYL
8422 037030 113765 005510 000004  MOV    FS,P.SECT(R5) ;SET SECTOR
8423 037036 113765 005654 000005  MOV    FT,P.TRACK(R5) ;SET TRACK
8424 037044 023727 005650 000624  CMP    FC,#624        ;SEE IF CYL SHOULD BE SCALED DOWN
8425 037052 003403          BLE    2$            ;BR IF NOT
8426 037054 012765 000624 000002  MOV    #624,P.CYLN(R5) ;SCALE DOWN THE CYLINDER
8427 037062 012737 064574 005600  2$:  MOV    #RWBUF,MA    ;GET MEMORY ADDRESS
8428 037070 105737 000041          TSTB   #41           ;SEE IF LOADED BY XXDP
8429 037074 001403          BEQ    4$            ;BR IF NOT
8430 037076 062737 006000 005600  4$:  ADD    #6000,MA     ;LEAVE ROOM TO SAVE XXDP
8431 037104 042737 003777 005600  BIC    #3777,MA
8432 037112 062737 004000 005600  ADD    #4000,MA
8433 037120 005037 005602          CLR    MA+2
8434 037124 000207          RTS    PC             ;RETURN
8435
8436
8437
8438
8439
8440
8441
8442
;*****
;* REFNEM - THIS SUBROUTINE PERFORMS A SUBSYSTEM COMMAND, WHILE
;* REPEATEDLY REFERENCING A NON-EXISTENT MEMORY LOCATION (760000). THIS
;* IS DONE FOR THE PURPOSE OF CAUSING UNIBUS CONTENTION DURING DISK
;* NPR'S, SINCE THE BUS IS SIEZED FOR 5-20 USEC AT EACH NEM TIME-OUT.

```


037126
037126
037132
037136
037142
037146
037154
037160
037162
037166
037172
037174
037176
037200
037202
037204
037210
037212
037214
037216
037220
037226
037230
037232
037234
037240
037242
037244
037246
037250
037252
037254
037256
037262
037266
037272
037274
037276
037276

004737 035562
105037 003113
004737 041032
012537 037160
012737 037214 000004
004737 051042
000000
004737 045412
105737 003113
001012
010304
001402
005304
001376
005737 160000
104111
000401
022626
000761
012737 000006 000004
000207

: *ALL DRIVER PARAMETERS (INCLUDING THE COMMAND) MUST BE SET IN THE
: *PARAMETER BLOCK ON ENTRY, AND R3 MUST CONTAIN THE TIMING CONSTANT
: *WHICH REGULATES THE FREQUENCY OF NEM REFERENCES, WHILE WAITING FOR
: *COMMAND COMPLETION.

REFNEM:
JSR PC,CTLOUT ;CHECK FOR (1C) OR (1Z) KBD INPUT
CLRB DONE ;CLEAR THE DONE FLAG
JSR PC,STRCMD ;STORE PREV AND CURRENT COMMANDS
MOV R5,45 ;SET PARAM BLK ADDRESS
MOV #205,2#ERRVEC ;SET TIME-OUT VECTOR ADDRESS
JSR PC,C.INIT ;CALL DRIVER
45: .WORD
65: JSR PC,W.WTCH ;CALL WATCH DOG
TSTB DONE ;DONE SET ?
BNE 265 ;BR IF YES
MOV R3,R4 ;GET COPY OF R3
BEQ 125 ;BR IF TIMER = 0
105: DEC R4 ;DECREMENT TIMER
BNE 105 ;BR IF NOT DONE TIMING
125: TST #160000 ;REFERENCE NON-EXISTENT MEMORY
ERROR 111 ;"NO NEM WHEN EXPECTED"
BR 225
205: CMP (SP)+,(SP)+ ;RESTORE THE STACK
225: BR 65 ;KEEP WAITING FOR COMMAND COMPLETION
265: MOV #6,2#ERRVEC ;RESTORE TIME-OUT VECTOR
RTS PC ;RETURN

: *INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).

INCRMA: SAVREG ;SAVE R0-R5
CLR R1
MOV P.WC(R5),R0 ;GET WORD COUNT
BNE 45 ;BR IF NOT 65,536(DEC)
INCR R1 ;SET HI BIT
BR 65 ;CONTINUE
45: NEG R0 ;GET TRUE WORD COUNT
65: CLC ;DOUBLE THE WORD COUNT TO GET BYTES
ROL R0
ROL R1
ADD R0,MA ;ADD IT TO COMPUTE NEW MA
ADC MA+2
ADD R1,MA+2
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

: *REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
: *INTO BSSOFT.

REDBSF: JSR PC,INITSS ;INIT THE S.S.

```

8499 037302 105065 000007          CLRB   P.CS1H(R5)          ;SET 22-SECTOR FORMAT
8500 037306 112765 000121 000001      MOVB   #R0DATA,P.CMND(R5) ;SET READ DATA COMMAND
8501 037314 012765 000632 000002      MOV    #632,P.CYLN(R5)   ;SET CYL = 632
8502 037322 112765 000002 000005      MOVB   #2,P.TRCK(R5)    ;SET TRACK = 2
8503 037330 012703 000012          MOV    #10,R3           ;SET FACTORY BSF SECTOR LIMIT
8504 037334 105737 003115          TSTB   FORMAT
8505 037340 001403          BEQ    6$              ;BR IF 22 SECTORS
8506 037342 105265 000004          INCB   P.SECT(R5)       ;SET STARTING FACTORY BSF SECTOR NO.
8507 037346 005203          INC    R3
8508 037350 012765 177400 000012 6$:      MOV    #-400,P.WC(R5)    ;SET WORD COUNT FOR 1 SECTOR
8509 037356 012765 004224 000010      MOV    #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
8510 037364 012737 037540 003036 8$:      MOV    #BDSCHD,A.ABNL   ;SET ERROR HANDLER FOR BSF READ
8511 037372 105037 003131          CLRB   WCEFLG          ;INIT THE ERROR FLAG
8512 037376 004737 040716          JSR    PC,DRVCAL       ;READ THE FACTORY BSF
8513 037402 105737 003131          TSTB   WCEFLG          ;SEE IF ANY ERRORS
8514 037406 001413          BEQ    12$             ;BR IF NOT
8515 037410 062765 000002 000004      ADD    #2,P.SECT(R5)    ;CHECK NEXT SECTOR
8516 037416 126503 000004          CMPB   P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
8517 037422 001360          BNE    8$              ;BR IF NOT YET
8518 037424 004737 042130          JSR    PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
8519 037430 104120          ERROR  120            ;ABORTING- BAD BSF READ
8520 037432 000137 044420          JMP    HLTPRG          ;HALT- CAN'T PROCEED
8521 037436 012765 003224 000010 12$:     MOV    #BSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
8522 037444 110365 000004          MOVB   R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
8523 037450 012703 000026          MOV    #22,R3         ;SET 22 SECTOR LIMIT
8524 037454 105737 003115          TSTB   FORMAT          ;SEE IF 22 SECTOR FORMAT
8525 037460 001402          BEQ    14$             ;BR IF YES
8526 037462 012703 000023          MOV    #19,R3         ;SET 20 SECTOR LIMIT
8527 037466 012737 037540 003036 14$:     MOV    #BDSCHD,A.ABNL   ;SET ERROR HANDLER FOR BSF READ
8528 037474 105037 003131          CLRB   WCEFLG          ;INIT THE ERROR FLAG
8529 037500 004737 040716          JSR    PC,DRVCAL       ;READ THE SOFTWARE BSF
8530 037504 105737 003131          TSTB   WCEFLG          ;SEE IF ANY ERRORS
8531 037510 001407          BEQ    16$             ;BR IF NOT
8532 037512 062765 000002 000004      ADD    #2,P.SECT(R5)    ;CHECK NEXT SECTOR
8533 037520 126503 000004          CMPB   P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
8534 037524 003760          BLE    14$             ;BR IF NOT YET
8535 037526 000736          BR     10$             ;REPORT ERROR AND ABORT
8536 037530 012737 042410 003036 16$:     MOV    #ERRHDL,A.ABNL   ;RESTORE ERROR HANDLER ADRS
8537 037536 000207          RTS    PC              ;RETURN
8538
8539 ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
8540 ;*POSSIBLE ERRORS IN READING BAD SECTORS.
8541 037540 032765 130600 000034 BDSCHD: BIT #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
8542 037546 001002          BNE    4$              ;BR IF A READ ERROR OCCURRED
8543 037550 000137 042410          JMP    ERRHDL          ;GO HANDLE OTHER ERROR
8544 037554 105237 003131 4$:      INCB   WCEFLG          ;SET ERROR FLAG
8545 037560 000137 044624          JMP    RETNML         ;TAKE NORMAL DRIVER RETURN

```

```

8546
8547
8548
8549
8550 ;*****
8551 ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
8552 ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
8553 ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
8554 ;*2 BSF'S.
8555 ;*****

```

8555	037564	104406		*YPBSF: SAVREG		:SAVE R0-R5
8556	037566	005001		CLR R1		:INIT LIMIT COUNTER
8557	037570	005004		CLR R4		:INIT BSF FILE INDICATOR
8558	037572	012700	004234	MOV #BSFACT+10,R0		:ADRS OF DATA IN FACTORY BSF
8559	037576	104400	011220	TYPE ,FACTBS		:TYPE "FACTORY"
8560	037602	104400	011245	TYPE ,BDSECT		:TYPE "BAD SECTORS"
8561	037606	104400	062660	3\$: TYPE ,DH6041		:TYPE "CYLNR TRACK SECTOR"
8562	037612	104400	001315	TYPE ,SCRLF		
8563	037616	005710		4\$: TST (R0)		:SEE IF A SECTOR LISTED HERE
8564	037620	100007		BPL BS		:BR IF YES
8565	037622	005701		TST R1		:SEE IF FILE IS EMPTY
8566	037624	001032		BNE 14\$:BR IF NOT EMPTY
8567	037626	104400	011265	TYPE ,NOFALS		:TYPE "NONE"
8568	037632	104400	001315	TYPE ,SCRLF		:TYPE <CR>,<LF>
8569	037636	000425		BR 14\$		
8570	037640	012046		9\$: MOV (R0)+,-(SP)		:GET A CYL NO.
8571	037642	104401		TYPOC		:TYPE IT
8572	037644	104400	013421	TYPE ,SPACE2		:TYPE SEPARATORS
8573	037650	011003		MOV (R0),R3		:GET TRACK AND SECTOR
8574	037652	105003		CLRB R3		
8575	037654	000303		SWAB R3		:GET THE TRACK NO.
8576	037656	010346		MOV R3,-(SP)		
8577	037660	104401		TYPOC		:TYPE IT
8578	037662	104400	013421	TYPE ,SPACE2		:TYPE SEPARATORS
8579	037666	111003		MOVB (R0),R3		:GET SECTOR NO.
8580	037670	010346		MOV R3,-(SP)		
8581	037672	104401		TYPOC		:TYPE IT
8582	037674	104400	001315	TYPE ,SCRLF		
8583	037700	005720		TST (R0)+		:INCR THE POINTER
8584	037702	005201		INC R1		:INCR THE COUNTER
8585	037704	020127	000176	CMP R1,#126.		:SEE IF LIMIT REACHED IN THIS FILE
8586	037710	002742		BLT 4\$:BR IF NOT YET
8587	037712	005704		14\$: TST R4		:SEE IF BOTH FILES TYPED YET
8588	037714	001006		BNE 18\$:BR IF NOT YET
8589	037716	005204		INC R4		:SET INDICATOR FOR SOFT. FILE
8590	037720	012700	003234	MOV #BSOFT+10,R0		:ADRS OF DATA IN SOFTWARE BSF
8591	037724	104400	011233	TYPE ,SOFTBS		:TYPE "SOFTWARE BAD SECTORS :"
8592	037730	000726		BR 3\$:GO TYPE SOFT. BSF
8593	037732	104407		18\$: RESREG		:RESTORE R0-R5
8594	037734	000207		RTS PC		:RETURN

```

8595
8596
8597
8598
8599
8600
8601
8602 037736 104406
8603 037740 016500 000002
8604 037744 116501 000005
8605 037750 116502 000004
8606 037754 005003
8607 037756 026500 000030
8608 037762 001006
8609 037764 126501 000027
8610 037770 001003

```

:*****
:FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
:*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
:*****

FINMEM: SAVREG		:SAVE R0-R5
MOV P.CYLN(R5),R0		:GET ORIG. CYL NO.
MOVB P.TRCK(R5),R1		:GET TRACK NO.
MOVB P.SECT(R5),R2		:SECTOR NO.
CLR R3		:INIT SECTOR COUNT TO 0
6\$: CMP P.DCYL(R5),R0		:COMPARE ORIG. AND CURRENT CYLS
BNE BS		:BR IF NOT EQUAL
CMPB P.DTS+1(R5),R1		:COMPARE TRACKS
BNE BS		:BR IF NOT EQUAL

```

8611 037772 126502 000026
8612 037776 001404
8613 040000 005203
8614 040002 004737 040022
8615 040006 000763
8616 040010 000303
8617 040012 010337 005564
8618 040016 104407
8619 040020 000207
8620
8621
8622
8623
8624
8625
8626
8627
8628 040022 012704 000025
8629 040026 105737 003115
8630 040032 001402
8631 040034 012704 000023
8632 040040 005202
8633 040042 020204
8634 040044 003407
8635 040046 005002
8636 040050 005201
8637 040052 020127 000002
8638 040056 003402
8639 040060 005001
8640 040062 005200
8641 040064 000207
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651 040066 104406
8652 040070 004737 037736
8653 040074 016500 000030
8654 040100 116501 000027
8655 040104 116502 000026
8656 040110 004737 040022
8657 040114 010065 000002
8658 040120 110165 000005
8659 040124 110265 000004
8660 040130 013703 005564
8661 040134 062703 000400
8662 040140 060365 000012
8663 040144 005004
8664 040146 006103
8665 040150 006104
8666 040152 060337 005604

```

```

CMPB P.DTS(R5),R2 ;COMPARE SECTORS
BEQ 12$ ;BR IF ADRS ARE EQUAL
8$: INC R3 ;INCR SECTOR COUNT
JSR PC,INCRSC ;INCR PACK ADRS BY 1 SECTOR
BR 6$ ;GO COMPARE ADDRESSES
12$: SWAB R3 ;COMPUTE NO. OF WORDS XFERRED
MOV R3,WDSXFR ;STORE IT
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

```

```

*****
*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
*IN R2.
*****

```

```

INCRSC: MOV #21,R4 ;SET SECTOR LIMIT
TSTB FORMAT ;SEE IF 22 SECTOR FORMAT
BEQ 4$ ;BR IF YES
MOV #19.,R4 ;SET SECTOR LIMIT
4$: INC R2 ;INCR. SECTOR NO.
CMP R2,R4 ;SEE IF SECTOR LIMIT EXCEEDED
BLE 6$ ;BR IF NOT
CLR R2 ;SET SECTOR = 0
INC R1 ;INCR. TRACK NO.
CMP R1,#2 ;SEE IF TRACK LIMIT EXCEEDED
BLE 6$ ;BR IF NOT
CLR R1 ;SET TRACK = 0
INC R0 ;INCR. CYLINDER NO.
6$: RTS PC ;RETURN

```

```

*****
*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
*THIS SUBROUTINE UPDATES PMA,PMA+2 MEM MGT REGS UNIBUS MAP,P.BALO(R5),
*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
*TERMINATED BY A BAD SECTOR ERROR (BSE).
*****

```

```

MIDXFR: SAVREG ;SAVE R0-R5
JSR PC,FINMEM ;COMPUTE NO. OF WORDS XFERRED
MOV P.DCYL(R5),R0 ;GET CYL NO.
MOVB P.DTS+1(R5),R1 ;GET TRACK NO.
MOVB P.DTS(R5),R2 ;GET SECTOR NO.
JSR PC,INCRSC ;INCREMENT PACK ADRS TO NEXT SECTOR
MOV R0,P.CYLN(R5) ;UPDATE CYLINDER
MOVB R1,P.TRCK(R5) ;UPDATE TRACK
MOVB R2,P.SECT(R5) ;UPDATE SECTOR
MOV WDSXFR,R3 ;GET NO. OF WORDS XFERRED
ADD #400,R3 ;SKIP BAD SECTOR
ADD R3,P.WC(R5) ;UPDATE P.WC(R5)
CLR R4 ;GET BYTES XFERRED
ROL R3
ROL R4
ADD R3,PMA ;UPDATE PMA,PMA+2

```

```

8667 040156 005537 005606      AUC      PMA+2
8668 040162 060437 005606      ADD      R4,PMA+2
8669 040166 013765 005604 000010    MOV      PMA,P.BALO(R5) ;UPDATE P.BALO(R5)
8670 040174 013700 005606      MOV      PMA+2,RO
8671 040200 042700 177774      BIC      #177774,RO
8672 040204 150065 000007      BISB     RO,P.BAHI(R5) ;UPDATE P.BAHI(R5)
8673 040210 005737 056216      TST      $K11 ;SEE IF MEM MGT
8674 040214 100002      BPL      16$
8675 040216 004737 030300      JSR      PC,PREPAR ;UPDATE MEM MGT AND UNIBUS MAP
8676 040222 104407      RESREG   ;RESTORE RO-R5
8677 040224 000207      RTS      PC ;RETURN

```

```

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

```

```

*****
;SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
;GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
*****

```

```

SVPRMS: SAVREG ;SAVE RO-R5
        MOV      #PARMO+2,RO ;ADRS OF PARAMS
        MOV      #SAVPRS,R1 ;ADRS OF SAVE AREA
        MOV      P.WC(R5),WDSXFR ;GET WORD COUNT
        NEG      WDSXFR ;MAKE IT POSITIVE
        MOV      #RWBUF,PMA ;INIT PMA TO RWBUF
        CLR      PMA+2 ;INIT PMA+2 TO 0
        TST      PATRN ;SEE IF DEFAULT DATA TEST
        BEQ      GTO ;BR IF YES
        MOV      MA,PMA ;SET PMA=MA
        MOV      MA+2,PMA+2 ;SET PMA+2=MA+2
        BR      GTO

GTPRMS: SAVREG ;SAVE RO-R5
        MOV      #SAVPRS,RO ;ADRS OF SAVE AREA
        MOV      #PARMO+2,R1 ;ADRS OF PARAMS
        MOV      #5,R2 ;SET FOR 5 WORDS
6$:     MOV      (RO)+,(R1)+ ;MOVE A WORD
        DEC      R2 ;SEE IF DONE YET
        BNE     6$ ;BR IF NOT YET
        RESREG ;RESTORE RO-R5
        RTS     PC ;RETURN

```

```

*****
;TRANSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS, IF
;ENCOUNTERED. THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
;ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED.
*****

```

```

TRANSFR: MOV      R4,-(SP) ;SAVE R4 ON STACK
        CLR      R4 ;CLEAR BSE ERROR INDICATOR
        CLRB     WCEFLG ;INIT WCE ERROR FLAG TO 0
4$:     JSR      PC,DRVCAL ;PERFORM THE S.S. FUNCTION
        BIT      #WCEERR,RECODE ;SEE IF WCE ERROR OCCURRED
        BEQ     5$ ;BR IF NOT
        MOVB     #1,WCEFLG ;SET WCE ERROR FLAG
5$:     BIT      #BSEERR,RECODE ;SEE IF BSE ERROR OCCURRED
        BEQ     6$ ;BR IF NOT

```

```

8723 040402 005204
8724 040404 004737 040066
8725 040410 005765 000012
8726 040414 001355
8727 040416 005704
8728 040420 001411
8729 040422 004737 040310
8730 040426 004737 040226
8731 040432 005737 056216
8732 040436 100002
8733 040440 004737 030300
8734 040444 012604
8735 040446 000207

```

```

INC R4 ; INCR BSE ERROR INDICATOR
JSR PC,MIDXFR ; UPDATE PARAMS TO RESUME XFER
TST P.WC(R5) ; SEE IF ENTIRE XFER IS COMPLETED
BNE 4$ ; BR IF NOT
TST R4 ; SEE IF ANY BSE ERRORS OCCURRED
BEQ 8$ ; BR IF NOT
JSR PC,GTPRMS ; RESTORE ORIGINAL PARAMS OF XFER
JSR PC,SVPRMS ; RESTORE WDSXFR,PMA,PMA+2
TST SKT11 ; SEE IF MEM MGT PRESENT
RPL R5 ; BR IF NOT
JSR PC,PREPAR ; PREPARE MEM MGT AND U.M.
MOV (SP)+,R4 ; RESTORE R4
RTS PC ; RETURN

```

```

8736
8737
8738
8739
8740
8741
8742
8743
8744
8745
8746
8747
8748
8749

```

```

*****
SBTTL SEARCH BAD SECTOR TABLES ROUTINE
*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
*
*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
*****

```

```

8750 040450 010237 001266
8751 040454 010337 001270
8752 040460 012637 001272
8753 040464 011402
8754 040466 012437 001264
8755 040472 062737 001000 001264
8756 040500 005003
8757 040502 062702 000010
8758 040506 005712
8759 040510 100430
8760 040512 020012
8761 040514 001406
8762 040516 062702 000004
8763 040522 020237 001264
8764 040526 002021
8765 040530 000766
8766 040532 005722
8767 040534 011237 001302
8768 040540 042737 174377 001302
8769 040546 123701 001303
8770 040552 001402
8771 040554 005722
8772 040556 000753
8773 040560 005203
8774 040562 012246
8775 040564 042716 177700
8776 040570 000746
8777 040572 010346
8778 040574 013702 001266

```

```

SRHTBS: MOV R2,$TMP2
MOV R3,$TMP3
MOV (SP)+,$TMP4 ; STORE RETURN CONTENTS OF R4
MOV (R4),R2 ; GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
MOV (R4)+,$TMP1 ; SETUP FOR LENGTH OF BAD SECTOR TABLE
ADD #1000,$TMP1
CLR R3 ; CLEAR R3 FOR COUNT
ADD #10,R2 ; SET R2 FOR POINTER TO CYLINDER ENTRY
1$: TST (R2) ; TEST IF ALL ONES
BMI 5$ ; YES-DONE
CMP R0,(R2) ; TEST IF BAD SECTOR IN PRESENT CYL
BEQ 3$ ; YES-GO CHECK TRACK
ADD #4,R2 ; ELSE BUMP POINTER
CMP R2,$TMP1 ; TEST IF OUT OF TABLE
BGE 5$ ; YES-EXIT
BR 1$ ; LOOP
3$: TST (R2)+ ; TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
MOV (R2),$TMP10 ; GET TRK/SEC WORD
BIC #174377,$TMP10 ; CLEAR ALL BUT TRACK
CMPB $TMP10+1,R1 ; CHECK IF BAD SECTOR IN THIS TRACK
BEQ 4$ ; YES - GO PUT SECTOR NUMBER ON STACK
TST (R2)+ ; ELSE BUMP POINTER TO NEXT CYL WORD
BR 1$ ; GO TEST NEXT CYL
4$: INC R3 ; BUMP BAD SECTOR COUNT
MOV (R2)+,-(SP) ; PUT TRK/SEC WORD ON STACK
BIC #177700,-(SP) ; CLEAR ALL BUT SECTOR NUMBER
BR 1$ ; GO CHECK REST OF FILE
5$: MOV R3,-(SP) ; EXIT - PUT NUMBER OF BAD
MOV $TMP2,R2 ; SECTORS ON STACK-RESTORE

```

```

8779 040600 013703 001270      MOV      STMP3,R3      ;REGISTERS
8780 040604 013746 001272      MOV      STMP4,-(SP)  ;PUT RETURN ON STACK
8781 040610 000204      RTS      R4           ;RETURN
8782                                     ;*****
8783 .SBTTL BAD SECTOR CHECK ROUTINE
8784 ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
8785 ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
8786 ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
8787 ;*IS NOT LISTED THE FLAG IS RESET.
8788                                     ;*****
8789 040612 104406      BDSRCK: SAVREG
8790 040614 012737 004224 040626      MOV      #BSFACT,1$   ;SET TABLE TO SEARCH
8791 040622 004437 040450 2$:      JSR      R4,SPHTBS    ;GO SEARCH IT
8792 040626 000000 1$:      .WORD
8793 040630 012603      MOV      (SP)+,R3    ;GET NUMBER OF BAD SECTORS, IF ANY
8794 040632 001015      BNE      6$          ;IF ANY, GO TEST WHICH ONES
8795 040634 023727 040626 003224 7$:      CMP      1$,#BSSOFT  ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
8796 040642 001404      BEQ      3$          ;IF YES-EXIT
8797 040644 012737 003224 040626      MOV      #BSSOFT,1$  ;SET OTHER TABLE FOR SEARCH
8798 040652 000763      BR       2$          ;GO SEARCH IT
8799 040654 042737 001000 005476 3$:      BIC      #BADSEC,RECODE ;CLEAR BAD SECTOR BIT
8800 040662 104407 4$:      RESREG
8801 040664 000207      RTS      PC          ;RETURN
8802 040666 022602 6$:      CMP      (SP)+,R2    ;THIS SECTOR IN TABLE?
8803 040670 001403      BEQ      8$          ;YES-GO SET BIT & EXIT
8804 040672 005303      DEC      R3         ;DECREMENT BAD SECTOR COUNT
8805 040674 001374      BNE      6$          ;IF NOT ZERO-CHECK NEXT ENTRY
8806 040676 000756      BR       7$          ;ELSE GO SEARCH OTHER TABLE
8807 040700 052737 001000 005476 8$:      BIS      #BADSEC,RECODE ;SET BAD SECTOR BIT
8808 040706 005303 9$:      DEC      R3         ;CLEAR STACK OF OTHER BAD SECTOR
8809 040710 001764      BEQ      4$          ;NUMBER
8810 040712 005726      TST      (SP)+
8811 040714 000774      BR       9$          ;EXIT

```

```

8812
8813
8814
8815                                     ;*****
8816 .SBTTL CALL DRIVER ROUTINE
8817 ;*ENTRY JSR PC,DRVCAL
8818 ;* WITH R5 POINTING TO PARAMETER BLOCK
8819 ;*RETURN RTS PC
8820 ;*
8821 ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
8822 ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
8823 ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
8824 ;*BLOCK WHEN THE ROUTINE IS CALLED.
8825 ;*
8826 ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
8827 ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
8828 ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
8829 ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
8830 ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
8831                                     ;*****
8832 040716 004737 033762      DRVCAL: JSR      PC,STALL    ;PERFORM A STALL IF REQUIRED
8833 040716 004737 035562      JSR      PC,CTLOUT   ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
8834 040722

```



```

8835 040726 105037 003113      CLRB      DONE      ;CLEAR DONE FLAG
8836 040732 105037 003127      CLRB      DRNAFG    ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
8837 040736 004737 041032      JSR       PC,STRCMD ;STORE PREV AND CURRENT COMMANDS
8838 040742 010537 040752      MOV       R5,4$     ;GET PARAM BLOCK ADDRESS
8839 040746 004737 051042      JSR       PC,C.INIT ;CALL DRIVER
8840 040752 000000          4$: .WORD          ;P.B. ADDRESS GOES HERE
8841 040754 004737 045412      JSR       PC,W.WTCH ;CALL WATCH DOG
8842 040760 105737 003113      TSTB     DONE      ;DONE SET?
8843 040764 001773          6$: BEQ          ;NO-LOOP
8844 040766 105737 003127      TSTB     DRNAFG    ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
8845 040772 001416          BEQ      12$      ;BR IF NOT
8846 040774 105037 003113      CLRB     DONE      ;CLEAR DONE FLAG
8847 041000 105037 003127      CLRB     DRNAFG    ;CLEAR DRIVE NOT AVAIL FLAG
8848 041004 010537 041014      MOV      R5,8$     ;GET PARAMETER BLOCK ADDRESS
8849 041010 004737 051042      JSR      PC,C.INIT ;RE-ISSUE THE COMMAND
8850 041014 000000          9$: .WORD          ;P.B. ADDRESS GOES HERE
8851 041016 004737 045412      JSR      PC,W.WTCH ;CALL WATCH DOG
8852 041022 105737 003113      TSTB     DONE      ;DONE SET ?
8853 041026 001773          10$: BEQ         ;NO - LOOP
8854 041030 000207          12$: RTS      PC   ;YES-RETURN
8855
8856
8857
8858
8859
8860
8861 041032 104406          ;*****
8862 041034 012701 005224      ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
8863 041040 012700 005240      ;*****
8864          STRCMD: SAVREG      ;SAVE R0-R5
8865          MOV      #PRVCMD,R1 ;ADDR OF PREV COMMAND STORAGE
8866          MOV      #COMSTR,R0 ;ADDR OF CURRENT COMMAND STORAGE
8867          ;STORE PREVIOUS COMMAND
8868          MOV      (R0)+,(R1)+
8869          MOV      (R0)+,(R1)+
8870          MOV      (R0)+,(R1)+
8871          MOV      (R0)+,(R1)+
8872          MOV      (R0)+,(R1)+
8873          MOV      (R0)+,(R1)+
8874          MOV      (R0)+,(R1)+
8875          MOV      (R0)+,(R1)+
8876          MOV      (R0)+,(R1)+
8877          ;STORE CURRENT COMMAND
8878          4$: MOV      #COMSTR,R1
8879          MOV      (R5)+,(R1)+ ;R5 POINTS TO DRIVER PARAM BLK
8880          MOV      (R5)+,(R1)+
8881          MOV      (R5)+,(R1)+
8882          MOV      (R5)+,(R1)+
8883          MOV      (R5)+,(R1)+
8884          MOV      (R5)+,(R1)+
8885          RESREG
8886          RTS      PC   ;RESTORE R0-R5
8887
8888
8889
8890          ;*****

```



```

8891 .SBTTL DRIVE ERROR FREE RETURN ROUTINE
8892 ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
8893 ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
8894 ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
8895 ;*ROUTINE.
8896 ;*****
8897 041142 152737 000377 003113 ERRFRE: SISB #377,DONE ;SET THE DONE FLAG
8898 041150 032737 100000 005476 BIT #ANYDER,RECODE ;TEST IF ANY DATA ERROR
8899 041156 001403 SEQ 2$ ;IF NO - DO ERROR RECOVERY PRINT TEST
8900 041160 105037 003116 CLRB ERRCNT ;CLEAR ERROR COUNT
8901 041164 000413 BR 1$ ;EXIT
8902 041166 105737 003116 2$: TSTB ERRCNT ;CHECK IF ANY ERRORS HAVE OCCURRED
8903 041172 001410 SEQ 1$ ;NO - SKIP TO EXIT
8904 041174 005037 001174 CLR $REG5
8905 041200 113737 003116 001174 MOVB ERRCNT,$REG5 ;GET RETRY COUNT
8906 041206 104101 ERROR 101 ;PRINT RETRY SUCCESSFUL MESSAGE
8907 041210 105037 003116 CLRB ERRCNT ;CLEAR ERROR COUNT
8908 041214 005037 005476 1$: CLR RECODE ;CLEAR RECOVERY FLAGS
8909 041220 000207 RTS PC ;RETURN
8910 ;*****
8911 .SBTTL TYPE ERROR ROUTINE
8912 ;*ENTRY - JSR PC,TYPERR
8913 ;*RETURN - RTS PC
8914 ;*
8915 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
8916 ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
8917 ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
8918 ;*THE ERROR.
8919 ;*****
8920 041222 104406 TYPERR: SAVREG
8921 041224 105237 003120 INCB DRVERS ;INCR ERROR COUNT FOR THIS DRIVE
8922 041230 032737 000040 005672 BIT #BITS,CS ;SEE IF DRIVE SHOULD BE DROPPED
8923 ; IF ERROR LIMIT EXCEEDED
8924 041236 001412 BEQ 9$ ;BR IF NOT
8925 041240 123727 003120 000024 CMPB DRVERS,#20. ;SEE IF 20(DEC) ERRORS EXCEEDED
8926 041246 002406 BLT 9$ ;BR IF NOT
8927 041250 105037 003120 CLRB DRVERS ;CLEAR DRIVE ERROR COUNT
8928 041254 104400 011077 TYPE ,DROPDR ;TYPE "DROPPING DRIVE"
8929 041260 000137 017436 JMP NEWDRV ;PROCEED TO TEST NEXT DRIVE
8930 041264 032777 020000 137646 9$: BIT #SW13,DSWR ;INHIBIT ERROR TYPEOUTS?
8931 041272 001402 BEQ 6$ ;BR IF NO
8932 041274 000137 041700 JMP 20$
8933 041300 005000 6$: CLR R0 ;CLR R0 FOR ERROR NUMBER
8934 041302 005005 CLR R5 ;INIT INDENT INDICATOR
8935 041304 005105 COM R5
8936 041306 113700 001114 MOVB $ITEMB,R0 ;ENTER ERROR NUMBER
8937 041312 005300 DEC R0 ;FORM INDEX FOR ERROR TABLE
8938 041314 006300 ASL R0
8939 041316 006300 ASL R0
8940 041320 006300 ASL R0
8941 041322 062700 001400 1$: ADD #ERRTB,R0 ;FORM ADDRESS OF ERROR ENTRY
8942 041326 012037 041346 MOV (R0)+,2$ ;GET EM POINTER
8943 041332 001406 BEQ 3$ ;BRANCH IF THERE ISN'T ONE
8944 041334 104400 013405 TYPE ,CR2LF
8945 041340 104400 056640 TYPE ,AS2SP2 ;TYPE "** "
8946 041344 104400 TYPE ;TYPE ERROR MESSAGE (EM)

```

```

8947 041346 000000          2$:      .WORD      0          ;EM POINTER GOES HERE
8948 041350 012037 041524    3$:      MOV        (R0)+,4$      ;GET DH POINTER
8949 041354 001467          BEQ        5$          ;BR IF THERE ISN'T ONE
8950 041356 104400 001315    TYPE      , $CRLF
8951 041362 104400 056626    TYPE      , TSTMSG      ;TYPE " TEST "
8952 041366 013746 001102    MOV        $TSTNM,-(SP) ;GET TEST NO. ON STACK
8953 041372 104402          TYPOS
8954 041374          002      .BYTE      2          ;2 DIGITS
8955 041375          000      .BYTE      0          ;SUPPRESS LEADING ZEROS
8956 041376 104400 013405    TYPE      , CR2LF
8957 041402 032777 010000 137530 BIT        #BIT12, $SWR   ;REPORT DESCRIPTION ONLY ?
8958 041410 001133          BNE        20$        ;BR IF YES
8959 041412 104400 061762    TYPE      , DH105      ;TYPE "PREVIOUS COMMAND :"
8960 041416 104400 001315    TYPE      , $CRLF
8961 041422 104400 062154    TYPE      , DH101+10    ;TYPE PREV COMMAND HEADER
8962 041426 104400 001315    TYPE      , $CRLF
8963 041432 012701 000006    MOV        #6, R1      ;SIX COMMAND VALUES
8964 041436 012702 001236    MOV        # $REG26, R2 ;STARTING ADDR OF PREV CMND VALUES
8965 041442 012246          30$:     MOV        (R2)+,-(SP) ;PUT A WORD ON STACK
8966 041444 104401          TYPOC
8967 041446 104400 013421    TYPE      , SPACE2     ;TYPE SEPARATORS
8968 041452 005301          DEC        R1          ;SEE IF 7 VALUES TYPED YET
8969 041454 001372          BNE        30$        ;BR IF NOT
8970 041456 104400 001315    TYPE      , $CRLF
8971 041462 104400 013421    TYPE      , SPACE2     ;INDENT
8972 041466 104400 062233    TYPE      , DH102      ;TYPE HDR FOR BA DATA
8973 041472 104400 001315    TYPE      , $CRLF
8974 041476 104400 013421    TYPE      , SPACE2     ;INDENT
8975 041502 012246          MOV        (R2)+,-(SP) ;TYPE PREV. HI BA BITS
8976 041504 104401          TYPOC
8977 041506 104400 013421    TYPE      , SPACE2     ;TYPE PREV. LO BA BITS
8978 041512 011246          MOV        (R2),-(SP)
8979 041514 104401          TYPOC
8980 041516 104400 013405    TYPE      , CR2LF
8981 041522 104400          TYPE
8982 041524 000000          4$:     .WORD      0          ;TYPE DATA HEADER
8983 041526 104400 001315    TYPE      , $CRLF      ;DH POINTER GOES HERE
8984 041532 005005          CLR        R5
8985 041534 032777 010000 137376 5$:     BIT        #BIT12, $SWR ;INIT INDENT INDICATOR
8986 041542 001056          BNE        20$        ;REPORT DESCRIPTION ONLY ?
8987 041544 012001          MOV        (R0)+, R1   ;BR IF YES
8988 041546 001454          BEQ        20$        ;GET DT POINTER
8989 041550 012000          MOV        (R0)+, R0   ;BRANCH IF THERE ARE NONE
8990 041552 012002          MOV        (R0)+, R2   ;GET DF POINTER
8991 041554 112003          10$:    MOV        (R0)+, R3    ;STORE NUMBER OF DH'S
8992 041556 105720          TSTB      (R0)+       ;GET & STORE NUMBER OF DATA WORDS
8993 041560 005703          TST        R3         ;BUMP PAST FORMAT WORD
8994 041562 001417          BEQ        14$        ;TEST IF ANY DATA FOR THIS HEADER
8995 041564 005705          TST        R5         ;NO - SKIP DATA PRINT
8996 041566 001002          BNE        11$        ;SEE IF SHOULD INDENT
8997 041570 104400 013421    TYPE      , SPACE2     ;BR IF NOT
8998 041574 013146          11$:    MOV        2(R1)+,-(SP) ;INDENT
8999 041576 104401          TYPOC                ;PUT FIRST DATA WORD ON STACK
9000 041600 005303          DEC        R3         ;TYPE IT
9001 041602 001403          BEQ        12$        ;MORE DATA WORDS
9002 041604 104400 013421    TYPE      , SPACE2     ;NO-BRANCH
                                ;TYPE SEPARATORS
    
```

```

9003 041610 000771          BR      11$      ;LOOP
9004 041612 005702          12$:   TST      R2      ;SEE IF <CR>,<LF> NEEDED
9005 041614 001402          BEQ      14$      ;BR IF NOT
9006 041616 104400 001315   TYPE    , $CRLF  ;TYPE IT
9007 041622 005302          14$:   DEC      R2      ;MORE DH'S?
9008 041624 003425          BLE      20$      ;NO-BRANCH
9009 041626 012037 041660   15$:   MOV      (R0)+,16$ ;GET NEXT DH POINTER
9010 041632 105710          TSTB    (R0)      ;SEE IF ANY DATA FOR HDR
9011 041634 001004          BNE     34$      ;BR IF YES
9012 041636 104400 001315   TYPE    , $CRLF  ;SKIP EXTRA LINE
9013 041642 005005          CLR     R5      ;RE-INIT INDENT INDICATOR
9014 041644 000404          BR      36$
9015 041646 005105          34$:   COM     R5      ;COMPLEMENT INDENT INDICATOR
9016 041650 001002          BNE     36$      ;BR IF NO INDENT REQUIRED
9017 041652 104400 013421   TYPE    , SPACE2 ;INDENT
9018 041656 104400          36$:   TYPE    DH      ;TYPE DH
9019 041660 000000          16$:   .WORD   0      ;DH POINTER GOES HERE
9020 041662 104400 001315   TYPE    , $CRLF
9021 041666 105710          TSTB    (R0)      ;TYPE A DT?
9022 041670 001331          BNE     10$      ;YES-BRANCH
9023 041672 062700 000002   ADD     #2,R0     ;INCREMENT DF POINTER
9024 041676 000751          BR      14$      ;SEE IF END OF DF BLOCK
9025 041700 104407          20$:   RESREG
9026 041702 000207          RTS     PC
9027
9028 ;*****
9029 ;SBTTL CONTROLLER ERROR REPORTER ROUTINE
9030 ;*ENTRY:      JSR PC, CONERR
9031 ;*RETURN:     RTS PC
9032 ;*
9033 ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
9034 ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
9035 ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
9036 ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
9037 ;*AT THIS TIME.
9038 ;*****
9038 041704 104406          CONERR: SAVREG ;SAVE R0-R5
9039 041706 152737 000377 003113 BISH    #377,DONE ;SET DONE FLAG
9040 041714 105237 003116          INCB    ERRCNT   ;INCREMENT ERROR COUNT
9041 041720 004737 044634          JSR    PC, TOPROC ;LOAD RK REGS INTO $REGS
9042 041724 032737 000001 003042 BIT     #BIT0,E.CONT ;ERROR 0?
9043 041732 001402          BEQ     1$      ;NO-BRANCH
9044 041734 104064          ERROR  64      ;CLEAR CONT DID NOT CLEAR ERROR
9045 041736 000470          BR      7$
9046 041740 032737 000002 003042 1$:   BIT     #BIT1,E.CONT ;ERROR 1?
9047 041746 001402          BEQ     2$      ;NO-BRANCH
9048 041750 104065          ERROR  65      ;NO ATTENTION IN ATTENTION SUM REG
9049 041752 000462          BR      7$
9050 041754 032737 000004 003042 2$:   BIT     #BIT2,E.CONT ;ERROR 2?
9051 041762 001407          BEQ     3$      ;NO-BRANCH
9052 041764 105737 003127          TSTB    DRNAFG   ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
9053 041770 001402          BEQ     15$     ;BR IF NOT
9054 041772 105237 003130          INCB    REISSU   ;SET FLAG TO RE-ISSUE COMMAND
9055 041776 104066          15$:   ERROR  66      ;UNSOLICITED ATTENTION
9056 042000 000447          BR      7$
9057 042002 032737 000010 003042 3$:   BIT     #BIT3,E.CONT ;ERROR 3?
9058 042010 001402          BEQ     4$      ;NO-BRANCH

```

M15

DZR6N-B - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB CONTROLLER ERROR REPORTER ROUTINE

MACY11 27(732) 03-NOV-76 21:43 PAGE 195

```

9059 042012 104067 ERROR 67 ;UNEXPECTED DATA TYPE ERROR
9060 042014 000441 BR 7$
9061 042016 032737 000020 003042 4$: BIT #BIT4,E.CONT ;ERROR 4?
9062 042024 001402 BEQ 5$ ;NO-BRANCH
9063 042026 104070 ERROR 70 ;ATTENTION DID NOT RESET WITH CLEAR
9064 042030 000433 BR 7$
9065 042032 032737 000040 003042 5$: BIT #BIT5,E.CONT ;ERROR 5?
9066 042040 001402 BEQ 6$ ;NO-BRANCH
9067 042042 104071 ERROR 71 ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
9068 042044 000425 BR 7$
9069 042046 032737 000400 003042 6$: BIT #BIT8,E.CONT
9070 042054 001401 BEQ 8$
9071 042056 104072 ERROR 72 ;DATA LATE WHEN UNLOADING HEADER
9072 042060 032737 001000 003042 8$: BIT #BIT9,E.CONT
9073 042066 001401 BEQ 9$
9074 042070 104073 ERROR 73 ;CONTROLLER ERROR DURING DRIVER SERVICE
9075 042072 032737 002000 003042 9$: BIT #BIT10,E.CONT
9076 042100 001401 BEQ 10$
9077 042102 104074 ERROR 74 ;DRIVE DETECTED PARITY ERROR
9078 042104 032737 100000 003042 10$: BIT #BIT15,E.CONT
9079 042112 001401 BEQ 11$
9080 042114 104052 ERROR 52 ;MULTIPLE DRIVE SELECT
9081 042116 104075 ERROR 75 ;UNDEFINED ERROR
9082 042120 005037 003042 7$: CLR E.CONT ;CLEAR CONTROLLER ERROR WORD
9083 042124 000137 044434 JMP BGNRTY ;GO DO RETRY
9084
9085 ;*****
9086 .SBTTL REPORT SUPPORT ROUTINE
9087 ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
9088 ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
9089 ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
9090 REPSUP:
9091 042130 104406 SAVREG
9092 042132 005037 005500 CLR ERRCOM ;CLEAR ERROR COMMAND STORE
9093 042136 116537 000001 005500 MOV P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
9094 042144 012700 001162 MOV #REG0,R0 ;FOR REPORTING
9095 042150 016520 000002 MOV P.CYLN(R5),(R0)+
9096 042154 116520 000005 MOV P.TRCK(R5),(R0)+
9097 042162 105020 000004 CLRB (R0)+
9098 042166 105020 CLRB P.SECT(R5),(R0)+
9099 042170 016520 000012 MOV (R0)+
9100 042174 012700 001174 MOV #REG5,R0
9101 042200 116503 000007 MOV P.BAHI(R5),R3
9102 042204 042703 177774 BIC #177774,R3
9103 042210 010337 001256 MOV R3,$REG36 ;HI BA BITS
9104 042214 016537 000010 001260 MOV P.BALO(R5),$REG37 ;LO BA BITS
9105 042222 016520 000016 MOV P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
9106 042226 016520 000020 MOV P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
9107 042232 016520 000030 MOV P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
9108 042236 016520 000026 MOV P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
9109 042242 016520 000022 MOV P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
9110 ;FOR ALL REPORTS (TO BE
9111 042246 016520 000024 MOV P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
9112 042252 016520 000032 MOV P.ASOF(R5),(R0)+ ;STORED ANY WAY.
9113 042256 016520 000036 MOV P.DS(R5),(R0)+
9114 042262 016520 000034 MOV P.ER(R5),(R0)+

```

```

9115 042266 016520 000040
9116 042272 016520 000042
9117 042276 016520 000044
9118 042302 016520 000046
9119 042306 016520 000050
9120 042312 016520 000052
9121 042316 016520 000054
9122 042322 016520 000056
9123
9124 042326 012701 001236
9125 042332 012700 005224
9126 042336 112021
9127 042340 105021
9128 042342 112021
9129 042344 105021
9130 042346 012021
9131 042350 116021 000001
9132 042354 105021
9133 042356 111021
9134 042360 105021
9135 042362 016021 000006
9136 042366 116003 000003
9137 042372 042703 177774
9138 042376 010321
9139 042403 016011 000004
9140 042404 104407
9141 042406 000207
9142
9143
9144
9145
9146
9147
9148
9149
9150
9151
9152 042410 104406
9153 042412 152737 000377 003113
9154 042420 105237 003116
9155 042424 005037 005476
9156 042430 032737 000400 005476
9157 042436 001402
9158 042440 012705 002704
9159 042444 012737 044606 003036
9160 042452 012737 044624 003034
9161 042460 004737 042130
9162
9163
9164
9165
9166
9167
9168
9169
9170

```

```

MOV P.A00(R5),(R0)+
MOV P.B00(R5),(R0)+
MOV P.A01(R5),(R0)+
MOV P.B01(R5),(R0)+
MOV P.A10(R5),(R0)+
MOV P.B10(R5),(R0)+
MOV P.A11(R5),(R0)+
MOV P.B11(R5),(R0)+
;STORE PREVIOUS COMMAND FOR PRINTOUT
MOV #SREG26,R1 ;ADRS OF PRINT BUF AREA
MOV #PRVCMD,R0 ;ADRS OF PREV CMND STORAGE
MOVB (R0)+,(R1)+ ;DRIVE NO.
CLRB (R1)+
MOVB (R0)+,(R1)+ ;COMMAND
CLRB (R1)+
MOV (R0)+,(R1)+ ;CYL ADDRESS
MOVB 1(R0),(R1)+ ;TRACK
CLRB (R1)+
MOVB (R0),(R1)+ ;SECTOR
CLRB (R1)+
MOV 6(R0),(R1)+ ;WORD COUNT
MOVB 3(R0),R3 ;HI BA BITS
BIC #177774,R3
MOV R3,(R1)+
MOV 4(R0),(R1) ;LO BA BITS
RESREG
RTS PC
;*****
;SBTTL REPORT ERROR ROUTINE
;* ENTRY JSR PC,ERRHDL
;*RETURN RTS PC
;*
;THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
;IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
;BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
;*****
ERRHDL: SAVREG
BISB #377,DONE ;SET DONE FLAG
INCB ERRCNT ;INCREMENT ERROR COUNT
CLR RECODE ;CLEAR RECOVERY CODE WORD
ER2ENT: BIT #LEV2EP,RECODE ;TEST IF 2ND LEVEL ERROR
BEQ 52$ ;NO - SKIP PARAM BLOCK CHANGE
MOV #PARM1,R5 ;ELSE SET R5 TO PARAMETER BLOCK 1
52$: MOV #RETANL,A.ABNL ;SET NOW ABNORMAL AND NORMAL RETURN FOR
MOV #RETNML,A.NORM ;DRIVER OPERATIONS IN ERROR PROCESSING
JSR PC,REPSUP ;GO SET UP REGISTERS FOR REPORT
;NOW BEGIN TESTING THE ERROR
;BITS. THE SEQUENCE IN
;WHICH THEY ARE TESTED IS
;CONSIDERED SIGNIFICANT IN
;THAT ERRORS OF A MORE
;CATASTROPHIC NATURE ARE FIRST
;TESTED.
;
;IF AN ERROR IS FOUND SET,

```

```

; THAT ERROR IS REPORTED AND
; THE REPORTING IS TERMINATED.
; IF ADDITIONAL ERRORS ARE SET,
; THE RK611 REGISTER PRINTOUTS
; WILL SHOW THIS BUT THE
; REGISTER CONTENTS MUST BE
; MANUALLY DECODED TO LOCATE THE
; SECOND ERROR
; SET R4 TO ERROR REGISTER
; TEST LPE. IF UES-SET
; REPORT ERROR

```

9200	042736	000034	020000	000020	MOV	P,ER(R5),R4	
9201	042737	000000			BIT	#LPE,P.CS2(R5)	
9202	104001				BEQ	15	
9203	052737	000200	005476		ERROR	!	
9204	050137	044160			BIS	#ABORT,RECODE	
9205	032765	004000	000020	15:	JMP	37\$	
9206	001406				BIT	#NEM,P.CS2(R5)	;TEST NON-EXISTANT MEMORY
9207	104002				BEQ	25	
9208	052737	000200	005476		ERROR	2	
9209	000137	044160			BIS	#ABORT,RECODE	
9210	032765	010000	000020	25:	JMP	37\$	
9211	001412				BIT	#NED,P.CS2(R5)	;TEST NON-EXISTANT DRIVE
9212	032765	000400	000020		BEQ	35	
9213	001403				BIT	#LPE,P.CS2(R5)	;TEST IF NED & LPE BOTH SET
9214	104035				BEQ	38\$	
9215	000137	044160			ERROR	35	;NED & LPE BOTH SET ERROR
9216	104003			38\$:	JMP	37\$	
9217	000137	044160			ERROR	3	;NED ONLY
9218	032765	000400	000020	35:	JMP	37\$	
9219	001412				BIT	#LPE,P.CS2(R5)	;TEST UNIT FIELD ERROR
9220	032765	010000	000020		BEQ	45	
9221	001403				BIT	#NED,P.CS2(R5)	;TEST IF LPE & NED BOTH SET
9222	104035				BEQ	39\$;NO-SKIP
9223	000137	044160			ERROR	35	;REPORT NED & LPE BOTH SET
9224	104004			39\$:	JMP	37\$	
9225	000137	044160			ERROR	4	;REPORT LPE ONLY
9226	032765	000100	000014	45:	JMP	37\$	
9227	001423				BIT	#CMDTO,P.PRST(R5)	
9228	004737	044634			BEQ	55	
9229	032737	002000	005476		JSR	PC, TOPROC	;GO PROCESS TIMEOUT
9230	001007				BIT	#TWOTOS,RECODE	;2ND TIMEOUT IN TIMEOUT PROC?
9231	032737	000400	005476		BNE	40\$;YES - SKIP TO ERROR 56
9232	001006				BIT	#LEV2ER,RECODE	;TEST IF LEVEL 2 ERROR
9233	104005				BNE	41\$;YES - SKIP TO ERROR 57
9234	000137	044160			ERROR	5	;ELSE MAKE FULL TIMEOUT REPORT
9235	104056			40\$:	JMP	37\$	
9236	000137	044160			ERROR	56	;TWO TIMEOUTS ERROR REPORT
9237	104057			41\$:	JMP	37\$	
9238	000137	042430			ERROR	57	;2ND LEVEL ERROR REPORT
9239	032765	010000	000014	55:	JMP	ERZENT	;GO BUILD AND MAKE 2ND REPORT
9240	001431				BIT	#DRVSZD,P.PRST(R5)	;SEE IF DRIVE SIEZED BY OTHER PORT
9241	105737	003126			BEQ	65\$;BR IF DRIVE IS AVAILABLE
9242	001003				TSTB	DULACS	;SEE IF DUAL-ACCESS FLAG SET
9243	104107				BNE	63\$;BR IF DUAL-ACCESS TEST
9244	000137	044160			ERROR	107	;DRIVE SIEZED BY OTHER PORT
9245	105237	003127		63\$:	JMP	37\$	
9246	012762	100000	000000		INCB	DRNAFG	;SET DRIVE NOT AVAILABLE FLAG
					MOV	#100000,RKCS1(R2)	;CLEAR CONTROLLER ERROR

9227	043274	003056	003104	MOV	W.MIN,W.DRV	:SET TIMER FOR ABOUT A MINUTE
9228	043275	005502		MOV	DRIVE,R0	:DRIVE NUMBER
9229	043276	116037	003071	MOV	I.DRV(R0),INTMSK	:SET MASK FOR THIS DRIVE
9230	043277	113737	003070	MOV	INTMSK,W.TIME	:SET DRIVE NUMBER FOR THIS DRIVE
9231	043278	105037	003113	CLAB	DONE	:CLEAR THE DONE FLAG
9232	043279	00207		RTS	PC	:GO WAIT FOR DRIVE ATT'N
9233	043280	032765	020000	BIT	#SPAR,P.CS1(R5)	:TEST SERCON PARITY ERROR
9234	043281	001406		BEQ	65	
9235	043282	104006		ERROR	6	
9236	043283	052737	004000	BIS	#RCLREQ,RECODE	
9237	043284	000137	044160	JMP	375	
9238	043285	032704	000010	BIT	#DRPAR,R4	:TEST DRIVE DETECTED PARITY ERROR
9239	043286	001406		BEQ	75	
9240	043287	104007		ERROR	7	
9241	043288	052737	004000	BIS	#RCLREQ,RECODE	
9242	043289	000137	044160	JMP	375	
9243	043290	032765	000010	BIT	#ACLO,P.DS(R5)	:TEST AC LOW
9244	043291	001403		BEQ	85	
9245	043292	104010		ERROR	10	
9246	043293	000137	044160	JMP	375	
9247	043294	032765	000020	BIT	#SPDLSS,P.DS(R5)	:TEST SPEED LOSS
9248	043295	001403		BEQ	245	
9249	043296	104011		ERROR	11	
9250	043297	000137	044160	JMP	375	
9251	043298	032765	004000	BIT	#CTO,P.CS1(R5)	:TEST FOR CONTROLLER TIMEOUT
9252	043299	001401		BEQ	255	
9253	043300	104027		ERROR	27	
9254	043301	032704	000001	BIT	#ILC,R4	:TEST ILLEGAL FUNCTION CODE
9255	043302	001403		BEQ	105	
9256	043303	104012		ERROR	12	
9257	043304	000137	044160	JMP	375	
9258	043305	032755	002000	BIT	#PGE,P.CS2(R5)	:TEST PROGRAMMING ERROR
9259	043306	001403		BEQ	115	
9260	043307	104013		ERROR	13	
9261	043308	000137	044160	JMP	375	
9262	043309	032704	000004	BIT	#ILF,R4	:TEST ILLEGAL DRIVE FUNCTION
9263	043310	001403		BEQ	125	
9264	043311	104014		ERROR	14	
9265	043312	000137	044160	JMP	375	
9266	043313	032704	000040	BIT	#DTYE,R4	:TEST DRIVE TYPE ERROR
9267	043314	001403		BEQ	135	
9268	043315	104015		ERROR	15	
9269	043316	000137	044160	JMP	375	
9270	043317	032704	000020	BIT	#FMTE,R4	:TEST FORMAT ERROR
9271	043318	001403		BEQ	145	
9272	043319	104016		ERROR	16	
9273	043320	000137	044160	JMP	375	
9274	043321	032704	004000	BIT	#WLE,R4	:TEST WRITE LOCK ERROR
9275	043322	001403		BEQ	155	
9276	043323	104017		ERROR	17	
9277	043324	000137	044160	JMP	375	
9278	043325	032704	040000	BIT	#UNS,R4	:TEST DRIVE UNSAFE
9279	043326	001406		BEQ	165	
9280	043327	104020		ERROR	20	
9281	043328	052737	000200	BIS	#ABORT,RECODE	
9282	043329	000137	044260	JMP	ALLTRM	

9283	043250	032704	000002		16:	BIT	#SKI,R4		;TEST SEEK INCOMPLETE
9284	043254	001406				BEQ	17		
9285	043256	104021				ERROR	21		
9286	043260	052737	004000	005476		BIS	#RCLREQ,RECODE		
9287	043266	000137	044160			JMP	37		
9288	043272	032704	001000		17:	BIT	#COE,R4		;TEST CYLINDER OVERFLOW
9289	043276	001406				BEQ	18		
9290	043300	104022				ERROR	22		
9291	043302	052737	004000	005476		BIS	#RCLREQ,RECODE		
9292	043310	000137	044160			JMP	37		
9293	043314	032704	002000		18:	BIT	#IDAE,R4		;TEST ILLEGAL CYLINDER
9294	043320	001406				BEQ	19		
9295	043322	104023				ERROR	23		
9296	043324	052737	004000	005476		BIS	#RCLREQ,RECODE		
9297	043332	000137	044160			JMP	37		
9298	043336	032765	000040	000036	19:	BIT	#DROT,P.DS(R5)		;TEST DRIVE OFF TRACK
9299	043344	001406				BEQ	20		
9300	043346	104024				ERROR	24		
9301	043350	052737	004000	005476		BIS	#RCLREQ,RECODE		
9302	043356	000137	044160			JMP	37		
9303	043362	032704	010000		20:	BIT	#DTE,R4		;TEST DRIVE TIMING ERROR
9304	043366	001406				BEQ	21		
9305	043370	104025				ERROR	25		
9306	043372	052737	000200	005476		BIS	#ABORT,RECODE		
9307	043400	000137	044160			JMP	37		
9308	043404	032765	100000	000020	21:	BIT	#DLT,P.US2(R5)		;TEST DATA LATE
9309	043412	001403				BEQ	22		
9310	043414	104026				ERROR	26		
9311	043416	000137	044160			JMP	37		
9312	043422	032704	020000		22:	BIT	#OPI,R4		;TEST IF OPI ERROR
9313	043426	001457				BEQ	29		
9314	043430	052737	000010	005476		BIS	#OPIERR,RECODE		
9315	043436	105737	003111			TSTB	DERCNT		;TEST IF FIRST ERROR
9316	043442	001402				BEQ	50		
9317	043444	000137	044160			JMP	37		;NO - SKIP REPORT
9318	043450	032737	000400	005476	50:	BIT	#LEV2ER,RECODE		;TEST IF A SECOND LEVEL 2 ERROR
9319	043456	001403				BEQ	26		;HAS ALREADY OCCURRED
9320	043460	104054			27:	ERROR	54		
9321	043462	000137	044160			JMP	37		;GET OUT OF ERROR REPORT
9322	043466	004737	045264		26:	JSR	PC,BLDEXH		;GO BUILD EXPECTED HEADER
9323	043472	004737	045004			JSR	PC,RDH00		;GET HEADER OF SECTOR 0
9324	043476	032737	000400	005476		BIT	#LEV2EF,RECODE		;TEST IF ERROR GETTING HDR
9325	043504	001025				BNE	28		;IF YES-GO MAKE ABBREVIATED REPORT
9326	043506	013702	003026			MOV	RKBAS,R2		;STORE HEADER 0 INTO REGISTERS
9327	043512	042762	000100	000000		BIC	#IE,RKCS1(R2)		;RESET INTERRUPT ENABLE
9328	043520	016237	000024	001202		MOV	RKDB(R2),SREG10		;FOR REPORTING
9329	043526	016237	000024	001204		MOV	RKDB(R2),SREG11		
9330	043534	016237	000024	001206		MOV	RKDB(R2),SREG12		
9331	043542	032762	100000	000000		BIT	#CERR,RKCS1(R2)		;TEST IF ERROR DURING STORAGE
9332	043550	001343				BNE	27		
9333	043552	104030				ERROR	30		;MAKE OPI REPORT
9334	043554	000137	044160			JMP	37		
9335	043560	104054			28:	ERROR	54		
9336	043562	000137	042430			JMP	ER2ENT		;GO MAKE 2ND LEVEL REPORT
9337	043566	032704	000400		29:	BIT	#HVRC,R4		;TEST IF HVRC ERROR
9338	043572	001457				BEQ	23		

9339	043574	052737	000004	005476	BIS	#HVRER,RECODE	
9340	043602	105737	003111		TSTB	DEPCNT	;TEST IF FIRST ERROR
9341	043606	001402			BEQ	30\$;YES - REPORT
9342	043610	000137	044160		JMP	37\$;JUMP TO RETURN
9343	043614	032737	000400	005476	30\$: BIT	#LEVZER,RECODE	;TEST IF A 2ND LEVEL ERROR HAS ALPEADY
9344	043622	001403			BEQ	31\$;OCCURRED. NO-SKIP EXIT
9345	043624	104055			51\$: ERROR	55	
9346	043626	000137	044160		JMP	37\$;GET OUT OF ERROR REPORT
9347	043632	004737	045264		31\$: JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
9348	043636	004737	045004		JSR	PC,RDHD	;GO GET HDR 0
9349	043642	032737	000400	005476	BIT	#LEVZER,RECODE	;TEST IF ERROR IN GETTING HDR
9350	043650	001025			32\$: SNE	32\$;IF YES-GO MAKE ABBREVIATED REPORT
9351	043652	013702	003026		MOV	RKBAS,R2	;SET PK611 BASE ADDRESS
9352	043656	042762	000100	000000	BIC	#IE,RKCS1(R2)	;CLEAR INTERRUPT ENABLE
9353	043664	016237	000024	001202	MOV	RKDB(R2),SREG10	;STORE OFF HEADER
9354	043672	016237	000024	001204	MOV	RKDB(R2),SREG11	
9355	043700	016237	000024	001206	MOV	RKDB(R2),SREG12	
9356	043706	032762	100000	000000	BIT	#CERR,RKCS1(R2)	;TEST IN ANY ERROR IN UNLOAD
9357	043714	001343			51\$: BNE	51\$;IF YES - GO MAKE SHORT REPORT
9358	043716	104031			ERROR	31	;MAKE FULL REPORT
9359	043720	000137	044160		JMP	37\$	
9360	043724	104055			32\$: ERROR	55	;ABBREVIATED HVRC ERROR RPORT
9361	043726	000137	042430		JMP	ERRCNT	;GO REPORT 2ND LEVEL ERROR
9362	043732	032704	000200		23\$: BIT	#BSE,R4	;TEST FOR BAD SECTOR ERROR
9363	043736	001430			BEQ	33\$;NO - SKIP
9364	043740	052737	000002	005476	BIS	#BSERR,RECODE	;SET ERROR FLAG
9365	043746	016500	000030		MOV	P.DCYL(R5),R0	;GET CYL NO.
9366	043752	116501	000027		MOVB	P.DTS+1(R5),R1	;GET TRACK NO.
9367	043756	042701	177774		BIC	#177774,R1	;CLEAR ALL BITS EXCEPT TRACK
9368	043762	016502	000026		MOV	P.DTS(R5),R2	;GET SECTOR IN ERROR
9369	043766	042702	177740		BIC	#177740,R2	;CLEAR ALL BITS EXCEPT SECTOR
9370	043772	004737	040612		JSR	PC,BDSACK	;GO SEE IF THIS SECTOR LISTED
9371	043776	032737	001000	005476	BIT	#BADSEC,RECODE	;TEST RESULT
9372	044004	001065			37\$: BNE	37\$;YES - EXIT, NO ERROR OR REPORT
9373	044006	104104			ERROR	104	;ELSE REPORT BAD BSE
9374	044010	042737	000002	005476	BIC	#BSERR,RECODE	;RESET BSE ERROR FLAG
9375	044016	000450			BR	37\$;GO EXIT
9376	044020	032704	100000		33\$: BIT	#DCK,R4	;TEST IF DATA CHECK
9377	044024	001435			BEQ	36\$	
9378	044026	052737	000020	005476	BIS	#DCKERR,RECODE	;SET DATA CHECK ERROR IN RECOVERY CODE
9379	044034	032704	000100		BIT	#ECH,R4	;TEST IF ECC IS HARD. IF
9380	044040	001406			BEQ	34\$;YES SET UNCORRECTABLE IN
9381	044042	052737	000040	005476	BIS	#ECCNC,RECODE	;RECOVERY FLAG AND A 0 IN
9382	044050	005037	001206		CLR	SREG12	;REG12 TO INDICATE UNCORRECTABLE,
9383	044054	000403			BR	35\$;A 1 IN REG 12 FOR CORRECTABLE
9384	044056	012737	000001	001206	34\$: MOV	#1,SREG12	
9385	044064	105737	003111		35\$: TSTB	DEPCNT	;TEST IF FIRST ERROR
9386	044070	001033			BNE	37\$;NO SKIP REPORT
9387	044072	004737	045140		JSR	PC,GTPKAD	;GO GET PACK ADDRESS OF ERROR
9388	044076	016537	000060	001202	MOV	P.EPOS(R5),SREG10	;STORE ECC POSITION &
9389	044104	016537	000062	001204	MOV	P.EPAT(R5),SREG11	;PATTERN
9390	044112	104032			ERROR	32	;REPORT DCK ERROR
9391	044114	000137	044160		JMP	37\$	
9392	044120	032765	040000	000020	36\$: BIT	#WCE,P.CS2(R5)	;TEST WRITE CHECK ERROR
9393	044126	001414			BEQ	37\$	
9394	044130	042737	000200	005476	BIC	#ABORT,RECODE	;CLEAR ABORT & SET WRITE

```

9395 044136 052737 000100 005476      BIS      #WCERR,RECODE      ;CHECK ERROR IN RECODE
9396 044144 105737 003111                    TSTB     DFRONT      ;TEST IF FIRST ERROR
9397 044150 001003                    BNE      37$        ;NO - SKIP
9398 044152 004737 045140      JSR      PC,GTPKAD   ;GO GET ADDRESS OF ERROR
9399 044156 104033      ERROR    33        ;REPORT WCE
9400
9401 044160 032765 000020 000014 37$:    BIT      #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
9402 044166 001404                    BEQ      43$
9403 044170 104036      ERROR    36
9404 044172 052737 000200 005476      BIS      #ABORT,RECODE
9405
9406 044200 032765 000040 000014 43$:    BIT      #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
9407 044206 001404                    BEQ      44$
9408 044210 104037      ERROR    37
9409 044212 052737 000200 005476      BIS      #ABORT,RECODE
9410
9411 044220 032765 004000 000014 44$:    BIT      #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
9412 044226 001404                    BEQ      46$
9413 044230 104040      ERROR    40
9414 044232 052737 000200 005476      BIS      #ABORT,RECODE
9415
9416 044240 032765 000010 000014 46$:    BIT      #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
9417 044246 001404                    BEQ      ALLTRM
9418 044250 104042      ERROR    42
9419 044252 052737 000200 005476      BIS      #ABORT,RECODE
9420
9421
9422
9423
9424 044260 012705 002620                    ALLTRM: MOV      #PARMO,R5      ;RESTORE PARAMETER BLOCK SELECTION
9425 044264 012737 042410 003036      MOV      #ERRHDL,A.ABNL ;RESTORE INTERRUPT VECTORS FOR RETRY
9426 044272 012737 041142 003034      MOV      #ERRFRE,A.NORM ;DRIVER OPERATIONS, IF ANY
9427 044300 032737 000200 005476      BIT      #ABORT,RECODE  ;IF ABORT IS NOT SET AND
9428 044306 001043                    BNE      48$        ;THE DRIVE IS READY (HAS NOT
9429
9430 044310 013702 003026                    MOV      RKBAS,R2    ;GET BASE ADDRESS
9431 044314 032762 000200 000012      BIT      #RDY,RKDS(R2) ;TEST IF DRIVE READY SET
9432 044322 001004                    BNE      47$        ;RECALIBRATE REQUIRED BIT IS SET
9433 044324 052737 000200 005476      BIS      #ABORT,RECODE ;ELSE ABORT WITH ABORT MESSAGE
9434 044332 000431                    BR       48$
9435 044334 032737 004000 005476 47$:    BIT      #RCLREQ,RECODE ;IF RECALIBRATE IS REQUIRED
9436 044342 001434                    BEQ      BGNRTY     ;FOR RETRY SET UP PARAM
9437 044344 112737 000113 000001      MOVB     #RECAL,P.CMND ;BLOCK TO DO IT.
9438 044352 012737 044606 003036      MOV      #RETANL,A.ABNL
9439 044360 004737 040716      JSR      PC,DRVCAL
9440 044364 012737 042410 003036      MOV      #ERRHDL,A.ABNL ;RESTORE ERROR RETURN
9441 044372 032737 000400 005476      BIT      #LEV2ER,RECODE ;IF AN ERROR OCCURRED IN THE
9442 044400 001415                    BEQ      BGNRTY     ;RECAL ATTEMP SET ABORT
9443 044402 052737 000200 005476      BIS      #ABORT,RECODE ;PRINT THE RECAL ERROR MESSAGE, AND
9444 044410 104060      ERROR    60        ;GO REPORT DETAILS
9445 044412 000137 042430                    JMP      ERZENT
9446 044416 104061      ERROR    61        ;REPORT ABORT-RETRY FAILED
9447
9448
9449
9450

```

;ALL ERRORS MUST EXIT THROUGH THIS POINT

;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
;IF THE DRIVE READY BIT IS RESET.

```

9451 044420 000000          HLTPRG: HHLT
9452 044422 105037 003116          CLR  ERRCNT          ;CLEAR ERROR COUNT
9453 044426 000005          RESET                ;RESET ALL DEVICES
9454 044430 000137 013516          JMP   CMSTRT
9455
9456
9457 ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
9458 ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
9459 ;THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
9460 ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
9461 ;FLAG IS SET AND PROGRAM HALTS.
9462
9463 044434 032737 000136 005476 BGNRTY: SIT          #BSERR!HVR CER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
9464 044442 001404          BEQ   3$
9465 044444 052737 100000 005476 9$: BIS          #ANYDER,RECODE
9466 044452 000453          BR   2$
9467 044454          3$:
9468 044454 105737 003133          TSTB NORTRY          ;SEE IF "NO-RETRY" FLAG SET
9469 044460 001371          BNE  9$              ;BR IF YES
9470 044462 032737 001000 005476 BIT          #BADSEC,RECODE ;TEST IF BAD SECTOR FLAG SET
9471 044470 001044          BNE  2$              ;IF YES-EXIT TO CALLER
9472 044472 123737 003116 003117 CMPB  ERRCNT,ERRLMT ;BEGIN RETRY IF ERROR COUNT HAS
9473 044500 001012          BNE  1$              ;NOT BEEN EXCEEDED
9474 044502 005037 001174          CLR   $REGS
9475 044506 113737 003116 001174 MOVB  ERRCNT,$REGS ;GET ERROR RETRY COUNT
9476 044514 104102          ERROR 102           ;REPORT RETRY UNSUCCESSFUL
9477 044516 052737 000200 005476 BIS   #ABORT,RECODE ;SET ABORT & QUIT
9478 044524 000735          BR   HLTPRG
9479 044526 013702 003026          1$: MOV   RKBAS,R2 ;GET RK BASE ADDRESS
9480 044532 112765 000177 000001 MOVB  #SUBCLR,P.CMND(R5) ;SET UP TO CLEAR SUBSYSTEM
9481 044540 004737 040716          JSR  PC,DRVCAL ;GO DO IT
9482 044544 012700 005240          MOV  #COMSTR,R0 ;GO AND REESTABLISH THE COMMAND
9483 044550 012025          MOV  (R0)+,(R5)+ ;AS IT WAS ENTERED INTO THE
9484 044552 012025          MOV  (R0)+,(R5)+ ;PARAMETER BLOCK
9485 044554 012025          MOV  (R0)+,(R5)+
9486 044556 012025          MOV  (R0)+,(R5)+
9487 044560 012025          MOV  (R0)+,(R5)+
9488 044562 012025          MOV  (R0)+,(R5)+
9489 044564 012705 002620          MOV  #PARMO,R5
9490 044570 012737 000000 177776 MOV  #PRO_2#PSW ;LOWER PRIORITY TO ALLOW INTERRUPT
9491 044576 004737 040716          JSR  PC,DRVCAL ;CALL DRIVER
9492 044602 104407          2$: RESREG ;IF RETURN GETS HERE, NO ERROR
9493 044604 000207          RTS   PC ;OCCURRED, RECOVERY WAS SUCCESSFUL
9494
9495
9496 044606 152737 000377 003113 RETANL: BISB  #377,DONE ;SET DONE
9497 044614 052737 000400 005476 BIS   #LEV2ER,RECODE ;SET LEVEL TWO ERROR
9498 044622 000207          RTS   PC
9499 044624 152737 000377 003113 RETNML: BISB  #377,DONE ;SET DONE
9500 044632 000207          RTS   PC
9501
9502
9503 ;*****
9504 ;SBTTL TIME OUT PROCESSOR ROUTINE
9505 ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
9506 ;*GATHERING DUTIES.

```

```

9507
9508 044634
9509 044634 104406
9510 044636 013702 003026
9511 044642 012701 001174
9512 044646 016221 000000
9513 044652 016221 000010
9514 044656 016221 000020
9515 044662 016221 000006
9516 044666 016221 000002
9517 044672 016221 000004
9518 044676 016221 000016
9519 044702 016221 000012
9520 044706 016221 000014
9521 044712 005000
9522
9523
9524 044714 012705 002704
9525 044720 112765 000141 000001
9526 044726 004737 040716
9527 044732 032765 000100 000014
9528 044740 001403
9529 044742 052737 002000 005476
9530 044750 062705 000040 1S:
9531 044754 012521
9532 044756 012521
9533 044760 012521
9534 044762 012521
9535 044764 012521
9536 044766 012521
9537 044770 012521
9538 044772 012521
9539
9540 044774 012705 002620
9541 045000 104407
9542 045002 000207
9543
9544
9545
9546
9547
9548
9549 045004
9550 045004 104406
9551 045006 016501 000026
9552 045012 016500 000052
9553 045016 042700 160017
9554 045022 006200
9555 045024 006200
9556 045026 006200
9557 045030 006200
9558 045032 012705 002704
9559 045036 010165 000004
9560 045042 010065 000002
9561 045046 132737 000020 003115
9562 045054 001404

```

```

*****
↑CPROC:
SAVREG
MOV RKBAS,R2
MOV #SREGS,R1 ;SET UP R1 FOR RK REGISTER STORAGE
MOV RKCS1(R2),(R1)+ ;STORE ALL VALID RKS11
MOV RKCS2(R2),(R1)+ ;REGISTERS
MOV RKDC(R2),(R1)+
MOV RKDA(R2),(R1)+
MOV RKWC(R2),(R1)+
MOV RKBA(R2),(R1)+
MOV RKASOF(R2),(R1)+
MOV RKDS(R2),(R1)+
MOV RKER(R2),(R1)+
CLR RO ;THIS CODE WILL ATTEMPT TO
;RETRIEVE THE STATUS FROM THE
;DRIVE.
MOV #PARM1,R5 ;SET UP TO USE PARM1
MOV #ROSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
JSR PC,DRVCAL ;CALL DRIVER
BIT #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
BEQ 1S ;NO - SKIP
BIS #TWOLOS,RECODE ;SET TWO TIMEOUTS FLAG
ADD #P.ADD,R5 ;BUMP R5 TO POINT TO DRIVE STATUS
MOV (R5)+,(R1)+ ;MOVE ALL THE DRIVE STATUS INTO THE
MOV (R5)+,(R1)+ ;TEMP REGS FOR REPORTING.
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV (R5)+,(R1)+
MOV #PARMO,R5 ;RESTORE PARM 0
RESREG
RTS PC

```

```

*****
.SBTTL READ HEADER 0 ROUTINE
*****
RDHDO:
SAVREG
MOV P.DTS(R5),R1 ;STORE TRACK AND SECTOR
MOV P.B10(R5),RO ;GET THE CYLINDER ADRS
BIC #160017,RO ;FROM THE DRIVE STATUS. CLEAR
ASR RO ;OFF UNUSED BITS AND POSITION
ASR RO ;FOR USE AS THE DESIRED
ASR RO ;CYLINDER IN THE READ
ASR RO ;HEADER COMMAND.
MOV #PARM1,R5 ;SET UP TO USE P.B. 1
MOV R1,P.SECT(R5) ;INSERT TRK,SECT FOR READ HDR
MOV RO,P.CYLN(R5) ;SET CYL NO.
BITB #B.CFMT,FORMAT ;TEST PRESENT FMT AND CHANGE IT
BEQ 1S ;TO THE OPPOSITE. THIS WILL CAUSE

```


9619	045312	016537	000026	001176		MOV	P.DTS(R5), \$REG6	:BIT.
9620	045320	042737	177740	001176		BIC	#177740, \$REG6	:CLEAR ALL BUT SECTOR
9621	045326	060137	001176			ADD	R1, \$REG6	:ADD TRACK AND SECTOR TOGETHER
9622	045332	052737	140000	001176		BIS	#140000, \$REG6	:INSERT BSE BITS
9623	045340	032765	010000	000016		BIT	#CFMT, P.CS1(R5)	
9624	045346	001403				BEQ	23\$	
9625	045350	052737	001000	001176		BIS	#1000, \$REG6	
9626	045356	013737	001174	001200	23\$:	MOV	\$REG5, \$REG7	; COMPUTE THE HEADER VRC
9627	045364	013701	001176			MOV	\$REG6, R1	
9628	045370	043737	001176	001200		BIC	\$REG6, \$REG7	
9629	045376	043701	001174			BIC	\$REG5, R1	
9630	045402	050137	001200			BIS	R1, \$REG7	
9631	045406	104407				RESREG		
9632	045410	000207				RTS	PC	
9633								

9634
9635
9636
9637
9638
9639
9540
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
9675
9676
9677
9678
9679
9680
9681
9682
9683
9684
9685
9686
9687
9688
9689

.SBTTL RK611/RK06 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.08)

;*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 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 SIMULTANEOUSLY. 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 Z0\$;IF NOT ZERO RETURN
MOV W.MILI,W.MTIM ;REINITIALIZE MILLISECOND TIMER
TSTB W.TIME ;CHECK IF DRIVE IS BEING TIMED
BEQ Z0\$;NO. RETURN
MOV RKPRI,PS ;LOCK OUT RK06 INTERRUPTS
MOV RKBAS,R2 ;LOAD BASE OF RK06 REGISTERS
DEC W.DRV ;DECREMENT COMMAND TIMER
BNE Z0\$;RETURN IF NO TIME OUT

045412 010546
045414 010446
045416 010346
045420 010246
045422 013746 177776
045426 005337 003046
045432 001034
045434 013737 003050 003046
045442 105737 003070
045446 001426
045450 013737 003032 177776
045456 013702 003026
045462 005337 003104
045466 001016

9690	045470	105037	003070		CLRB	W.TIME	:RESET TIMING INDICATOR
9691	045474	013705	003102		MOV	PBLKT,R5	:LOAD ADDRESS OF PARAMETER BLOCK
9692							:TABLE FOR INDEXING
9693	045500	052765	000100	000014	BIS	*CMDTO,P.PRST(R5)	:SET COMMAND TIME OUT
9694	045506	020537	003044		CMP	PS,O.WAIT	:CHECK IF DRIVER IS WAITING FOR
9695							:COMMAND COMPLETION
9696	045512	001002			BNE	5\$:NO, DO NOT ALTER WAITING FOR
9697							:COMMAND COMPLETION
9698	045514	005037	003044		CLR	O.WAIT	:CLEAR WAIT FOR COMMAND COMPLETION
9699	045520	004737	050774	5\$:	JSR	PC,R.ABNL	:BRANCH TO ERROR ROUTINE
9700	045524	012637	177776	20\$:	MOV	(SP)+,PS	:RESTORE PSW
9701	045530	012602			MOV	(SP)+,R2	:RESTORE R2
9702	045532	012603			MOV	(SP)+,R3	:RESTORE R3
9703	045534	012604			MOV	(SP)+,R4	:RESTORE R4
9704	045536	012605			MOV	(SP)+,R5	:RESTORE R5
9705	045540	000207			RTS	PC	:RETURN

9706
9707
9708
9709
9710
9711
9712
9713
9714
9715
9716
9717
9718
9719
9720
9721
9722
9723
9724
9725
9726
9727
9728
9729
9730
9731
9732
9733
9734
9735
9736
9737
9738
9739
9740
9741
9742
9743
9744
9745
9746
9747
9748
9749
9750
9751
9752
9753
9754
9755
9756
9757
9758
9759
9760
9761

.SBTTL *RK06 INTERRUPT SERVICE ROUTINE

THIS ROUTINE WILL SERVICE ALL RK06 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

```

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,R.CONT ;REPORT ERROR

```

```

045542 010546
045544 010446
045546 010346
045550 010246
045552 010146
045554 010046
045556 013702 003026
045562 016237 000010 002772
045570 032737 001000 002772
045576 001407
045600 052737 100000 003042
045606 004737 051020

```

B01

```

01775 045612 000137 047772          JMP      I.RTRN          ;RETURN
01776 045616 105737 003064          1$:    TSTB      I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
01777 045622 001410          BEQ      6$              ;NO, CHECK IF DRIVE AVAILABLE
01778 045624 100433          BMI      5$              ;CHECK IF RELEASE COMMAND
01779 045626 105037 003064          CLRB     I.ISRL          ;YES, CLEAR FLAG
01780 045632 000473          SR       I.IOC           ;CONTINUE PROCESSING INTERRUPT
01781 045634 105037 003064          5$:    CLRB     I.ISRL          ;CLEAR FLAG
01782 045640 000137 046754          JMP      I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
01783 045644 032737 010400 002772 6$:    BIT       #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
01784 045652 001413          BEQ      7$              ;UNIT FIELD ERROR
01785 045654 013704 002772          MOV      T.CS2,R4        ;NO, WAIT FOR DUAL ACCESS INTERRUPT
01786 045660 042704 177773          BIC      #IC(DRVMSK),R4 ;LOAD R4 FOR DRIVE NUMBER
01787 045664 013705 003102          MOV      PBLKT,R5        ;KEEP DRIVE BITS
01788 045670 016237 000000 002770  MOV      RKCS1(R2),T.CS1 ;STORE PARAMETER BLOCK ADDRESS
01789 045676 000137 046164          JMP      I.ERRC          ;LOAD TEMPORARY CSI FOR STATUS REPORT
01790 045702 016237 000012 003010 7$:    MOV      RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
01791 045710 032737 000001 003010  BIT      #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
01792 045716 001041          BNE      I.IOC           ;PORT
01793 045720 032737 164000 002772  ;NO, CONTINUE PROCESSING INTERRUPT
01794 045726 001007          ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
01795 045730 016237 000014 003006  BIT      #DLT!WCE!UPE!NEM,T.CS2
01796 045736 032737 125700 003006  BNE      10$             ;INDICATE ERROR
01797 045744 001407          MOV      RKER(R2),T.ER   ;STORE ERROR REGISTER
01798 045746 052737 000010 003042  ;CHECK FOR DATA TRANSFER ERROR TYPE ERROR
01799 045754 004737 051020          BIT      #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
01800 045760 000137 047772          BEQ      11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
01801 045764 105037 003070          10$:   BIS      #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
01802 045770 005037 003104          JSR      PC.R.CONT       ;REPORT ERROR
01803 045774 013705 003102          JMP      I.RTRN          ;RESTORE REGISTERS
01804 046000 052765 010000 000014  BIS      #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
01805 046006 005037 003044          CLR      O.WAIT          ;PROGRAM DRIVE STATUS REGISTER
01806 046012 004737 050774          CLR      W.DRV           ;CLEAR WAIT FOR COMMAND COMPLETION
01807 046016 000137 047772          JSR      PC.R.ABNL       ;INDICATE ABNORMAL TERMINATION
01808 046022 013705 003044          JMP      I.RTRN          ;GO RESTORE REGISTERS
01809 046026 001002          I.IOC: MOV      O.WAIT,R5    ;LOAD PARAMETER BLOCK ADDRESS INTO R5
01810 046030 000137 046754          BNE      2$              ;IS COMMAND WAITING PROCESSING
01811 046034 013704 002772          JMP      I.ATTN          ;YES, DO PROCESSING
01812 046034 013704 002772          2$:    MOV      T.CS2,R4    ;NO, PROCESS ATTENTION
01813 046034 013704 002772          ;STORE RKCS2 FOR DRIVE NUMBER
  
```

CO1

DZR6N-B - RK611, RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB *RK06 INTERRUPT SERVICE ROUTINE

MACY11 27(732) 03-NOV-76 21:43 PAGE 210

```

9818 046040 042704 177770          BIC      #IC<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
9819
9820
9821 046044 126504 000000          CMPB    P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
9822 046050 001401          BEQ     3$ ;YES, CONTINUE
9823 046052 000000          HALT    ;NO, DRIVER ERROR
9824 046054 122765 000164 000001 3$:    CMPB    #RDALHD,P.COMND(R5) ;CHECK IF READ ALL HEADERS
9825 046062 001002          BNE     10$ ;NO, EXECUTE NORMAL DATA TRANSFER
9826 046064 000137 046422          JMP     I.HDAL ;GO EXECUTE SPECIAL HEADER SEQUENCE
9827
9828 046070 005037 003044          10$:   CLR     O.WAIT ;CLEAR WAIT FOR COMMAND COMPLETION
9829 046074 005037 003104          CLR     W.DRV ;CLEAR WATCH-DOG TIME
9830 046080 105037 003070          CLRB   W.TIME ;RESET TIMING ON THIS DRIVE
9831 046104 016237 000000 002770      MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
9832 046112 032737 100000 002770      BIT     #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
9833 046120 001021          BNE     I.ERRC ;YES, PROCESS ERROR
9834 046122 016237 000016 003004      MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9835 046130 133737 003071 003005      BITB   INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
9836 046136 001004          BNE     15$ ;YES, REPORT ERROR
9837 046140 004737 051006          JSR    PC,R.NORM ;INDICATE NORMAL RETURN
9838 046144 000137 047772          JMP     I.RTRN ;RESTORE REGISTERS
9839
9840 046150 052765 000010 000014 15$:   BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
9841
9842 046156 004737 050442          I.ERRA: JSR    PC,I.CSTS ;STORE CONTROLLER STATUS
9843 046162 000405          BR     I.ERR ;STORE PATTERN AND POSITION INFORMATION
9844
9845 046164 013765 002770 000016  I.ERRC: MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
9846 046172 004737 050464          JSR    PC,I.CST1 ;GET REST OF CONTROLLER STATUS
9847 046176 016265 000032 000062  I.ERR:  MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
9848 046204 016265 000030 000060      MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
9849 046212 004037 050010          JSR    RD,I.CCLR ;CLEAR CONTROLLER
9850 046216 047772          I.RTR: I.RTRN ;ERROR RETURN
9851 046220 032765 010400 000020      BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
9852                                     ; UNIT FIELD ERROR
9853 046225 001046          BNE     5$ ;YES, REPORT ERROR
9854 046230 004037 050546          JSR    RD,I.STAT ;GATHER DRIVE STATUS
9855 046234 047772          I.RTRN ;ERROR RETURN
9856 046236 112737 000005 002770      MOVB   #DR.CLR,T.CS1 ;LOAD COMMAND
9857 046244 004037 050072          JSR    RD,I.ISSU ;ISSUE DRIVE CLEAR
9858 046250 047772          I.RTRN ;ERROR RETURN
9859 046252 133737 003071 003005      BITB   INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
9860 046260 001407          BEQ     2$ ;NO, INDICATE DRIVE ERROR
9861 046262 052737 000020 003042      BIS     #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
9862                                     ; WITH CLEAR
9863 046270 004737 051020          JSR    PC,R.CONT ;REPORT CONTROLLER ERROR
9864 046274 000137 047772          JMP     I.RTRN ;GO RESTORE REGISTERS
9865
9866 046300 032737 040000 003014 2$:   BIT     #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE CLEARED
9867 046306 001403          BEQ     3$ ;YES, CHECK FAULT
9868 046310 052765 000040 000014      BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
9869 046316 032737 001000 003016 3$:   BIT     #S.PAR,T.MR3 ;CHECK IF DRIVE PARITY ERROR
9870 046324 001407          BEQ     5$ ;NO, INDICATE ABNORMAL TERMINATION
9871 046326 052737 002000 003042      BIS     #E.DPAR,E.CONT ;SET DRIVE PARITY ERROR
9872 046334 004737 051020          JSR    PC,R.CONT ;INDICATE CONTROLLER ERROR
9873 046340 000137 047772          JMP     I.RTRN ;RETURN

```

```

9874
9875 046344 032765 000020 000014 5$: BIT #DRVHRC,P.PAST(R5) ;CHECK IF HARD DRIVE ERROR
9876 046352 001017 BNE 10$ ;YES, GO REPORT ERROR
9877 046354 032737 020000 003014 3IT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
9878 046362 001413 BEQ 10$ ;NO, REPORT ERROR
9879 046364 052765 020000 000014 BIS #E.UNLD,P.PAST(R5) ;SET DRIVE UNLOADING
9880 046372 113737 003071 003070 MOV#B INT.MSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
9881 046400 013737 003054 003104 MOV W.BSEC,W.DRV
9882 046406 000137 047772 JMP I.RTR' ;GO RESTORE REGISTERS
9883
9884 046412 004737 050774 10$: JSR PC,R.ABNL ;GO REPORT ERROR
9885 045416 000137 047772 JMP I.RTRN ;GO RESTORE REGISTERS
9886
9887 .S9TTL *READ ALL HEADERS INTERRUPT SEQUENCE
9888
9889 046422 016237 000000 002770 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
9890 ; ERROR
9891 046430 032737 100000 002770 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
9892 046436 001422 BEQ 5$ ;NO, CHECK FOR ATTENTION
9893
9894 046440 005037 003044 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
9895 046444 105037 003070 CLR#B W.TIME ;RESET TIMING ON DRIVE
9896 046450 005037 003104 CLR W.DRV ;CLEAR TIME OUT COUNT
9897 046454 013765 002770 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
9898 045462 004737 050464 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
9899 046466 004037 050010 JSR RO,I.CCLR ;CLEAR CONTROLLER
9900 045472 047772 I.RTRN ;ERROR RETURN
9901 046474 004737 050774 JSR PC,R.ABNL ;INDICATE ERROR RETURN
9902 046500 000137 047772 JMP I.RTRN ;RESTORE REGISTERS
9903
9904 046504 016537 000016 003004 5$: MOV RKASOF(R5),T.ASOF ;STORE ATTENTION SUMMARY
9905 046512 133737 003071 003005 BIT#B INTMSK,T.ASCF+1 ;CHECK IF DRIVE ATTENTION IS SET
9906 046520 001410 BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
9907 046522 005037 003044 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
9908 046526 105037 003070 CLR#B W.TIME ;RESET TIMING ON DRIVE
9909 046532 005037 003104 CLR W.DRV ;CLEAR TIME OUT COUNT
9910 046536 000137 046156 JMP I.ERRA ;GO REPORT ERROR
9911
9912 046542 013701 003060 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
9913 046546 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
9914 046552 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
9915 046556 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
9916 046562 010137 003060 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
9917 046566 016237 000010 002772 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
9918 046574 032737 100000 002772 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
9919 046602 001055 BNE 35$ ;YES, REPORT ERROR
9920 046604 005337 003062 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
9921 046610 001026 BNE 25$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
9922 046612 005037 003044 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
9923 046616 005037 003104 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
9924 046622 105037 003070 CLR#B W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
9925 046626 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
9926 046634 112737 000001 002770 MOV#B #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
9927 046642 004037 050072 JSR RO,I.ISSU ;GET SECTOR COUNT
9928 046646 047772 I.RTRN ;ERROR RETURN
9929 046650 013765 003016 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

E01

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB *READ ALL HEADERS INTERRUPT SEQUENCE

MACY11 27(732) 03-NOV-76 21:43 PAGE 212

```

9930 046656 004737 051006      JSR   PC,R.NORM      ;INDICATE NORMAL TERMINATION
9931 046662 000137 047772      JMP   I.RTRN        ;RESTORE REGISTERS
9932
9933 046666 016562 000002 000020 25$:  MOV   P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
9934 046674 016562 000004 000006      MOV   P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
9935 046702 115565 000007 000017      MCVB  P.CSIH(R5),P.CSI+1(R5) ;STORE BITS 8-15 OF CSI
9936 046710 042765 165777 000016      BIC   #t<CDT!CFMT>,P.CSI(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
9937                                     ; DRIVE TYPE
9938 046716 112765 000125 000016      MCVB  #RDHEAD,P.CSI(R5) ;STORE COMMAND ISSUED
9939 046724 016562 000016 000000      MOV   P.CSI(R5),RKCSI(R2) ;ISSUE READ HEADER
9940 046732 000137 047772      JMP   I.RTRN        ;RESTORE REGISTERS
9941
9942 046736 052737 000400 003042 35$:  BIS   #E.DLT,E.CONT      ;SET DATA LATE WHILE UNLOADING HEADER
9943 046744 004737 051020      JSR   PC,R.CONT      ;REPORT ERROR
9944 046750 000137 047772      JMP   I.RTRN        ;RESTORE REGISTERS
9945
9946                                     .SBTTL  *DRIVE ATTENTION SCANNER
9947
9948 046754 016237 000000 002770  I.ATTN: MOV   RKCSI(R2),T.CSI ;STORE COMMAND AND STATUS
9949                                     ; REGISTER 1 FOR COMPARISON
9950 046762 032737 100000 002770      BIT   #CERR,T.CSI     ;CHECK IF CONTROLLER ERROR OCCURRED
9951 046770 001441      BEQ   SS              ;NO, CHECK IF ATTENTION
9952
9953                                     ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
9954 046772 032737 164000 002772      BIT   #DLT!WCE!UPE!NEM,T.CS2
9955
9956 047000 001007      BNE   IS              ;INDICATE ERROR
9957 047002 016237 000014 003006      MOV   RKER(R2),T.ER   ;STORE ERROR REGISTER
9958
9959                                     ; CHECK FOR DATA TRANSFER ERROR TYPE
9960 047010 032737 125700 003006      BIT   #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
9961
9962 047016 001407      BEQ   ZS              ;NO DATA TRANSFER ERROR
9963
9964 047020 052737 000010 003042 1$:  BIS   #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
9965 047026 004737 051020      JSR   PC,R.CONT      ;REPORT ERROR
9966 047032 000137 047772      JMP   I.RTRN        ;RESTORE REGISTERS
9967
9968 047036 013704 002772 2$:  MOV   T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
9969 047042 042704 177770      BIC   #t<DRVMSK>,R4  ;STRIP OFF JUNK
9970 047046 105037 003070      CLRB  W.TIME         ;CLEAR WATCH DOG TIMER
9971 047052 005037 003104      CLR   W.DRV         ;RESET TIMER VALUE
9972 047056 013705 003102      MOV   PBLKT,R5       ;STORE PARAMETER BLOCK ADDRESS IN R5
9973
9974                                     ; CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
9975                                     ; IN PROGRAM DEVICE STATUS REGISTER
9976 047062 042765 000006 000014      BIC   #DRVPOS!DRVPDT,P.PRST(R5)
9977
9978 047070 000137 046164      JMP   I.ERRC        ;GO REPORT ERROR
9979
9980 047074 032737 040000 002770 5$:  BIT   #DI,T.CS1      ;CHECK IF ANY DRIVE ATTENTION
9981 047102 001002      BNE   BS              ;YES, PROCESS INTERRUPT
9982 047104 000137 047772      JMP   I.RTRN        ;RESTORE REGISTERS
9983
9984 047110 016237 000016 003004 6$:  MOV   RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9985 047116 105737 003005      TSTB  T.ASOF+1       ;CHECK IF ANY ATTENTIONS SET

```

F01

DZR6A-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB *DRIVE ATTENTION SCANNER

MACY11 27(732) 03-NOV-76 21:43 PAGE 213

9986	047122	001007				BNE	7\$:YES GO PROCESS INTERRUPT
9987	047124	052737	000002	003042		BIS	#E.NOAT,E.CONT	:SET NO ATTENTION IN ATTENTION SUMMARY
9988	047132	004737	051020			JSR	PC,R.CONT	:GO REPORT ERROR
9989	047136	000137	047772			JMP	I.RTRN	:GO RESTORE REGISTERS
9990								
9991	047142	133737	003071	003005	7\$:	BITB	INTMSK,T.ASOF+1	:CHECK IF DESIRED INTERRUPT
9992	047150	001007				BNE	8\$:YES, GO PROCESS IT
9993	047152	052737	000004	003042		BIS	#E.UATT,E.CONT	:SET UNSOLICATED ATTENTION
9994	047160	004737	051020			JSR	PC,R.CONT	:GO REPORT ERROR
9995	047164	000137	047772			JMP	I.RTRN	:GO RESTORE REGISTERS
9996								
9997	047170	013705	003102		8\$:	MOV	PBLKT,R5	:STORE PARAMETER BLOCK TABLE
9998	047174	116504	000000			MOVB	P.DRVN(R5),R4	:STORE DRIVE NUMBER
9999	047200	032765	020000	000014		BIT	#E.UNLD,P.PRST(R5)	:CHECK IF DRIVE UNLOADING
10000	047206	001402				BEQ	11\$:NO, CONTINUE
10001	047210	000137	047672			JMP	I.UNLD	:SERVICE DRIVE IN POSITION AFTER ERROR
10002								
10003	047214	042765	000002	000014	11\$:	BIC	#DRVPOS,P.PRST(R5)	:RESET DRIVE POSITIONING
10004	047222	005062	000026			CLR	RKMRI(R2)	:CLEAR MAINTENANCE REGISTER 1
10005	047226	112737	000001	002770		MOVB	#DR.SEL,T.CS1	:LOAD COMMAND
10006	047234	004037	050072			JSR	RO,I.ISSU	:SELECT DRIVE WITH ATTENTION HIGH
10007	047240	047772				I.RTRN		:ERROR RETURN
10008	047242	013765	003016	000042		MOV	T.MR3,P.B00(R5)	:STORE STATUS BYTE 00 MESS B
10009	047250	032765	000200	000042		BIT	#S.FLT,P.B00(R5)	:CHECK IF DRIVE FAULT
10010	047256	001401				BEQ	12\$:NO, CHECK FOR DRIVE STATUS CHANGE
10011	047260	000461				BR	I.AERR	:PROCESS ERROR
10012								
10013	047262	013765	003014	000040	12\$:	MOV	T.MR2,P.A00(R5)	:STORE MAINTENANCE REGISTER 2
10014	047270	032765	040000	000040		BIT	#S.DSC,P.A00(R5)	:CHECK FOR DRIVE STATUS CHANGE
10015	047276	001004				BNE	13\$:YES, PROCESS DRIVE STATUS CHANGE
10016	047300	052765	004000	000014		BIS	#NODSC,P.PRST(R5)	:SET NO DRIVE STATUS CHANGE
10017	047306	000446				BR	I.AERR	:PROCESS ERROR
10018								
10019	047310	112737	000005	002770	13\$:	MOVB	#DR.CLR,T.CS1	:LOAD COMMAND
10020	047316	004037	050072			JSR	RO,I.ISSU	:CLEAR DRIVE STATUS CHANGE
10021	047322	047772				I.RTRN		:ERROR RETURN
10022	047324	013765	003004	000032		MOV	T.ASOF,P.ASOF(R5)	:STORE ATTENTION SUMMARY
10023	047332	133765	003071	000033		BITB	INTMSK,P.ASOF+1(R5)	:CHECK IF ATTENTION RESET
10024	047340	001407				BEQ	15\$:YES, CONTINUE INTERRUPT PROCESSING
10025	047342	052737	000020	003042		BIS	#E.CLAT,E.CONT	:SET ATTENTION DID NOT RESET
10026								:WITH DRIVE CLEAR
10027	047350	004737	051020			JSR	PC,R.CONT	:FLAG ERROR
10028	047354	000137	047772			JMP	I.RTRN	:RESTORE REGISTERS
10029								
10030	047360	013765	003014	000040	15\$:	MOV	T.MR2,P.A00(R5)	:STORE MAINTENANCE REGISTER 2
10031	047366	032765	040000	000040		BIT	#S.DSC,P.A00(R5)	:CHECK IF DRIVE STATUS CHANGE
10032								:RESET
10033	047374	001404				BEQ	16\$:YES, CONTINUE INTERRUPT PROCESSING
10034	047376	052765	000040	000014		BIS	#DRVDSC,P.PRST(R5)	:SET DRIVE STATUS CHANGE DID NOT CLEAR
10035	047404	000407				BR	I.AERR	:GO PROCESS ERROR
10036								
10037	047406	105037	003070		16\$:	CLRB	W.TIME	:RESET TIMING ON THIS DRIVE
10038	047412	005037	003104			CLR	W.DRV	:CLEAR DRIVE TIMING COUNT
10039	047416	004737	051006			JSR	PC,R.NORM	:REPORT SUCCESSFUL COMMAND COMPLETION
10040	047422	000563				BR	I.RTRN	:RESTORE REGISTERS
10041								

```

10042          .SBTTL  *ATTENTION ERROR HANDLER
10043
10044 047424 042765 000004 000014 I.AERR: BIC      #DRV PDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
10045          ;OF DATA TRANSFER
10046 047432 105037 003070          CLR B      W.TIME      ;CLEAR TIMING FOR THIS DRIVE
10047 047436 005037 003104          CLR      W.DRV      ;RESET WATCH-DOG TIME
10048 047442 042765 177741 000016 BIC      #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
10049 047450 042737 000036 002770 BIC      #36,T.CS1      ;KEEP CURRENT CONTROLLER STATUS
10050 047456 053765 002770 000016 BIS      T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
10051 047464 013765 002772 000020 MOV      T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
10052 047472 013765 002774 000022 MOV      T.WCR,P.WCR(R5)
10053 047500 013765 002776 000024 MOV      T.BA,P.BAR(R5)
10054 047506 013765 003000 000026 MOV      T.DA,P.DTS(R5)
10055 047514 013765 003002 000030 MOV      T.DC,P.DCYL(R5)
10056 047522 013765 003004 000032 MOV      T.ASOF,P.ASOF(R5)
10057 047530 013765 003006 000034 MOV      T.ER,P.ER(R5)
10058 047536 013765 003010 000036 MOV      T.DS,P.DS(R5)
10059 047544 004037 050546          JSR      RD,I.STAT    ;GATHER DRIVE STATUS
10060 047550 047772          I.RTRN    ;ERROR RETURN
10061 047552 112737 000005 002770 MOV B     #DR.CLR,T.CS1 ;LOAD COMMAND
10062 047560 004037 050072          JSR      RD,I.ISSU   ;CLEAR DRIVE ERRORS
10063 047564 047772          I.RTRN    ;ERROR RETURN
10064 047566 133737 003071 003005 BIT B     INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
10065 047574 001407          BEQ      2$        ;YES, FLAG DRIVE ERROR
10066 047576 052737 000020 003042 BIS      #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
10067 047604 004737 051020          JSR      PC,R.CONT  ;REPORT ERROR
10068 047610 000137 047772          JMP      I.RTRN    ;RESTORE REGISTERS
10069
10070 047614 032765 000020 000014 2$: BIT     #DRVHLD,P.PRST(R5) ;CHECK IF AWARD DRIVE ERROR
10071 047622 001017          BNE      10$      ;YES, REPORT ERROR
10072 047624 032737 020000 003014 BIT     #S.PIP,T.MR2   ;CHECK IF DRIVE IS UNLOADING
10073 047632 001413          BEQ      10$      ;NO, REPORT ERROR
10074 047634 052765 020000 000014 BIS      #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
10075 047642 113737 003071 003070 MOV B     INTMSK,W.TIME ;SET TIMING ON THIS DRIVE
10076 047650 013737 003054 003104 MOV      W.BSEC,W.DRV ;LOAD 8 SECONDS FOR CYCLE UP TIME
10077 047656 000137 047772          JMP      I.RTRN    ;RESTORE REGISTERS
10078
10079 047662 004737 050774          10$: JSR      PC,R.ABNL  ;REPORT ERROR
10080 047666 000137 047772          JMP      I.RTRN    ;RESTORE REGISTERS
10081
10082          .SBTTL  *ERROR CAUSING DRIVE TO UNLOAD
10083
10084 047672 052765 020000 000014 I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
10085 047700 112737 000005 002770 MOV B     #DR.CLR,T.CS1 ;LOAD IN DRIVE CLEAR
10086 047706 004037 050072          JSR      RD,I.ISSU   ;GO ISSUE DRIVE CLEAR
10087 047712 047772          I.RTRN    ;ERROR RETURN
10088 047714 136437 003071 003005 BIT B     INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
10089 047722 001406          BEQ      15$      ;YES, CONTINUE
10090 047724 012737 000020 003042 MOV      #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
10091 047732 004737 051020          JSR      PC,R.CONT  ;REPORT ERROR
10092 047736 000415          BR       I.RTRN    ;RESTORE REGISTERS
10093
10094 047740 032737 040000 003014 15$: BIT     #S.DSC,T.MR2   ;CHECK IF DRIVE STAU CHANGE RESET
10095 047746 001403          BEQ      20$      ;YES, CONTINUE
10096 047750 052765 000040 000014 BIS      #DRV DSC,P.PRST(R5) ;SET DRIVE STAU CHANGE DID NOT CLEAR
10097 047756 105037 003070          20$: CLR B     W.TIME      ;RESET TIMING ON THIS DRIVE

```


H01

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6NB.CMB *ERROR CAUSING DRIVE TO UNLOAD

MACY11 27(732) 03-NOV-76 21:43 PAGE 215

10098	047762	005037	003104	CLR	W.DRV	;CLEAR TIME COUNT
10099	047766	004737	050774	JSR	PC,R.ABNL	;REPORT ERROR
10100						
10101	047772	012600		I.RTRN: MOV	(SP)+,R0	;RESTORE R0
10102	047774	012601		MOV	(SP)+,R1	;RESTORE R1
10103	047776	012602		MOV	(SP)+,R2	;RESTORE R2
10104	050000	012603		MOV	(SP)+,R3	;RESTORE R3
10105	050002	012604		MOV	(SP)+,R4	;RESTORE R4
10106	050004	012605		MOV	(SP)+,R5	;RESTORE R5
10107	050006	000002		RTI		;RETURN
10108						

I01

```
10109 .SBTTL *CONTROLLER CLEAR ROUTINE
10110
10111 ;*****
10112 ;
10113 ; THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
10114 ; AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT
10115 ; CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
10116 ; E.CCLR SET IN E.CONT.
10117 ;
10118 ; REGISTER USE
10119 ; -----
10120 ;
10121 ; R2 ADDRESS OF RK06 REGISTERS
10122 ; R5 ADDRESS OF PARAMETER BLOCK
10123 ;
10124 ;CALL JSR R0,I.CCLR
10125 ; <ADDRESS OF ERROR RETURN>
10126 ; RETURN
10127 ;
10128 ;*****
10129 ;
10130 050010 012762 100000 000000 I.CCLR: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
10131 050016 016237 000000 002770 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
10132 050024 032737 100000 002770 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER CLEAR DID
10133 ; CLEAR ERROR
10134 050032 001407 BEQ S$ ;YES, RETURN TO DRIVER PROCESSING
10135 050034 052737 000001 003042 BIS #E.CCLR,E.CONT ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
10136 050042 004737 051020 TSK PC,R.CONT ;REPORT CONTROLLER ERROR
10137 050046 011000 MOV (R0),R0 ;SET UP ERROR RETURN
10138 050050 000200 RTS R0 ;RETURN
10139 ;
10140 050052 012762 000100 000000 S$: MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
10141 050060 112737 177777 003064 MOVB #-1,I.ISRL ;SET INTERRUPT ENABLE ISSUED
10142 050066 005720 TST (R0)+ ;ADJUST FOR NORMAL RETURN
10143 050070 000200 RTS R0 ;RETURN
```

10144
10145
10146
10147
10148
10149
10150
10151
10152
10153
10154
10155
10156
10157
10158
10159
10160
10161
10162
10163
10164
10165
10166
10167
10168
10169
10170
10171
10172 050072 013746 002770
10173 050076 005037 002772
10174 050102 116537 000000 002772
10175 050110 013762 002772 000010
10176 050116 116537 000007 002771
10177 050124 142737 177753 002771
10178
10179 050132 013762 002770 000000
10180 050140 105762 000000
10181 050144 100375
10182 050146 004737 050314
10183 050152 032737 100000 002770
10184 050160 001437
10185 050162 032737 001000 002772
10186 050170 001406
10187 050172 052737 100000 003042
10188 050200 004737 051020
10189 050204 000440
10190
10191
10192 050206 032737 024000 002770 25:
10193 050214 001027
10194 050216 032737 176400 002772
10195 050224 001023
10196 050226 032737 131761 003006
10197 050234 001017
10198
10199 050236 122716 000005

.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          R0,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     #1<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
        ;          FORMAT AND DRIVE TYPE
15:     MOV      T.CS1,RKCS1(R2) ;ISSUE COMMAND
        TSTB     RKCS1(R2)      ;WAIT FOR READY
        BPL     15$
        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,R.CONT       ;REPORT CONTROLLER ERROR
        BR      10$            ;RETURN

        ;CHECK IF ANY CONTROLLER ERROR IS SET
25:     BIT     #CTO!SPAR,T.CS1
        BNE     7$
        BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
        BNE     7$
        BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
        BNE     7$

        CMPB     #DR.CLR,(SP)   ;CHECK IF CLEAR DRIVE
```

K01

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2 MACY11 27(732) 03-NOV-76 21:43 PAGE 218
 DZR6NB.CMB *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

10200	050242	001003				BNE	3\$;NO, DO NOT SET DRIVE HARD ERROR
10201	050244	052765	000020	000014		BIS	#DRVHRD,P.PRST(R5)		;SET HARD DRIVE ERROR
10202	050252	004037	050010		3\$:	JSR	RO,I.CCLR		;GO ISSUE A CONTROLLER CLEAR
10203	050256	050306				10\$;ERROR RETURN
10204	050260	012762	000100	000000	5\$:	MOV	#IE,RKCS1(R2)		;SET INTERRUPT ENABLE
10205	050266	005726				TST	(SP)+		;ADJUST STACK
10206	050270	005720				TST	(RO)+		;ADJUST RO FOR NORMAL RETURN
10207	050272	000200				RTS	RO		;RETURN
10208									
10209	050274	052737	001000	003042	7\$:	BIS	#E.CERR,E.CONT		;SET CONTROLLER ERROR DURING
10210									; DRIVER SERVICING
10211	050302	004737	051020			JSR	PC,R.CONT		;REPORT ERROR
10212	050306	005726			10\$:	TST	(SP)+		;ADJUST STACK
10213	050310	011000				MOV	(RO),RO		;ADJUST RO FOR ERROR RETURN
10214	050312	000200				RTS	RO		;RETURN

10215
10216
10217
10218
10219
10220
10221
10222
10223
10224
10225
10226
10227
10228
10229
10230
10231
10232
10233
10234
10235
10236
10237
10238
10239
10240
10241
10242
10243
10244
10245
10246

.SBTTL *STORE RK611 UNIBUS REGISTERS

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

050314 016237 000000 002770
050322 016237 000010 002772
050330 016237 000002 002774
050336 016237 000004 002776
050344 016237 000006 003000
050352 016237 000012 003010
050360 016237 000014 003006
050366 016237 000016 003004
050374 016237 000020 003002
050402 016237 000026 003012
050410 016237 000034 003014
050416 016237 000036 003016
050424 016237 000030 003020
050432 016237 000032 003022
050440 000207

```
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
```

10247
10248
10249
10250
10251
10252
10253
10254
10255
10256
10257
10258
10259
10260
10261
10262
10263
10264
10265
10266
10267
10268
10269
10270
10271
10272
10273
10274
10275
10276
10277
10278
10279
10280
10281
10282
10283
10284
10285
10286
10287
10288
10289
10290
10291

.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 BASE ADDRESS
R5	ADDRESS OF PARAMETER BLOCK

```

050442 J42765 177741 000016 I.CSTS: BIC #177741,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FUNCTION
;OF LAST COMMAND ISSUED
050450 042737 000036 002770 BIC #36,T.CS1 ;CLEAR FUNCTION OF CS1 STATUS
050456 053765 002770 000016 BIS T.CS1,P.CS1(R5) ;GENERATE CS1 STATUS INFORMATION
050464 016265 000010 000020 I.CST1: MOV RKCS2(R2),P.CS2(R5) ;STORE COMMAND AND STATUS REGISTER 2
050472 016265 000002 000022 MOV RKWC(R2),P.WCR(R5) ;STORE WORD COUNT REGISTER
050500 016265 000004 000024 MOV RKBA(R2),P.BAR(R5) ;STORE BUS ADDRESS REGISTER
050506 016265 000006 000026 MOV RKDA(R2),P.DTS(R5) ;STORE DESIRED TRACK AND SECTOR
050514 016265 000012 000036 MOV RKDS(R2),P.DS(R5) ;STORE DRIVE STATUS REGISTER
050522 016265 000014 000034 MOV RKER(R2),P.ER(R5) ;STORE ERROR REGISTER
050530 016265 000016 000032 MOV RKASOF(R2),P.ASOF(R5) ;STORE ATTENTION SUMMARY AND
;OFFSET
050536 016265 000020 000030 MOV RKDCYL(R2),P.DCYL(R5) ;STORE CYLINDER ADDRESS
050544 000207 RTS PC ;RETURN

```

.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 RO,I,STAT
<ADDRESS OF ERROR RETURN>
RETURN

THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:

REGISTER

CONTENTS

R2
R5

RK06 BASE ADDRESS
ADDRESS OF PARAMETER BLOCK

ROUTINES USED:
I.ISSU

10292
10293
10294
10295
10296
10297
10298
10299
10300
10301
10302
10303
10304
10305
10306
10307
10308
10309
10310
10311
10312
10313
10314
10315
10316
10317
10318
10319
10320
10321
10322
10323
10324
10325
10326
10327
10328
10329
10330
10331
10332
10333
10334
10335
10336
10337
10338
10339
10340
10341
10342
10343
10344
10345
10346
10347

050546 012762 000001 000026
050554 112737 000001 002770
050562 004037 050072
050566 050756
050570 013765 003014 000044
050576 013765 003016 000046
050604 012762 000002 000026
050612 112737 000001 002770
050620 004037 050072
050624 050756
050626 013765 003014 000050
050634 013765 003016 000052
050642 012762 000003 000026
050650 112737 000001 002770
050656 004037 050072
050662 050756
050664 013765 003014 000054
050672 013765 003016 000056
050700 005062 000026
050704 112737 000001 002770
050712 004037 050072
050716 050756
050720 013765 003014 000040
050726 013765 003016 000042
050734 032737 001000 003016
050742 001407

I. STAT: MOV #1,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
; FOR STATUS BYTE 01
MOV #DR_SEL,T_CS1 ;LOAD COMMAND
JSR RO,I_ISSU ;GET STATUS BYTES 01
3\$;ERROR RETURN
MOV T_MR2,P_A01(R5) ;STORE STATUS BYTE 01 MESS A
MOV T_MR3,P_B01(R5) ;STORE STATUS BYTE 01 MESS B
MOV #2,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
; FOR STATUS BYTE 10
MOV #DR_SEL,T_CS1 ;LOAD COMMAND
JSR RO,I_ISSU ;GET STATUS BYTES 10
3\$;ERROR RETURN
MOV T_MR2,P_A10(R5) ;STORE STATUS BYTE 10 MESS A
MOV T_MR3,P_B10(R5) ;STORE STATUS BYTE 10 MESS B
MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER
; FOR STATUS BYTE 11
MOV #DR_SEL,T_CS1 ;LOAD COMMAND
JSR RO,I_ISSU ;GET STATUS BYTES 11
3\$;ERROR RETURN
MOV T_MR2,P_A11(R5) ;STORE STATUS BYTE 11 MESS A
MOV T_MR3,P_B11(R5) ;STORE STATUS BYTE 11 MESS B
CLR RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
; FOR STATUS BYTE 00
MOV #DR_SEL,T_CS1 ;LOAD COMMAND
JSR RO,I_ISSU ;GET STATUS BYTES 00
3\$;ERROR RETURN
MOV T_MR2,P_A00(R5) ;STORE STATUS BYTE 00 MESS A
MOV T_MR3,P_B00(R5) ;STORE STATUS BYTE 00 MESS B
BIT #S_PAR,T_MR3 ;CHECK IF BAD PARITY DETECTED BY DRIVE
BEQ 5\$;NO, RETURN NORMALLY

10348	050744	052737	002000	003042		BIS	#E.DPAR,E.CONT	:INDICATE BAD PARITY DETECTED BY DRIVE
10349	050752	004737	051020			JSR	PC,R.CONT	:REPORT ERROR
10350	050756	011000			3\$:	MOV	(R0),R0	:LOAD R0 FOR ERROR RETURN
10351	050760	000200				RTS	R0	:RETURN
10352								
10353	050762	052765	001000	000014	5\$:	BIS	#PBSVAL,P.PRST(R5)	:SET PARAMETER BLOCK STATUS VALID
10354	050770	005720				TST	(R0)+	:ADJUST R0 FOR NORMAL RETURN
10355	050772	000200				RTS	R0	:RETURN
10356								

```

10357          .SBTTL  *COMMON DRIVER RETURNS
10358
10359 050774 105037 003071  P.ABNL: CLRB  INTMSK  ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10360 051000 004777 132032      JSR    PC, @A.ABNL ;INDICATE ABNORMAL RETURN
10361 051004 000207      RTS    PC          ;RETURN
10362
10363 051006 105037 003071  R.NORM: CLRB  INTMSK  ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10364 051012 004777 132016      JSR    PC, @A.NORM ;INDICATE NORMAL RETURN
10365 051016 000207      RTS    PC          ;RETURN
10366
10367 051020 105037 003071  R.CONT: CLRB  INTMSK  ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10368 051024 105037 003070      CLRB  W.TIME   ;RESET WATCH DOG TIMING CN THIS DRIVE
10369 051030 005037 003104      CLR   W.DRV   ;CLEAR TIMING COUNT FOR THIS DRIVE
10370 051034 004777 13200C      JSR    PC, @A.CONT ;INDICATE CONTROLLER ERROR RETURN
10371 051040 000207      RTS    PC          ;RETURN

```


10372
10373
10374
10375
10376
10377
10378
10379
10380
10381
10382
10383
10384
10385
10386
10387
10388
10389
10390
10391
10392
10393
10394
10395
10396
10397
10398
10399
10400
10401
10402
10403
10404
10405
10406
10407
10408
10409
10410
10411
10412
10413
10414
10415
10416
10417
10418
10419
10420
10421
10422
10423
10424
10425
10426
10427

.SBTTL *COMMAND INITIATOR

```
*****
*
* 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
*     .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    #1<DRVMSK>, R4 ;MASK OUT JUNK
        MOV    R5, PBLKT      ;LOAD PARAMETER BLOCK TABLE
        MOVB  I.DRV(R4), INTMSK ;LOAD INTERRUPT MASK
        MOVB  I.DRV(R4), W.TIME ;SET WATCH-DOG TIMER FLAG
        MOV    W.SEC, W.DRV    ;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
        ;
```

```
051042 010546
051044 010446
051046 010346
051050 010246
051052 010146
051054 010046
051056 013746 177776
051062 013737 003032 177776
051070 017605 000016
051074 062766 000002 000016
051102 016504 000000
051106 042704 177770
051112 010537 003102
051116 116437 003072 003071
051124 116437 003072 003070
051132 013737 003052 003104
051140 013702 003026
```

E02

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB *COMMAND INITIATOR

MACY11 27(732) 03-NOV-76 21:43 PAGE 225

10428	051144	042765	075176	000014		BIC	#↑C<DRVUSE!W.WCK!NOCHK!DRPDRY!DTBAII>,P.PRST(R5)
10429							
10430	051152	010500				MOV	R5,R0 ;STORE PARAMETER BLOCK ADDRESS
10431	051154	062700	000016			ADD	#P.CS1,R0 ;CALCULATE FIRST LOCATION TO BE CLEARED
10432	051160	010501				MOV	R5,R1 ;STORE PARAMETER BLOCK ADDRESS
10433	051162	062701	000062			ADD	#P.EPAT,R1 ;CALCULATE LAST LOCATION TO BE CLEARED
10434							
10435	051166	005020			1\$:	CLR	(R0)+ ;CLEAR RETURN PARAMETER
10436	051170	020001				CMP	R0,R1 ;CHECK IF FINISHED
10437	051172	101775				BLOS	1\$;NO, CLEAR NEXT RETURN PARAMETER
10438	051174	105037	003064			CLRB	I.ISRL ;CLEAR RELEASE OR INTERRUPT ISSUED
10439	051200	010465	000020			MOV	R4,P.CS2(R5) ;STORE DRIVE NUMBER
10440	051204	005062	000026			CLR	RKMR1(R2) ;CLEAR RK06 MAINTENANCE REGISTER 1
10441	051210	132765	000040	000001		BITB	#BITS,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
10442	051216	001402				BEG	3\$;NO, PROCESS
10443	051220	000137	051734			JMP	C.SPEC ;JUMP TO SPECIAL COMMAND PROCESSOR
10444							
10445	051224	122765	000107	000001	3\$:	CMPB	#UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
10446							START SPINDLE
10447							RECALIBRATE
10448							OFFSET
10449							SEEK
10450							UNLOAD
10451							
10452	051232	101174				BHI	25\$;NO, DRIVE COMMAND
10453							SELECT DRIVE
10454							PACK ACKNOWLEDGE
10455							CLEAR
10456							
10457	051234	122765	000117	000001		CMPB	#SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
10458	051242	103540				BLO	20\$;YES, DATA TRANSFER COMMAND
10459							READ DATA
10460							WRITE DATA
10461							READ HEADER
10462							WRITE HEADER
10463							WRITE CHECK
10464	051244	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10465	051252	052765	000002	000014		BIS	#DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
10466	051260	005037	003044			CLR	O.WAIT ;CLEAR WAIT FOR COMMAND
10467	051264	122765	000117	000001		CMPB	#SEEK,P.CMND(R5) ;CHECK IF SEEK
10468	051272	001007				BNE	5\$;NO, CHECK FOR OFFSET
10469	051274	016562	000002	000020		MOV	P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10470	051302	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10471	051310	000431				BR	8\$;GO ISSUE COMMAND
10472							
10473	051312	122765	000115	000001	5\$:	CMPB	#OFFSET,P.CMND(R5) ;CHECK IF OFFSET
10474	051320	001007				BNE	6\$;NO, CHECK FOR UNLOAD
10475	051322	116565	000006	000032		MOVB	P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
10476	051330	016562	000032	000016		MOV	P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
10477	051336	000416				BR	8\$;GO ISSUE COMMAND
10478							
10479	051340	122765	000111	000001	6\$:	CMPB	#SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
10480	051346	001003				BNE	7\$;NO, CHECK IF RECAL
10481	051350	013737	003056	003104		MOV	W.MIN,W.DRV ;LOAD WATCH DOG TIME FOR 1 MINUTE
10482	051356	122765	000113	000001	7\$:	CMPB	#RECAL,P.CMND(R5) ;CHECK IF RECAL
10483	051364	001003				BNE	8\$;NO, CONTINUE

10484	051366	013737	003054	003104		MOV	W.BSEC,W.DRV	:LOAD RECAL TIME FOR 8 SECONDS
10485	051374	116565	000007	000017	85:	MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10486	051402	042765	165777	000016		BIC	#1C<CFMT!CDT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT
10487								: AND DRIVE TYPE
10488	051410	116565	000001	000016		MOVB	P.CMND(R5),P.CS1(R5)	:MOVE COMMAND INTO CS1
10489	051416	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:RESET WRITE FOR WRITE CHECK
10490	051424	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IN NO CHECK MODE
10491	051432	001533				BEQ	30\$:NO, SKIP CLEAR OF INTERRUPT ENABLE
10492	051434	042765	000100	000016		BIC	#IE,P.CS1(R5)	:CLEAR INTERRUPT ENABLE
10493	051442	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
10494	051450	004737	045412		10\$:	JSR	PC.W.WTCH	:CALL WATCH DOG TIMER
10495	051454	016237	000000	002770		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REGISTER 1
10496	051462	032737	000200	002770		BIT	#RDY,T.CS1	:WAIT FOR READY
10497	051470	001767				BEQ	10\$	
10498	051472	032737	100000	002770		BIT	#CERR,T.CS1	:CHECK FOR ERROR
10499	051500	001011				BNE	15\$:YES, GIVE NORMAL RETURN
10500	051502	004737	045412		11\$:	JSR	PC.W.WTCH	:CALL WATCH DOG TIMER
10501	051506	016237	000016	003004		MOV	RKASOF(R2),T.ASOF	:STORE ATTENTION SUMMARY
10502	051514	133737	003071	003005		BITB	INTMSK,T.ASOF+1	:CHECK IF INTERRUPT HAS OCCURRED
10503	051522	001767				BEQ	11\$:WAIT FOR DRIVE INTERRUPT
10504	051524	105037	003070		15\$:	CLRB	W.TIME	:RESET TIMING ON THIS DRIVE
10505	051530	005037	003104			CLR	W.DRV	:CLEAR DRIVE TIMING COUNT
10506	051534	004737	051006			JSR	PC,R.NORM	:INDICATE COMMAND IS FINISHED
10507	051540	000137	052714			JMP	C.RTRN	:RESTORE REGISTERS
10508								
10509	051544	016562	000010	000004	20\$:	MOV	P.BALO(R5),RKBA(R2)	:LOAD BUS ADDRESS REGISTER
10510	051552	016562	000012	000002		MOV	P.WC(R5),RKWC(R2)	:LOAD WORD COUNT REGISTER
10511	051560	016562	000002	000020		MOV	P.CYLN(R5),RKDCYL(R2)	:LOAD CYLINDER ADDRESS REGISTER
10512	051566	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	:LOAD SECTOR AND TRACK NUMBER
10513	051574	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK COMMAND
10514	051602	001010				BNE	25\$:NO, GO ISSUE THE COMMAND
10515	051604	032765	000200	000014		BIT	#W.WCK,P.PRST(R5)	:CHECK IF WRITE COMMAND SHOULD BE ISSUED
10516	051612	001404				BEQ	25\$:NO, GO ISSUE THE COMMAND
10517	051614	012765	000123	000016		MOV	#WRDATA,P.CS1(R5)	:ISSUE WRITE COMMAND
10518	051622	000406				BR	26\$:GO ISSUE COMMAND
10519								
10520	051624	116565	000001	000016	25\$:	MOVB	P.CMND(R5),P.CS1(R5)	:MOVE COMMAND INTO CS1
10521	051632	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:RESET WRITE FOR WRITE CHECK
10522	051640	116565	000007	000017	26\$:	MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10523	051646	142765	177750	000017		BICB	#1C<B.CFMT!B.CDT!B.BAI6!B.BAI7>,P.CS1+1(R5)	:CLEAR ALL BITS EXCEPT
10524								: FORMAT, DRIVE TYPE, AND BUS ADDRESS
10525								: BITS 16-17
10526	051654	010537	003044			MOV	R5,0.WAIT	:LOAD WAITING FOR COMMAND
10527	051660	032765	100000	000014		BIT	#DTBAII,P.PRST(R5)	:CHECK IF INHIBIT BUS ADDRESS INCREMENT
10528	051666	001403				BEQ	27\$:NO, LOAD CS2
10529	051670	052765	000020	000020		BIS	#BAI,P.CS2(R5)	:SET IN-BIT BUS ADDRESS INCREMENT
10530	051676	016562	000020	000010	27\$:	MOV	P.CS2(R5),RKCS2(R2)	:LOAD CS2
10531	051704	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IN NO CHECK MODE
10532	051712	001403				BEQ	30\$:NO, SKIP CLEAR OF INTERRUPT ENABLE
10533	051714	042765	000100	000016		BIC	#IE,P.CS1(R5)	:CLEAR INTERRUPT ENABLE
10534	051722	016562	000016	000000	30\$:	MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
10535	051730	000137	052714			JMP	C.RTRN	:RESTORE REGISTERS
10536								
10537						.SBTTL	*SPECIAL COMMAND PROCESSING	
10538								
10539	051734	122765	000141	000001	C.SPEC:	CMPB	#RDSTAT,P.CMND(R5)	:CHECK IF READ DRIVE STATUS

10596	052302	112765	000101	000016		MOV	#SELDRV,P.CS1(R5) ;STORE COMMAND
10597	052310	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10598	052316	001403				BEQ	11\$;NO, DO NOT RESET INTERRUPT ENABLE
10599	052320	042765	000100	000016		BIC	#IE,P.CS1(R5) ;RESET INTERRUPT ENABLE
10600	052326	016562	000016	000000	11\$:	MOV	P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10601	052334	000137	052714			JMP	C.RTRN ;RESTORE REGISTERS
10602							
10603	052340	122765	000164	000001	13\$:	CMPB	#RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
10604	052346	001053				BNE	30\$;NO, CHECK IF CONTROLLER CLEAR
10605	052350	010537	003044			MOV	RS,O.WAIT ;SET WAITING FOR COMMAND COMPLETION
10606	052354	016537	000010	003060		MOV	P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
10607	052362	132765	000020	000007		BITB	#B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
10608	052370	001404				BEQ	14\$;YES, LOAD 22 IN HEADER COUNT
10609	052372	012737	000024	003062		MOV	#20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
10610	052400	000403				BR	22\$;GO ISSUE READ HEADER COMMAND
10611							
10612	052402	012737	000026	003062	14\$:	MOV	#22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
10613	052410	016562	000002	000020	22\$:	MOV	P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10614	052416	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
10615	052424	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10616	052432	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10617	052440	042765	165777	000016		BIC	#1C<CFMT!COT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
10618							AND FORMAT
10619	052446	112765	000125	000016		MOVB	#RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
10620	052454	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10621	052462	001027				BNE	34\$;YES, INDICATE ILLEGAL DRIVER COMMAND
10622	052464	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
10623	052472	000137	052714			JMP	C.RTRN ;RESTORE REGISTERS
10624							
10625	052476	122765	000176	000001	30\$:	CMPB	#CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
10626	052504	001012				BNE	32\$;NO, CHECK IF SUBSYSTEM CLEAR
10627	052506	004037	050010			JSR	RO,I.CCLR ;CLEAR CONTROLLER
10628	052512	052714				C.RTRN	ERROR RETURN
10629	052514	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10630	052522	001472				BEQ	40\$;NO, INDICATE NORMAL RETURN
10631	052524	005062	000000			CLR	RKCS1(R2) ;RESET INTERRUPT ENABLE
10632	052530	000467				BR	40\$;INDICATE NORMAL RETURN
10633							
10634	052532	122765	000177	000001	32\$:	CMPB	#SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
10635	052540	001406				BEQ	36\$;YES, CLEAR SUBSYSTEM
10636	052542	052737	000100	003042	34\$:	BIS	#E.ILLD,E.CONT ;SET ILLEGAL DRIVER COMMAND
10637	052550	004737	051020			JSR	PC,R.CONT ;REPORT ERROR
10638	052554	000457				BR	C.RTRN ;RESTORE REGISTERS
10639							
10640	052556	012762	000040	000010	36\$:	MOV	#SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
10641	052564	016265	000000	000016		MOV	RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
10642	052572	032765	100000	000016		BIT	#CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
10643	052600	001406				BEQ	37\$;NO, FINISH COMMAND
10644	052602	052737	000001	003042		BIS	#BITO,E.CONT ;SET CLEAR SUBSYSTEM DID NOT CLEAR
10645							CONTROLLER ERROR
10646	052610	004737	051020			JSR	PC,R.CONT ;REPORT ERROR
10647	052614	000437				BR	C.RTRN ;RESTORE REGISTERS
10648							
10649	052616	013746	003050		37\$:	MOV	W.MILI,-(SP) ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
10650							TO DISAPPEAR
10651	052622	016265	000000	000016	38\$:	MOV	RKCS1(R2),P.CS1(R5) ;STORE CS1

```

10652 052630 032765 040000 000016 BIT #DI,P.CS1(R5) ;CHECK IF ATTENTIONS CLEARED
10653 052636 001411 BEQ 39$ ;YES, FINISH COMMAND
10654 052640 005316 DEC (SP) ;DECREMENT 16 MILISECCND COUNT
10655 052642 001367 BNE 38$ ;CHECK DRIVE INTERRUPT AGAIN
10656 052644 005726 TST (SP)+ ;ADJUST STACK
10657 052646 052737 000040 003042 BIS #E.SCLR,E.CONT ;SET SUBSYSTEM CLEAR DID NCT CLEAR
10658 ; DRIVE ATTENTIONS
10659 052654 004737 051020 JSR PC,R.CONT ;REPORT ERROR
10660 052660 000415 BR C.RTRN ;RESTORE REGISTER
10661
10662 052662 005726 39$: TST (SP)+ ;ADJUST STACK
10663 052664 032765 000400 000014 BIT #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10664 052672 001010 BNE C.RTRN ;YES, RESTORE REGISTERS
10665 052674 112737 177777 003064 MOVB #-1,I.ISRL ;SET INTERRUPT ENABLE SET
10666 052702 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
10667 052710 004737 051006 40$: JSR PC,R.NORM ;INDICATE NORMAL TERMINATION
10668
10669 052714 012637 177776 C.RTRN: MOV (SP)+,PS ;RESTORE PSW
10670 052720 012600 MOV (SP)+,R0 ;RESTORE R0
10671 052722 012601 MOV (SP)+,R1 ;RESTORE R1
10672 052724 012602 MOV (SP)+,R2 ;RESTORE R2
10673 052726 012603 MOV (SP)+,R3 ;RESTORE R3
10674 052730 012604 MOV (SP)+,R4 ;RESTORE R4
10675 052732 012605 MOV (SP)+,R5 ;RESTORE R5
10676 052734 000207 RTS PC ;RETURN
10677
10678 .SBTTL OCTAL TO BINARY CONVERSION ROUTINE
10679
10680 ;*****
10681 ;
10682 ; THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10683 ; WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL
10684 ; IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10685 ; ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HI0CT.
10686 ;
10687 ;CALL
10688 ; MOV <ADDRESS OF ASCII STRING>,-(SP)
10689 ; JSR PC,OCTBIN
10690 ; <ADDRESS OF ERROR RETURN>
10691 ; RETURN
10692 ;*****
10693
10694 052736 010046 OCTBIN: MOV R0,-(SP) ;SAVE R0
10695 052740 010146 MOV R1,-(SP) ;SAVE R1
10696 052742 010246 MOV R2,-(SP) ;SAVE R2
10697 052744 016600 000010 MOV 10(SP),R0 ;GET ADDRESS OF ASCII STRING
10698 052750 005001 CLR R1 ;CLEAR DATA WORDS
10699 052752 005002 CLR R2
10700 052754 112046 2$: MOVB (R0)+,-(SP) ;PICK THIS CHARACTER
10701 052756 001423 BEQ 3$ ;IF ZERO GET OUT
10702 052760 121627 000054 CMPB (SP),#', ;CHECK IF COMMA
10703 052764 001420 BEQ 3$ ;IF COMMA GET OUT
10704 052766 122716 000060 CMPB #'0,(SP) ;MAKE SURE THIS CHARACTER IS
10705 052772 003030 BGT 4$ ; AN OCTAL DIGIT
10706 052774 122716 000067 CMPB #'7,(SP)
10707 053000 002425 BLT 4$

```

```

10708 053002 006301          ASL    R1          ; *2
10709 053004 006102          ROL    R2
10710 053006 006301          ASL    R1          ; *4
10711 053010 006102          ROL    R2
10712 053012 006301          ASL    R1          ; *8
10713 053014 006102          ROL    R2
10714 053016 042716 177770  BIC    #1C7,(SP)   ;STRIP THE ASCII JUNK
10715 053022 062601          ADD    (SP)+,R1   ;ADD THIS DIGIT
10716 053024 000753          BR     2$         ;LOOP
10717 053026 005726          3$:   TST    (SP)+   ;CLEAN PARTIAL FROM STACK
10718 053030 010166 000010  MOV    R1,10(SP)  ;SAVE RESULT
10719 053034 010237 053070  MOV    R2,$HI OCT
10720 053040 012602          MOV    (SP)+,R2   ;RESTORE R2
10721 053042 012601          MOV    (SP)+,R1   ;RESTORE R1
10722 053044 012600          MOV    (SP)+,R0   ;RESTORE R0
10723 053046 062716 000002  ADD    #2,(SP)    ;ADJUST RETURN
10724 053052 000207          RTS     PC        ;RETURN
10725
10726 053054 005726          4$:   TST    (SP)+   ;CLEAN UP PARTIAL FROM STACK
10727 053056 012602          MOV    (SP)+,R2   ;RESTORE R2
10728 053060 012601          MOV    (SP)+,R1   ;RESTORE R1
10729 053062 012600          MOV    (SP)+,R0   ;RESTORE R0
10730 053064 013616          MOV    2(SP)+,(SP) ;PUT ADDRESS OF ERROR ROUTINE ON STACK
10731 053066 000207          RTS     PC        ;GO PROCESS ERROR
10732 053070 000000          $HI OCT: .WORD 0  ;HIGH ORDER BITS GO HERE
10733
10734          .SBTTL RANDOM NUMBER GENERATOR ROUTINE
10735
10736          ;*****
10737          ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
10738          ;*WITH A RANGE OF 0 TO 2(+33)-1.
10739          ;*CALL:
10740          ;*   JSR    PC,$RAND          ;CALL THE ROUTINE
10741          ;*   RETURN                    ;RETURN HERE THE RANDOM
10742          ;*                               ;NUMBER WILL BE IN
10743          ;*                               ;SHINUM,$LONUM
10744
10745          $RAND:
10746          MOV    R0,-(SP)          ;PUSH R0 ON STACK
10747          MOV    R1,-(SP)          ;PUSH R1 ON STACK
10748          MOV    R2,-(SP)          ;PUSH R2 ON STACK
10749          MOV    $LONUM,R0         ;SET R0 WITH LOW
10750          MOV    $SHINUM,R1        ;SET R1 WITH HIGH
10751          MOV    #-7,R2           ;SET SHIFT COUNT
10752          1$: ASL    R0            ;SHIFT R0 LEFT AND
10753          ROL    R1                ;ROTATE CARRY INTO R1 AND
10754          INC    R2                ;CHECK FOR DONE
10755          BNE    1$               ;CONTINUE SHIFT LOOP
10756          ADD    $LONUM,R0         ;ADD NUMBER TO MAKE X 129
10757          ADC    R1                ;PROPOGATE CARRY
10758          ADD    $SHINUM,R1        ;ADD NUMBER TO MAKE X 129
10759          ADD    #1057,R0         ;ADD LOW CONSTANT
10760          ADC    R1                ;PROPOGATE CARRY
10761          ADD    #47401,R1         ;ADD HIGH CONSTANT
10762          MOV    R0,$LONUM         ;SAVE R0
10763          MOV    R1,$SHINUM        ;SAVE R1
10764          MOV    (SP)+,R2         ;POP STACK INTO R2
    
```



```

10764 053162 012501          MOV      (SP)+,R1          ;; POP STACK INTO R1
10765 053164 012600          MOV      (SP)+,R0          ;; POP STACK INTO R0
10766 053166 000207          RTS      PC                ;; RETURN
10767 053170 176543          $HINUM: .WORD 176543
10768 053172 123456          $LONUM: .WORD 123456
10769                                .SBTTL  TYPE ROUTINE
10770
10771                                ;:*****
10772                                ;:ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10773                                ;:THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10774                                ;:NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10775                                ;:NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10776                                ;:NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10777                                ;:
10778                                ;:CALL:
10779                                ;:1) USING A TRAP INSTRUCTION
10780                                ;:      TYPE      ,MESADR          ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
10781                                ;:OR
10782                                ;:      TYPE
10783                                ;:      MESADR
10784                                ;:
10785
10786 053174 105737 001157          $TYPE:  TSTB      $STPFLG          ;; IS THERE A TERMINAL?
10787 053200 100002          BPL      1$                ;; BR IF YES
10788 053202 000000          HALT                    ;; HALT HERE IF NO TERMINAL
10789 053204 000430          BR      3$                ;; LEAVE
10790 053206 010046          1$:  MOV      RD, -(SP)          ;; SAVE RD
10791 053210 017600 000002          MOV      @2(SP), R0          ;; GET ADDRESS OF ASCIZ STRING
10792 053214 122737 000001 001340          CMPB     #APTENV, $ENV          ;; RUNNING IN APT MODE
10793 053222 001011          BNE     62$                ;; NO, GO CHECK FOR APT CONSOLE
10794 053224 132737 000100 001341          BITB     #APTSPool, $ENVM          ;; SPOOL MESSAGE TO APT
10795 053232 001405          BEQ     62$                ;; NO, GO CHECK FOR CONSOLE
10796 053234 010037 053244          MOV      RD, 61$          ;; SETUP MESSAGE ADDRESS FOR APT
10797 053240 004737 055672          JSR     PC, $ATY3          ;; SPOOL MESSAGE TO APT
10798 053244 000000          61$: .WORD 0                ;; MESSAGE ADDRESS
10799 053246 132737 000040 001341          62$: BITB     #APTCsup, $ENVM          ;; APT CONSOLE SUPPRESSED
10800 053254 001003          BNE     60$                ;; YES, SKIP TYPE OUT
10801 053256 112046          2$:  MOVB     (R0)+, -(SP)          ;; PUSH CHARACTER TO BE TYPED ONTO STACK
10802 053260 001005          BNE     4$                ;; BR IF IT ISN'T THE TERMINATOR
10803 053262 005726          TST     (SP)+              ;; IF TERMINATOR POP IT OFF THE STACK
10804 053264 012600          60$: MOV      (SP)+, R0          ;; RESTORE RD
10805 053266 062716 000002          3$:  ADD     #2, (SP)          ;; ADJUST RETURN PC
10806 053272 000002          RTI                    ;; RETURN
10807 053274 122716 000011          4$:  CMPB     #HT, (SP)          ;; BRANCH IF <HT>
10808 053300 001430          BEQ     8$                ;; BRANCH IF NOT <CRLF>
10809 053302 122716 000200          CMPB     #CRLF, (SP)
10810 053306 001006          BNE     5$                ;; POP <CR><LF> EQUIV
10811 053310 005726          TST     (SP)+              ;; TYPE A CR AND LF
10812 053312 104400          TYPE
10813 053314 001315          $CRLF
10814 053316 105037 053452          CLRB     $CHARCNT          ;; CLEAR CHARACTER COUNT
10815 053322 000755          BR      2$                ;; GET NEXT CHARACTER
10816 053324 004737 053406          5$:  JSR     PC, $TYPEC          ;; GO TYPE THIS CHARACTER
10817 053330 123726 001156          6$:  CMPB     $FILLC, (SP)+          ;; IS IT TIME FOR FILLER CHARS.?
10818 053334 001350          BNE     2$                ;; IF NO GO GET NEXT CHAR.
10819 053336 013746 001154          MOV     $NULL, -(SP)          ;; GET # OF FILLER CHARS. NEEDED

```



```

10820                                     ;; AND THE NULL CHAR.
10821 053342 105366 000001 7$: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
10822 053346 002770          BLT 6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
10823 053350 004737 053406          JSR PC,$TYPEC ;; GO TYPE A NULL
10824 053354 105337 053452          DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
10825 053360 000770          BR 7$      ;; LOOP
10826
10827                                     :HORIZONTAL TAB PROCESSOR
10828
10829 053362 112716 000040 8$: MOVB #' (SP) ;; REPLACE TAB WITH SPACE
10830 053366 004737 053406 9$: JSR PC,$TYPEC ;; TYPE A SPACE
10831 053372 132737 000007 053452 BITB #7,$CHARCNT ;; BRANCH IF NOT AT
10832 053400 001372          BNE 9$      ;; TAB STOP
10833 053402 005726          TST (SP)+ ;; POP SPACE OFF STACK
10834 053404 000724          BR 2$      ;; GET NEXT CHARACTER
10835 053406 105777 125536 $TYPEC: TSTB 2$TPS ;; WAIT UNTIL PRINTER IS READY
10836 053412 100375          BPL $TYPEC
10837 053414 116677 000002 125530 MOVB 2(SP),2$TPB ;; LOAD CHAR TO BE TYPED INTO DATA REG.
10838 053422 122766 000015 000002 CMPB #CR,2(SP) ;; IS CHARACTER A CARRIAGE RETURN?
10839 053430 001003          BNE 1$      ;; BRANCH IF NO
10840 053432 105037 053452          CLRB $CHARCNT ;; YES--CLEAR CHARACTER COUNT
10841 053436 000406          BR $TYPEX ;; EXIT
10842 053440 122766 000012 000002 1$: CMPB #LF,2(SP) ;; IS CHARACTER A LINE FEED?
10843 053446 001402          BEQ $TYPEX ;; BRANCH IF YES
10844 053450 105227          INCB (PC)+ ;; COUNT THE CHARACTER
10845 053452 000000          $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
10846 053454 000207          $TYPEX: RTS PC
10847
10848
10849
10850
10851                                     ;*****
10852 .SBTTL DOUBLE-PRECISION MULTIPLY SUBROUTINE
10853 * SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
10854 * USES ALL REGISTERS (R0-R5)
10855 *
10856 * ENTER WITH JSR PC,M.DPIM
10857 * MULTIPLIER IN R2-R3
10858 * MULTIPLICAND IN R4-R5
10859 * PRODUCT RETURNED IN R0-R1-R2-R3
10860 *
10861 ;*****
10862 053456 005000 M.DPIM: CLR R0 ;CLEAR HI ORDER WORDS
10863 053460 005001          CLR R1
10864 053462 012746 000041          MOV #41,-(SP) ;MOVE 33 (DEC) TO COUNTER
10865 053466 006000 M.DP01: ROR R0
10866 053470 006001          ROR R1
10867 053472 006002          ROR R2 ;SHIFT TO ADD
10868 053474 006003          ROR R3
10869 053476 103003          BCC M.DP02 ;NO CARRY NO ADD
10870 053500 060501          ADD R5,R1
10871 053502 005500          ADC R0 ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
10872 053504 060400          ADL R4,R0 ;PRODUCT
10873 053506 005316 M.DP02: DEC 2$P ;DECREMENT COUNTER
10874 053510 001366          BNE M.DP01
10875 053512 005726          TST (SP)+ ;REMOVE THE COUNTER

```

```

10876 053514 000207
10877
10878
10879
10880
10881
10882
10883
10884
10885
10886
10887
10888
10889
10890
10891 053516 012746 000040
10892 053522 010446
10893 053524 010546
10894 053526 005466 000002
10895 053532 005416
10896 053534 005666 000002
10897 053540 061601
10898 053542 005500
10899 053544 066600 000002
10900 053550 103445
10901 053552 005046
10902 053554 006103
10903 053556 006102
10904 053560 006101
10905 053562 006100
10906 053564 005716
10907 053566 001410
10908 053570 005016
10909 053572 066601 000002
10910 053576 005500
10911 053600 005516
10912 053602 066600 000004
10913 053606 000404
10914 053610 060501
10915 053612 005500
10916 053614 005516
10917 053616 060400
10918 053620 005516
10919 053622 005716
10920 053624 001401
10921 053626 005203
10922 053630 005366 000006
10923 053634 003347
10924 053636 006003
10925 053640 103404
10926 053642 060501
10927 053644 005500
10928 053646 060400
10929 053650 000241
10930 053652 006103
10931 053654 062706 000010

```

```

RTS PC
;*****
;SBTTL DOUBLE-PRECISION DIVIDE SUBROUTINE
;      SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
;      USES ALL REGISTERS (R0-R5)
;
;      ENTER WITH JSR PC,M.DPID
;      DIVIDEND IN R0-R1-R2-R3
;      DIVISOR IN R4-R5
;      REMAINDER RETURNED IN R0-R1
;      QUOTIENT RETURNED IN R2-R3
;*****
M.DPID: MOV #40,-(SP) ;COUNTER FOR DIVISION CYCLES
        MOV R4,-(SP) ;HI ORDER
        MOV R5,-(SP) ;LO ORDER DIVISOR TO THE STACK
        NEG 2(SP) ;FORM NEGATIVE
        NEG @SP ; VERSION OF THE DIVISOR
        SBC 2(SP)
        ADD @SP,R1
        ADC R0 ;PERFORM THE INITIAL SUBTRACTION
        ADD 2(SP),R0
        BCS M.DP50 ;IF CARRY THEN OVERFLOW HAS OCCURRED
        CLR -(SP) ;THIS IS A LONGER LASTING CARRY BIT
M.DP40: ROL R3
        ROL R2
        ROL R1
        ROL R0
        TST @SP ;TEST "CARRY INDICATOR"
        BEQ M.DP41 ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
        CLR @SP ;CLEAR UP FOR NEXT TIME
        ADD 2(SP),R1
        ADC R0 ;ADD -(DIVISOR)
        ADC @SP ;SET "CARRY"
        ADD 4(SP),R0 ;I
        BR M.DP42 ;<
M.DP41: ADD R5,R1
        ADC R0 ;ADD +(DIVISOR)
        ADC @SP ;SET "CARRY"
        ADD R4,R0 ;I
        ADC @SP ;<
M.DP42: ADC @SP ;SET "CARRY"
        TST @SP ;TEST THE UPDATE INDICATOR
        BEQ .+4 ;> ;IF ZERO FORGET IT
        INC R3 ;> ;NO CARRY POSSIBLE HERE
        DEC 6(SP) ;< ;DECREMENT COUNTER
        BGT M.DP40 ;BR IF MORE TO DO
        ROR R3
        BCS M.DP44
        ADD R5,R1
        ADC R0
        ADD R4,R0
M.DP44: CLC
        ROL R3
        ADD #10,SP ;ADJUST STACK BY 4 WORDS

```

10932 053660 000242
 10933 053662 000207
 10934 053664 062706 000006
 10935 053670 000262
 10936 053672 000207

CLV
 RTS PC
 M.DP50: ADD #6,SP
 SEV
 RTS PC

.SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

 ; THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
 ; UNSIGNED OCTAL ASCII NUMBER.

*CALL
 ; * MOV #PNTR, -(SP) ; ; POINTER TO LOW WORD OF BINARY NUMBER
 ; * JSR PC, @#\$DB20 ; ; CALL THE ROUTINE
 ; * RETURN ; ; THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK

10951 053674 104406
 10952 053676 016601 000002
 10953 053702 012705 054013
 10954 053706 012704 000014
 10955 053712 012703 177770
 10956 053716 012100
 10957 053720 012101
 10958 053722 005002
 10959 053724 110245
 10960 053726 010002
 10961 053730 005304
 10962 053732 003007
 10963 053734 001405
 10964 053736 005205
 10965 053740 010566 000002
 10966 053744 104407
 10967 053746 000207
 10968 053750 006203
 10969 053752 006001
 10970 053754 006007
 10971 053756 006001
 10972 053760 006000
 10973 053762 006001
 10974 053764 006000
 10975 053766 040302
 10976 053770 062702 000060
 10977 053774 000753
 10978 053776 000016

\$DB20: SAVREG ; ; SAVE ALL REGISTERS
 MOV 2(SP), R1 ; ; PICKUP THE POINTER TO LOW WORD
 MOV #SOCTVL+13., R5 ; ; POINTER TO DATA TABLE
 MOV #12., R4 ; ; DO ELEVEN CHARACTERS
 MOV #1C7, R3 ; ; MASK
 MOV (R1)+, R0 ; ; LOWER WORD
 MOV (R1)+, R1 ; ; HIGH WORD
 CLR R2 ; ; TERMINATOR
 1\$: MOVB R2, -(R5) ; ; PUT CHARACTER IN DATA TABLE
 MOV R0, R2 ; ; GET THIS DIGIT
 DEC R4 ; ; COUNT THIS CHARACTER
 BGT 3\$; ; BR IF NOT THE LAST DIGIT
 BEQ 2\$; ; BR IF IT IS THE LAST DIGIT
 INC R5 ; ; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
 MOV R5, 2(SP) ; ; ASCII CHAR. & PUT IT ON THE STACK
 RESREG ; ; RESTORE ALL REGISTERS
 RTS PC ; ; RETURN TO USER
 2\$: ASR R3 ; ; POSITION THE MASK FOR THE LAST DIGIT
 3\$: ROR R1 ; ; POSITION THE BINARY NUMBER FOR
 ROR R0 ; ; THE NEXT OCTAL DIGIT
 ROR R1
 ROR R0
 ROR R1
 ROR R0
 ROR R1
 ROR R0
 BIC R3, R2 ; ; MASK OUT ALL JUNK
 ADD #0, R2 ; ; MAKE THIS CHAR. ASCII
 BR 1\$; ; GO PUT IT IN THE DATA TABLE
 SOCTVL: .BLKB 14. ; ; RESERVE DATA TABLE
 .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

 ; THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
 ; DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
 ; POSITIVE.

*CALL
 ; * MOV #PNTR, -(SP) ; ; POINTER TO LOW WORD OF BINARY NUMBER
 ; * JSR PC, @#\$DB20

10981
 10982
 10983
 10984
 10985
 10986
 10987

```

10988 ;* RETURN ;:THE FIRST ADDRESS OF ASCIZ
10989 ;:IS ON THE STACK
10990
10991
10992 054014 104406 $DB2D: SAVRE. ;:SAVE REGISTERS
10993 054016 016602 000002 MOV 2(SP),R2 ;:PICKUP THE DATA POINTER
10994 054022 012700 054174 MOV #SDECVL,R0 ;:GET ADDRESS OF "SDECVL" STRING
10995 054026 010066 000002 MOV R0,2(SP) ;:PUT ADDRESS OF ASCIZ STRING ON STACK
10996 054032 012201 MOV (R2)+,R1 ;:PICKUP THE BINARY NUMBER
10997 054034 012202 MOV (R2)+,R2
10998 054036 012737 000012 054112 MOV #10.4$ ;:SET UP TO DO 10 CONVERSIONS
10999 054044 012704 054124 MOV #STNPNR,R4 ;:ADDRESS OF TEN POWER
11000 054050 012705 054126 MOV #STNPNR+2,R5
11001 054054 005003 1$: CLR R3 ;:CLEAR PARTIAL
11002 054056 161401 2$: SUB (R4),R1 ;:SUBTRACT TEN POWER
11003 054060 005602 SBC R2
11004 054062 161502 SUB (R5),R2
11005 054064 002402 BLT 3$ ;:BR IF TEN POWER TO LARGE
11006 054066 005203 INC R3 ;:ADD 1 TO PARTIAL
11007 054070 000772 BR 2$ ;:LOOP
11008 054072 062401 3$: ADD (R4)+,R1 ;:RESTORE SUBTRACTED VALUE
11009 054074 005502 ADC R2
11010 054076 062402 ADD (R4)+,R2
11011 054100 022525 CMP (R5)+,(R5)+ ;:MOVE TO NEXT TEN POWER
11012 054102 052703 000060 BVS #0,R3 ;:CHANGE PARTIAL TO ASCII
11013 054106 110320 MOVB R3,(R0)+ ;:SAVE IT
11014 054110 005327 DEC (PC)+ ;:DONE?
11015 054112 000000 4$: .WORD 0
11016 054114 001357 BNE 1$ ;:BR IF NO
11017 054116 105020 CLRB (R0)+ ;:TERMINATOR
11018 054120 104407 RESREG ;:RESTORE REGISTERS
11019 054122 000207 RTS PC ;:RETURN
11020 054124 145000 $TNPWR: 145000 ;:1.0E09
11021 054126 035632 35632 ;:1.0E08
11022 054130 160400 160400 ;:1.0E07
11023 054132 002765 2765 ;:1.0E06
11024 054134 113200 113200 ;:1.0E05
11025 054136 000230 230 ;:1.0E04
11026 054140 041100 041100 ;:1.0E03
11027 054142 000017 17 ;:1.0E02
11028 054144 103240 103240 ;:1.0E01
11029 054146 000001 1 ;:1.0E00
11030 054150 023420 23420 ;:1.0E00
11031 054152 000000 0
11032 054154 001750 1750 ;:1.0E00
11033 054156 000000 0
11034 054160 000144 144 ;:1.0E00
11035 054162 000000 0
11036 054164 000012 12 ;:1.0E00
11037 054166 000000 0
11038 054170 000001 1 ;:1.0E00
11039 054172 000000 0
11040 054174 000014 $SDECVL: .BLKB 12. ;:RESERVE STORAGE FOR ASCIZ STRING
11041 .SBTTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
11042
11043 ;:*****

```

```

11044      ;*THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
11045      ;*LEADING NUMBERS.
11046      ;*CALL
11047      ;*      MOV      #NUMADR,-(SP)      ;;FIRST ADDRESS OF ASCII STRING
11048      ;*      JSR      PC,@#SSUPRS
11049
11050
11051      054210 010046      SSUPRS: MOV      RO,-(SP)      ;;SAVE RO
11052      054212 016600 000004      MOV      4(SP),RO      ;;PICKUP THE POINTER
11053      054216 105710      1$: TSTB      (RO)      ;;TERMINATEOR?
11054      054220 001403      BEQ      2$      ;;BR IF YES
11055      054222 122720 000060      CMPB      #'0,(RO)+      ;;IS THIS AN ASCII "0" ?
11056      054226 001773      BEQ      1$      ;;BR IF YES
11057      054230 005300      2$: DEC      RO      ;;BACKUP BY "1"
11058      054232 010037 054240      MOV      RO,3$      ;;SAVE FOR TYPING
11059      054236 104400      TYPE      ;;GO TYPE
11060      054240 000000      3$: .WORD      0      ;;ASCII POINTER GOES HERE
11061      054242 012600      MOV      (SP)+,RO      ;;RESTORE RO
11062      054244 012616      MOV      (SP)+,(SP)      ;;RESTORE THE STACK
11063      054246 000207      RTS      PC      ;;RETURN
11064      .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
11065
11066      ;*****
11067      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11068      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11069      ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11070      ;*CALL:
11071      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11072      ;*      TYPOS      ;;CALL FOR TYPEOUT
11073      ;*      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11074      ;*      .BYTE      M      ;;M=1 OR 0
11075      ;*      ;;1=TYPE LEADING ZEROS
11076      ;*      ;;0=SUPPRESS LEADING ZEROS
11077
11078      ;*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11079      ;*STYPOS OR STYPOC
11080      ;*CALL:
11081      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11082      ;*      TYPON      ;;CALL FOR TYPEOUT
11083
11084      ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11085      ;*CALL:
11086      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
11087      ;*      TYPOC      ;;CALL FOR TYPEOUT
11088
11089      054250 017646 000000      STYPOS: MOV      @2(SP),-(SP)      ;;PICKUP THE MODE
11090      054254 116637 000001 054473      MOV      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
11091      054262 112637 054475      MOV      (SP)+,SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
11092      054266 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
11093      054272 000406      BR      STYPON
11094      054274 112737 000001 054473      STYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
11095      054302 112737 000006 054475      MOV      #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
11096      054310 112737 000005 054472      STYPON: MOV      #5,SOCNT      ;;SET THE ITERATION COUNT
11097      054316 010346      MOV      R3,-(SP)      ;;SAVE R3
11098      054320 010446      MOV      R4,-(SP)      ;;SAVE R4
11099      054322 010546      MOV      R5,-(SP)      ;;SAVE R5
    
```

```

11100 054324 113704 054475      MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
11101 054330 005404          NEG      R4              ;;
11102 054332 062704 000006      ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11103 054336 110437 054474      MOVB    R4,$OMODE       ;;SAVE IT FOR USE
11104 054342 113704 054473      MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11105 054346 016605 000012      MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11106 054352 005003          CLR     R3              ;;CLEAR THE OUTPUT WORD
11107 054354 006105          1$:    ROL     R5         ;;ROTATE MSB INTO "C"
11108 054356 000404          BR     3$              ;;GO DO MSB
11109 054360 006105          2$:    ROL     R5         ;;FORM THIS DIGIT
11110 054362 006105          ROL     R5
11111 054364 006105          ROL     R5
11112 054366 010503          MOV     R5,R3
11113 054370 006103          3$:    ROL     R3         ;;GET LSB OF THIS DIGIT
11114 054372 105337 054474      DECB    $OMODE         ;;TYPE THIS DIGIT?
11115 054376 100016          BPL     7$             ;;BR IF NO
11116 054400 042703 177770      BIC     #177770,R3     ;;GET RID OF JUNK
11117 054404 001002          BNE     4$             ;;TEST FOR 0
11118 054406 005704          TST     R4             ;;SUPPRESS THIS 0?
11119 054410 001403          BEQ     5$             ;;BR IF YES
11120 054412 005204          4$:    INC     R4         ;;DON'T SUPPRESS ANYMORE 0'S
11121 054414 052703 000060      B'5     #'0,R3         ;;MAKE THIS DIGIT ASCII
11122 054420 052703 000040      5$:    BIS     #' ,R3     ;;MAKE ASCII IF NOT ALREADY
11123 054424 110337 054470      MOVB    R3,$S          ;;SAVE FOR TYPING
11124 054430 104400 054470      TYPE    8$             ;;GO TYPE THIS DIGIT
11125 054434 105337 054472      7$:    DECB    $OCNT     ;;COUNT BY 1
11126 054440 003347          BGT     2$             ;;BR IF MORE TO DO
11127 054442 002402          BLT     6$             ;;BR IF DONE
11128 054444 005204          INC     R4             ;;INSURE LAST DIGIT ISN'T A BLANK
11129 054446 000744          BR     2$              ;;GO DO THE LAST DIGIT
11130 054450 012605          6$:    MOV     (SP)+,R5   ;;RESTORE R5
11131 054452 012604          MOV     (SP)+,R4       ;;RESTORE R4
11132 054454 012603          MOV     (SP)+,R3       ;;RESTORE R3
11133 054456 016666 000002 000004  MOV     2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
11134 054464 012616          MOV     (SP)+,(SP)
11135 054466 000002          RTI                    ;;RETURN
11136 054470          8$:    .BYTE    0        ;;STORAGE FOR ASCII DIGIT
11137 054471          .BYTE    0        ;;TERMINATOR FOR TYPE ROUTINE
11138 054472          $OCNT:  .BYTE    0        ;;OCTAL DIGIT COUNTER
11139 054473          $OFILL: .BYTE    0        ;;ZERO FILL SWITCH
11140 054474 000000          $OMODE: .WORD    0        ;;NUMBER OF DIGITS TO TYPE
11141          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11142
11143          ;;*****
11144          ;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11145          ;;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11146          ;;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11147          ;;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11148          ;;REPLACED WITH SPACES.
11149          ;;CALL:
11150          ;;      MOV     NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
11151          ;;      TYPDS                    ;;GO TO THE ROUTINE
11152
11153          $TYPDS:
11154          MOV     R0,-(SP)      ;;PUSH R0 ON STACK
11155          MOV     R1,-(SP)      ;;PUSH R1 ON STACK

```

11156	054502	010246			MOV	R2,-(SP)	::PUSH R2 ON STACK
11157	054504	010345			MOV	R3,-(SP)	::PUSH R3 ON STACK
11158	054506	010546			MOV	R5,-(SP)	::PUSH R5 ON STACK
11159	054510	012746	020200		MOV	#20200,-(SP)	::SET BLANK SWITCH AND SIGN
11160	054514	016605	000020		MOV	20(SP),R5	::GET THE INPUT NUMBER
11161	054520	100004			BPL	1\$::BR IF INPUT IS POS.
11162	054522	005405			NEG	R5	::MAKE THE BINARY NUMBER POS.
11163	054524	112766	000055	000001	MOVB	#'-,1(SP)	::MAKE THE ASCII NUMBER NEG.
11164	054532	005000			CLR	R0	::ZERO THE CONSTANTS INDEX
11165	054534	012703	054712		MOV	\$DBLK,R3	::SETUP THE OUTPUT POINTER
11166	054540	112723	000040		MOVB	#',(R3)+	::SET THE FIRST CHARACTER TO A BLANK
11167	054544	005002			CLR	R2	::CLEAR THE BCD NUMBER
11168	054546	016001	054702		MOV	\$DTBL(R0),R1	::GET THE CONSTANT
11169	054552	160105			SUB	R1,R5	::FORM THIS BCD DIGIT
11170	054554	002402			BLT	4\$::BR IF DONE
11171	054556	005202			INC	R2	::INCREASE THE BCD DIGIT BY 1
11172	054560	000774			BR	3\$	
11173	054562	060105			ADD	R1,R5	::ADD BACK THE CONSTANT
11174	054564	005702			TST	R2	::CHECK IF BCD DIGIT=0
11175	054566	001002			BNE	5\$::FALL THROUGH IF 0
11176	054570	105716			TSTB	(SP)	::STILL DOING LEADING 0'S?
11177	054572	100407			BMI	7\$::BR IF YES
11178	054574	106316			ASLB	(SP)	::MSD?
11179	054576	103003			BCC	6\$::BR IF NO
11180	054600	116663	000001	177777	MOVB	1(SP),-1(R3)	::YES--SET THE SIGN
11181	054606	052702	000060		BIS	#'0,R2	::MAKE THE BCD DIGIT ASCII
11182	054612	052702	000040		BIS	#',R2	::MAKE IT A SPACE IF NOT ALREADY A DIGIT
11183	054616	110223			MOVB	R2,(R3)+	::PUT THIS CHARACTER IN THE OUTPUT BUFFER
11184	054620	005720			TST	(R0)+	::JUST INCREMENTING
11185	054622	020027	000010		CMP	R0,#10	::CHECK THE TABLE INDEX
11186	054626	002746			BLT	2\$::GO DO THE NEXT DIGIT
11187	054630	003002			BGT	8\$::GO TO EXIT
11188	054632	010502			MOV	R5,R2	::GET THE LSD
11189	054634	000764			BR	6\$::GO CHANGE TO ASCII
11190	054636	105726			TSTB	(SP)+	::WAS THE LSD THE FIRST NON-ZERO?
11191	054640	100003			BPL	9\$::BR IF NO
11192	054642	116663	177777	177776	MOVB	-1(SP),-2(R3)	::YES--SET THE SIGN FOR TYPING
11193	054650	105013			CLRB	(R3)	::SET THE TERMINATOR
11194	054652	012605			MOV	(SP)+,R5	::POP STACK INTO R5
11195	054654	012603			MOV	(SP)+,R3	::POP STACK INTO R3
11196	054656	012602			MOV	(SP)+,R2	::POP STACK INTO R2
11197	054660	012601			MOV	(SP)+,R1	::POP STACK INTO R1
11198	054662	012600			MOV	(SP)+,R0	::POP STACK INTO R0
11199	054664	104400	054712		TYPE	\$DBLK	::NOW TYPE THE NUMBER
11200	054670	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
11201	054676	012616			MOV	(SP)+,(SP)	
11202	054700	000002			RTI		::RETURN TO USER
11203	054702	023420			\$DTBL:	10000.	
11204	054704	001750				1000.	
11205	054706	000144				100.	
11206	054710	000012				10.	
11207	054712	000004			\$DBLK:	.BLKW 4	
11208					.SBTTL	ERROR HANDLER ROUTINE	
11209							
11210							
11211							

::*****
 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,

F03

```

11212      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11213      ;*AND GO TO TYPERR ON ERROR
11214      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11215      ;*SW15=1      HALT ON ERROR
11216      ;*SW13=1      INHIBIT ERROR TYPEOUTS
11217      ;*SW10=1      BELL ON ERROR
11218      ;*SW09=1      LOOP ON ERROR
11219      ;*CALL
11220      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
11221
11222      054722      ;ERROR:
11223      054722      105237      001103      7$:      INCB      $ERFLG      ;; SET THE ERROR FLAG
11224      054726      001775      BEQ      7$      ;; DON'T LET THE FLAG GO TO ZERO
11225      054730      013777      001102      124204      MOV      $TSTNM, @DISPLAY      ;; DISPLAY TEST NUMBER AND ERROR FLAG
11226      054736      032777      002000      124174      BIT      @BIT10, @SWR      ;; BELL ON ERROR?
11227      054744      001402      BEQ      1$      ;; NO - SKIP
11228      054746      104400      001310      TYPE      $BELL      ;; RING BELL
11229      054752      005237      001112      1$:      INC      $ERTTL      ;; COUNT THE NUMBER OF ERRORS
11230      054756      011637      001116      MOV      (SP), $ERRPC      ;; GET ADDRESS OF ERROR INSTRUCTION
11231      054762      162737      000002      001116      SUB      #2, $ERRPC
11232      054770      117737      124122      001114      MOVB    @ERRPC, $ITEMB      ;; STRIP AND SAVE THE ERROR ITEM CODE
11233      054776      032777      020000      124134      BIT      @BIT13, @SWR      ;; SKIP TYPEOUT IF SET
11234      055004      001004      BNE      20$      ;; SKIP TYPEOUTS
11235      055006      004737      041222      JSR      PC, TYPERR      ;; GO TO USER ERROR ROUTINE
11236      055012      104400      001315      TYPE      , $CRLF
11237      055016
11238      055016      122737      000001      001340      20$:      CMPB    @APTENV, $ENV      ;; RUNNING IN APT MODE
11239      055024      001007      BNE      2$      ;; NO SKIP APT ERROR REPORT
11240      055026      113737      001114      055040      MOVB    $ITEMB, 21$      ;; SET ITEM NUMBER AS ERROR NUMBER
11241      055034      004737      055702      JSR      PC, $ATY4      ;; REPORT FATAL ERROR TO APT
11242      055040      000      21$:      .BYTE  0
11243      055041      000      .BYTE  0
11244      055042      000777      22$:      BR      22$      ;; APT ERROR LOOP
11245      055044      005777      124070      2$:      TST    @SWR      ;; HALT ON ERROR
11246      055050      100001      BPL      3$      ;; SKIP IF CONTINUE
11247      055052      000000      HALT      ;; HALT ON ERROR!
11248      055054      032777      001000      124056      3$:      BIT      @BIT09, @SWR      ;; LOOP ON ERROR SWITCH SET?
11249      055062      001402      BEQ      4$      ;; BR IF NO
11250      055064      013716      001110      MOV      $LPERR, (SP)      ;; FUDGE RETURN FOR LOOPING
11251      055070      005737      001306      4$:      TST    $ESCAPE      ;; CHECK FOR AN ESCAPE ADDRESS
11252      055074      001402      BEQ      5$      ;; BR IF NONE
11253      055076      013716      001306      MOV      $ESCAPE, (SP)      ;; FUDGE RETURN ADDRESS FOR ESCAPE
11254      055102
11255      055102      022737      023546      000042      5$:      CMP      @SENDAD, @#42      ;; ACT-11 AUTO-ACCEPT?
11256      055110      001001      BNE      6$      ;; BRANCH IF NO
11257      055112      000000      HALT      ;; YES
11258      055114
11259      055114      000002      6$:      RTI      ;; RETURN
11260      .SBTTL      TTY INPUT ROUTINE
11261
11262      ;*****
11263      .ENABL      LSB
11264
11265      .DSABL      LSB
11266
11267

```



```

11268                                     ;:*****
11269                                     ;:THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11270                                     ;:CALL:
11271                                     ;:RDCHR                                     ;:INPUT A SINGLE CHARACTER FROM THE TTY
11272                                     ;:RETURN HERE                               ;:CHARACTER IS ON THE STACK
11273                                     ;:                                           ;:WITH PARITY BIT STRIPPED OFF
11274
11275
11276 055116 011646 $RDCHR: MOV (SP),-(SP) ;:PUSH DOWN THE PC
11277 055120 016666 MOV 4(SP),2(SP) ;:SAVE THE PS
11278 055126 105777 124012 1$: TSTB 2$TKS ;:WAIT FOR
11279 055132 100375 BPL 1$ ;:A CHARACTER
11280 055134 117766 124006 000004 MOVB 2$TKB,4(SP) ;:READ THE TTY
11281 055142 042766 177600 000004 BIC #1C<177>,4(SP) ;:GET RID OF JUNK IF ANY
11282 055150 026627 000004 000023 CMP 4(SP),#23 ;:IS IT A CONTROL-S?
11283 055156 001013 BNE 3$ ;:BRANCH IF NO
11284 055160 105777 123760 2$: TSTB 2$TKS ;:WAIT FOR A CHARACTER
11285 055164 100375 BPL 2$ ;:LOOP UNTIL ITS THERE
11286 055166 117746 123754 MOVB 2$TKB,-(SP) ;:GET CHARACTER
11287 055172 042716 177600 BIC #1C177,(SP) ;:MAKE IT 7-BIT ASCII
11288 055176 022627 000021 CMP (SP)+,#21 ;:IS IT A CONTROL-Q?
11289 055202 001366 BNE 2$ ;:IF NOT DISCARD IT
11290 055204 000750 BR 1$ ;:YES, RESUME
11291 055206 026627 000004 000140 3$: CMP 4(SP),#140 ;:IS IT UPPER CASE?
11292 055214 002407 BLT 4$ ;:BRANCH IF YES
11293 055216 026627 000004 000175 CMP 4(SP),#175 ;:IS IT A SPECIAL CHAR?
11294 055224 003003 BGT 4$ ;:BRANCH IF YES
11295 055226 042766 000040 000004 BIC #40,4(SP) ;:MAKE IT UPPER CASE
11296 055234 000002 4$: RTI ;:GO BACK TO USER
11297 055236 052536 005015 000 $CNTLU: .ASCIZ /1U<15><12> ;:CONTROL "U"
11298 055243 136 006507 000012 $CNTLG: .ASCIZ /1G<15><12> ;:CONTROL "G"
11299 055250 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
11300 055256 020075 000
11301 055261 040 047040 053505 $MNEW: .ASCIZ / NEW = /
11302 055266 036440 000040
11303
11304
11305
11306 .SBTTL POWER DOWN AND UP ROUTINES
11307
11308 ;POWER DOWN ROUTINE
11309 055272 012737 055304 000024 $PWRDN: MOV #PWRUP,PWRVEC ;SET VECTOR FOR POWER UP
11310 055300 000000 HALT ;HANG UP
11311 055302 000776 BR .-2
11312
11313 ;POWER UP ROUTINE
11314 055304 005037 055356 $PWRUP: CLR $PWRCT ;WAIT LOOP FOR TTY TO COME UP
11315 055310 005237 055356 4$: INC $PWRCT
11316 055314 001375 BNE 4$
11317 055316 012737 055272 000024 MOV #PWRDN,PWRVEC ;SET VECTOR FOR POWER DOWN
11318 055324 012737 000340 000026 MOV #PR7,PWRVEC+2 ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11319 055332 012737 000340 000036 MOV #PR7,TRAPVEC+2
11320 055340 012706 001100 MOV #STACK,SP ;RE-INITIALIZE THE STACK
11321 055344 104400 055360 TYPE ,PWRMSG ;TYPE "POWER FAILED"
11322 055350 000005 RESET ;CLEAR THE UNIBUS
11323 055352 000177 123530 JMP 2$LPADR ;RESTART THE CURRENT TEST

```

11324 055356 000000
 11325 055360 005015 047520 042527
 11326 055366 020122 040506 046111
 11327 055374 042105 005015 000
 11328
 11329
 11330
 11331
 11332
 11333
 11334
 11335
 11336 055402 105737 001103
 11337 055406 001406
 11338 055410 032777 001000 123522
 11339 055416 001402
 11340 055420 013716 001110
 11341 055424 000002
 11342
 11343
 11344
 11345
 11346
 11347
 11348
 11349
 11350
 11351
 11352
 11353
 11354
 11355
 11356
 11357
 11358 055426
 11359 055426 005737 001304
 11360 055432 001404
 11361 055434 032777 040000 123476
 11362 055442 001101
 11363
 11364 055444 000416
 11365
 11366 055446 013746 000004
 11367 055452 012737 055472 000004
 11368 055460 005737 177060
 11369 055464 012637 000004
 11370 055470 000450
 11371 055472 022626
 11372 055474 012637 000004
 11373 055500 000413
 11374 055502
 11375 055502 105737 001103
 11376 055506 001421
 11377 055510 123737 001115 001103
 11378 055516 101015
 11379 055520 032777 001000 123412

```

$PWACT: .WORD 0
$PWRMSG: .ASCIZ <15><12>/POWER FAILED/<15><12>

.EVEN

*****
* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
* CALLED BY "SCOPE"
*****
SCOPE1: TSTB $ERFLG ;SEE IF AN ERROR HAS OCCURRED
        BEQ 6$ ;BR IF NOT
        BIT #BIT9,$SWR ;SEE IF LOOP ON ERROR DESIRED
        BEQ 6$ ;BR IF NOT
        MOV $LPERR,(SP) ;SET ERROR LOOP ADDRESS ON STACK
6$: RTI ;RETURN

.SBttl SCOPE HANDLER ROUTINE

*****
* THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
* AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
* AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
* THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
* SW14=1 LOOP ON TEST
* SW11=1 INHIBIT ITERATIONS
* SW09=1 LOOP ON ERROR
* CALL
* SCOPE ;;SCOPE=IOT

$SCOPE:
        TST $TIMES ;CHECK CURRENT ITERATION NUMBER
        BEQ $XTSTR ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
1$: BIT #BIT14,$SWR ;LOOP ON PRESENT TEST?
        BNE $OVER ;YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ;IF RUNNING ON THE "XOR" TESTER CHANGE
        MOV 2*$ERRVEC,-(SP) ;THIS INSTRUCTION TO A "NOP" (NOP=240)
        MOV #55,2*$ERRVEC ;SAVE THE CONTENTS OF THE ERROR VECTOR
        TST 2*$177060 ;SET FOR TIMEOUT
        BR $SVLAD ;TIME OUT ON XOR?
        BR $SVLAD ;RESTORE THE ERROR VECTOR
        CMP (SP)+,(SP)+ ;GO TO THE NEXT TEST
        MOV (SP)+,2*$ERRVEC ;CLEAR THE STACK AFTER A TIME OUT
        BR 7$ ;RESTORE THE ERROR VECTOR
        ;LOOP ON THE PRESENT TEST

6$; *****END OF CODE FOR THE XOR TESTER*****
2$: TSTB $ERFLG ;HAS AN ERROR OCCURRED?
        BEQ 3$ ;BR IF NO
        CMPB $ERMAX,$ERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?
        BHI 3$ ;BR IF NO
        BIT #BIT09,$SWR ;LOOP ON ERROR?

```

```

11380 055526 001404          BEQ      4$          ;;BR IF NO
11381 055530 013737 001110 001106 7$:  MOV     $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
11382 055536 000443          BR       $OVER
11383 055540 105037 001103          4$:  CLRB   $ERFLG       ;;ZERO THE ERROR FLAG
11384 055544 005037 001304          CLR     $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11385 055550 000412          BR       1$          ;;ESCAPE TO THE NEXT TEST
11386 055552 032777 004000 123360 3$:  BIT    #BIT11,$SWR  ;;INHIBIT ITERATIONS?
11387 055560 001006          BNE     1$          ;;BR IF YES
11388 055562 005237 001104          INC     $ICNT       ;;INCREMENT ITERATION COUNT
11389 055566 023737 001304 001104  CMP     $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
11390 055574 002024          BGE     $OVER       ;;BR IF MORE ITERATION REQUIRED
11391 055576 012737 000001 001104 1$:  MOV     #1,$ICNT    ;;REINITIALIZE THE ITERATION COUNTER
11392 055604 013737 055662 001304  MOV     $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
11393 055612 105237 001102          $SVLAD: INCB   $STNM      ;;COUNT TEST NUMBERS
11394 055616 113737 001102 001324  MOVVB  $STNM,$STN   ;;SET TEST NUMBER IN APT MAILBOX
11395 055624 011637 001106          MOV     (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
11396 055630 011637 001110          MOV     (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
11397 055634 005037 001306          CLR     $ESCAPE    ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11398 055640 112737 000001 001115  MOVVB  #1,$ERMAX   ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11399 055646 013777 001102 123266 $OVER: MOV     $STNM,$DISPLAY ;;DISPLAY TEST NUMBER
11400 055654 013716 001106          MOV     $LPADR,(SP) ;;FUDGE RETURN ADDRESS
11401 055660 000002          RTI
11402 055662 003720          $MXCNT: 2000.     ;;FIXES PS
11403          .SBTTL APT COMMUNICATIONS ROUTINE ;;MAX. NUMBER OF ITERATIONS
11404
11405          ;;*****
11406 055664 112737 000001 056130 $ATY1: MOVVB  #1,$FFLG  ;;TO REPORT FATAL ERROR
11407 055672 112737 000001 056126 $ATY3: MOVVB  #1,$MFLG  ;;TO TYPE A MESSAGE
11408 055700 000403          BR       $ATYC
11409 055702 112737 000001 056130 $ATY4: MOVVB  #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
11410 055710          $ATYC:
11411 055710 010046          MOV     R0,-(SP)    ;;PUSH R0 ON STACK
11412 055712 010146          MOV     R1,-(SP)    ;;PUSH R1 ON STACK
11413 055714 105737 056126          TSTB   $MFLG       ;;SHOULD TYPE A MESSAGE?
11414 055720 001450          BEQ     5$          ;;IF NOT: BR
11415 055722 122737 000001 001340  CMPB   #APTENV,$ENV ;;OPERATING UNDER APT?
11416 055730 001031          BNE     3$          ;;IF NOT: BR
11417 055732 132737 000100 001341  BITB   #APTSPool,$ENVM ;;SHOULD SPOOL MESSAGES?
11418 055740 001425          BEQ     3$          ;;IF NOT: BR
11419 055742 017600 000004          MOV     @4(SP),R0   ;;GET MESSAGE ADDR.
11420 055746 062766 000002 000004  ADD     #2,4(SP)     ;;BUMP RETURN ADDR.
11421 055754 005737 001320          1$:  TST    $MSGTYPE    ;;SEE IF DONE W/ LAST XMISSION?
11422 055760 001375          BNE     1$          ;;IF NOT: WAIT
11423 055762 010037 001334          MOV     R0,$MSGAD  ;;PUT ADDR IN MAILBOX
11424 055766 105720          2$:  TSTB   (R0)+       ;;FIND END OF MESSAGE
11425 055770 001376          BNE     2$
11426 055772 163700 001334          SUB     $MSGAD,R0   ;;SUB START OF MESSAGE
11427 055776 006200          ASR    R0           ;;GET MESSAGE LNGTH IN WORDS
11428 056000 010037 001336          MOV     R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
11429 056004 012737 000004 001320  MOV     #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
11430 056012 000413          BR       5$
11431 056014 017637 000004 056040 3$:  MOV     @4(SP),4$   ;;PUT MSG ADDR IN JSR LINKAGE
11432 056022 062766 000002 000004  ADD     #2,4(SP)     ;;BUMP RETURN ADDRESS
11433 056030 013746 177776          MOV     177776,-(SP) ;;PUSH 177776 ON STACK
11434 056034 004737 053174          JSR    PC,$TYPE    ;;CALL TYPE MACRO
11435 056040 000000          4$:  .WORD  0

```

```

11436 056042          5$:
11437 056042 105737 056130 10$: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
11438 056046 001416          BEQ 12$ ;; IF NOT: BR
11439 056050 005737 001340 TST $ENV ;; RUNNING UNDER APT?
11440 056054 001413          BEQ 12$ ;; IF NOT: BR
11441 056056 005737 001320 11$: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
11442 056062 001375          BNE 11$ ;; IF NOT: WAIT
11443 056064 017637 000004 001322 MOV @4(SP), $FATAL ;; GET ERROR #
11444 056072 062766 000002 000004 ADD #2, 4(SP) ;; BUMP RETURN ADDR.
11445 056100 005237 001320 INC $MSGTYPE ;; TELL APT TO TAKE ERROR
11446 056104 105037 056130 12$: CLRB $FFLG ;; CLEAR FATAL FLAG
11447 056110 105037 056127 CLRB $LFLG ;; CLEAR LOG FLAG
11448 056114 105037 056126 CLRB $MFLG ;; CLEAR MESSAGE FLAG
11449 056120 012601          MOV (SP)+, R1 ;; POP STACK INTO R1
11450 056122 012600          MOV (SP)+, R0 ;; POP STACK INTO R0
11451 056124 000207          RTS PC ;; RETURN
11452 056126 000          $MFLG: .BYTE 0 ;; MESSG. FLAG
11453 056127 000          $LFLG: .BYTE 0 ;; LOG FLAG
11454 056130 000          $FFLG: .BYTE 0 ;; FATAL FLAG
11455 056132          .EVEN
11456 000200 APTSIZE=200
11457 000001 APTENV=001
11458 000100 APTSPool=100
11459 000040 APTCSUP=040
11460 .SBTTL ROUTINE TO SIZE MEMORY
11461
11462 *****
11463 *CALL:
11464 * JSR PC, $SIZE
11465 * RETURN
11466 * $LSTAD WILL CONTAIN:
11467 * WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11468 * WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11469 * $LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11470 * $KT11 IS THE MEMORY MANAGEMENT KEY
11471 * BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11472 * MUST BE SETUP BEFORE THE CALL
11473 * BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11474 * DETERMINED BY ROUTINE
11475
11476 056132 010046 $SIZE: MOV R0, -(SP) ;; SAVE R0 ON THE STACK
11477 056134 010146 MOV R1, -(SP) ;; SAVE R1 ON THE STACK
11478 056136 010246 MOV R2, -(SP) ;; SAVE R2 ON THE STACK
11479 056140 010346 MOV R3, -(SP) ;; SAVE R3 ON THE STACK
11480 056142 013746 000004 MOV @#ERRVEC, -(SP) ;; SAVE PRESENT ERROR VECTOR PS & PC
11481 056146 013746 000006 MOV @#ERRVEC+2, -(SP)
11482 056152 010600 MOV SP, R0 ;; SAVE THE STACK POINTER
11483 ;; SET THE ERRVEC PS TO THE PRESENT PS
11484 056154 013746 000034 MOV @#TRAPVEC, -(SP) ;; SAVE CURRENT TRAP VECTOR
11485 056160 012737 056170 000034 MOV #64$, @#TRAPVEC ;; SETUP NEW TRAP VECTOT
11486 056166 104400 TRAP ;; PUSH OLD PSW AND PC ON STACK
11487 056170 013637 000002 000006 64$: MOV 2(SP), @#ERRVEC+2 ;; SAVE PSW IN @#ERRVEC+2
11488 056176 012716 056204 MOV #65$, (SP) ;; REPLACE OLD PC WITH NEW
11489 056202 000002 RTI ;; RESTORE PSW
11490 056204 012637 000034 65$: MOV (SP)+, @#TRAPVEC ;; RESTORE OLD TRAP VECTOR
11491 056210 012701 003776 MOV #3776, R1 ;; SETUP ADDRESS

```

```

11492 056214 105727          TSTB      (PC)+          ;;USE MEMORY MANAGEMENT?
11493 056216 000200          $KT11: .WORD 200          ;;SET TO USE MEMORY MANAGEMENT
11494 056220 100062          BPL      SCORE          ;;BR IF NO
11495 056222 012737 056360 000004    MOV      # $KTNEX, @ $ERRVEC ;;SET FOR TIMEOUT
11496 056230 005737 177572    TST      @ $SR0          ;;KT11 ARE YOU THERE?
11497 056234 052737 100000 056216  BIS      #100000, $KT11    ;;YES--SET KT11 KEY
11498 056242 005046          CLR      -(SP)          ;;INITIALIZE FOR "PAR" LOADING
11499 056244 012702 172340    MOV      #KIPAR0, R2     ;;ADDRESS OF FIRST "PAR"
11500 056250 012703 000010    MOV      #108, R3        ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
11501 056254 012762 077406 177740 1$:  MOV      #77406, -40(R2)  ;;PDR = 4K, UP, READ/WRITE
11502 056262 011622          MOV      (SP), (R2)+    ;;LOAD "PAR"
11503 056264 062716 000200    ADD      #200, (SP)     ;;UPDATE FOR NEXT "PAR"
11504 056270 077307          SOB      R3, 1$        ;;LOOP UNTIL ALL EIGHT ARE LOADED
11505 056272 012742 177600    MOV      #177600, -(R2)  ;;SETUP KIPAR7 FOR I/O
11506 056276 005042          CLR      -(R2)          ;;SETUP KIPAR6 FOR TESTING
11507 056300 012737 056316 000004    MOV      #2$, @ $ERRVEC  ;;CATCH TIMEOUT IF NO SR3
11508 056306 012737 000020 172516  MOV      #20, @ $SR3    ;;ENABLE 22 BIT MODE
11509 056314 000401          BR       3$            ;;THIS PDP-11 HAS A SR3 REGISTER
11510 056316 022626          2$:  CMP      (SP)+, (SP)+ ;;CLEAN OFF THE STACK--NO SR3
11511 056320 005237 177572          3$:  INC      @ $SR0      ;;TURN ON MEMORY MANAGEMENT
11512 056324 012737 056350 000004    MOV      # $KTOUT, @ $ERRVEC ;;SET FOR TIME OUT
11513 056332 005737 143776          4$:  TST      @ #143776   ;;TRAP ON NON-EX-MEM
11514 056336 062712 000040    ADD      #40, (R2)      ;;MAKE A 1K STEP
11515 056342 023712 172356    CMP      @ #KIPAR7, (R2) ;;LAST ONE?
11516 056346 101371          BHI     4$            ;;NO--TRY IT
11517 056350 011202          $KTOUT: MOV      (R2), R2   ;;GET LAST BANK+1
11518 056352 005037 177572          CLR      @ $SR0        ;;TURN OFF MEMORY MANAGEMENT
11519 056356 000421          BR       $SIZEX
11520 056360 042737 100000 056216  $KTNEX: BIC      #100000, $KT11 ;;KT11 NON-EXISTENT
11521 056366 012737 056416 000004  SCORE:  MOV      # $SCROUT, @ $ERRVEC ;;SET FOR TIMEOUT
11522 056374 005002          CLR      R2            ;;SET UP BANK
11523 056376 062701 004000          1$:  ADD      #4000, R1    ;;INCREMENT BY 1K
11524 056402 062702 000040    ADD      #40, R2        ;;1K STEP
11525 056406 005711          TST      (R1)          ;;TRAP ON TIME OUT
11526 056410 022701 177776    CMP      #177776, R1    ;;LAST ONE
11527 056414 001370          BNE     1$            ;;NO--TRY AGAIN
11528 056416 162701 004000          $SCROUT: SUB     #4000, R1 ;;DROP BACK
11529 056422 162702 000040          $SIZEX: SUB     #40, R2  ;;RESTORE THE STACK
11530 056426 010006          MOV      R0, SP        ;;RESTORE ERROR VECTOR
11531 056430 012637 000006    MOV      (SP)+, @ $ERRVEC+2 ;;RESTORE ERROR VECTOR
11532 056434 012637 000004    MOV      (SP)+, @ $ERRVEC
11533 056440 010137 056462          MOV      R1, $LSTAD    ;;LAST ADDRESS
11534 056444 010237 056464          MOV      R2, $LSTBK    ;;LAST BANK
11535 056450 012603          MOV      (SP)+, R3     ;;RESTORE R3
11536 056452 012602          MOV      (SP)+, R2     ;;RESTORE R2
11537 056454 012601          MOV      (SP)+, R1     ;;RESTORE R1
11538 056456 012600          MOV      (SP)+, R0     ;;RESTORE R0
11539 056460 000207          RTS      PC
11540 056462 000000          $LSTAD: .WORD 0        ;;CONTAINS THE LAST ADDRESS
11541 056464 000000          $LSTBK: .WORD 0        ;;CONTAINS THE LAST BANK
11542          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11543          ;;*****
11544          ;;SAVE R0-R5
11545          ;;CALL:
11546          ;;
11547          ;; SAVREG

```

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

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

;;*RESTORE RO-R5
*CALL:
* RESREG

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

.SBTTL TRAP DECODER

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

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

.SBTTL TRAP TABLE

11548
11549
11550
11551
11552
11553
11554
11555
11556
11557
11558
11559 056466
11560 056466 010046
11561 056470 010146
11562 056472 010246
11563 056474 010346
11564 056476 010446
11565 056500 010546
11566 056502 016646 000022
11567 056506 016646 000022
11568 056512 016646 000022
11569 056516 016646 000022
11570 056522 000002
11571
11572
11573
11574
11575 056524
11576 056524 012666 000022
11577 056530 012666 000022
11578 056534 012666 000022
11579 056540 012666 000022
11580 056544 012605
11581 056546 012604
11582 056550 012603
11583 056552 012602
11584 056554 012601
11585 056556 012600
11586 056560 000002
11587
11588
11589
11590
11591
11592
11593
11594
11595 056562 010046
11596 056564 016600 000002
11597 056570 005740
11598 056572 111000
11599 056574 006300
11600 056576 016000 056604
11601 056502 000200
11602
11603

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

: ROUTINE
:-----
: \$TRPAD:

\$TYPE	:::CALL=TYPE	TRAP+0(104400)	TTY TYPEOUT ROUTINE
\$TYPOC	:::CALL=TYPOC	TRAP+1(104401)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS	:::CALL=TYPOS	TRAP+2(104402)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON	:::CALL=TYPON	TRAP+3(104403)	TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPDS	:::CALL=TYPDS	TRAP+4(104404)	TYPE DECIMAL NUMBER (WITH SIGN)
\$RDCHR	:::CALL=RDCHR	TRAP+5(104405)	TTY TYPEIN CHARACTER ROUTINE
\$SAVREG	:::CALL=SAVREG	TRAP+6(104406)	SAVE R0-R5 ROUTINE
\$RESREG	:::CALL=RESREG	TRAP+7(104407)	RESTORE R0-R5 ROUTINE
\$SCOPE1	:::CALL=SCOPE1	TRAP+10(104410)	INTERNAL LOOP ON ERROR ROUTINE

11604					
11605					
11606					
11607					
11608					
11609					
11610	056604				
11611	056604	053174			
11612	056606	054274			
11613	056610	054250			
11614	056612	054310			
11615	056614	054476			
11616					
11617					
11618	056616	055116			
11619	056620	056466			
11620	056622	056524			
11621	056624	055402			
11622					
11623	056626	020040	020040	042524	TSTMSG: .ASCIZ / TEST /
11624	056634	052123	000040		
11625	056640	025052	020040	000	AS2SP2: .ASCIZ /** /
11626	056645	125	044516	052502	EM1: .ASCIZ /UNIBUS PARITY ERROR/
11627	056652	020123	040520	044522	
11628	056660	054524	042440	051122	
11629	056666	051117	000		
11630	056671	116	047117	042455	EM2: .ASCIZ /NON-EXISTANT MEMORY/
11631	056676	044530	052123	047101	
11632	056704	020124	042515	047515	
11633	056712	054522	000		
11634	056715	116	047117	042455	EM3: .ASCIZ /NON-EXISTANT DRIVE/
11635	056722	044530	052123	047101	
11636	056730	020124	051104	053111	
11637	056736	000105			
11638	056740	047125	052111	043040	EM4: .ASCIZ /UNIT FIELD ERROR/
11639	056746	042511	042114	042440	
11640	056754	051122	051117	000	
11641	056761	123	041125	054523	EM5: .ASCIZ /SUBSYS TIMEOUT/
11642	056766	020123	044524	042515	
11643	056774	052517	000124		
11644	057000	042523	041522	047117	EM6: .ASCIZ /SERCON PARITY ERROR/
11645	057006	050040	051101	052111	
11646	057014	020131	051105	047522	
11647	057022	000122			
11648	057024	051104	053111	020105	EM7: .ASCIZ /DRIVE DETECTED PARITY ERROR/
11649	057032	042504	042524	052103	
11650	057040	042105	050040	051101	
11651	057046	052111	020131	051105	
11652	057054	047522	000122		
11653	057060	041501	046040	053517	EM10: .ASCIZ /AC LOW/
11654	057066	000			
11655	057067	123	042520	042105	EM11: .ASCIZ /SPEED LOSS/
11656	057074	046040	051517	000123	
11657	057102	046111	042514	040507	EM12: .ASCIZ /ILLEGAL FUNCTION/
11658	057110	020114	052506	041516	
11659	057116	044524	047117	000	

11660	057123	120	047522	051107	EM13:	.ASCIZ	/PROGRAMMING ERROR/
11661	057130	046501	044515	043516			
11662	057136	042440	051122	051117			
11663	057144	000					
11664	057145	116	047117	042455	EM14:	.ASCIZ	/NON-EXISTANT FUNCTION/
11665	057152	044530	052123	047101			
11666	057160	020124	052506	041516			
11667	057166	044524	047117	000			
11668	057173	104	044522	042526	EM15:	.ASCIZ	/DRIVE TYPE ERROR/
11669	057200	052040	050131	020105			
11670	057206	051105	047522	000122			
11671	057214	047506	046522	052101	EM16:	.ASCIZ	/FORMAT ERROR/
11672	057222	042440	051122	051117			
11673	057230	000					
11674	057231	127	044522	042524	EM17:	.ASCIZ	/WRITE LOCK ERROR/
11675	057236	046040	041517	020113			
11676	057244	051105	047522	000122			
11677	057252	051104	053111	020105	EM20:	.ASCIZ	/DRIVE UNSAFE/
11678	057260	047125	040523	042506			
11679	057266	000					
11680	057267	123	042505	020113	EM21:	.ASCIZ	/SEEK INCOMPLETE/
11681	057274	047111	047503	050115			
11682	057302	042514	042524	000			
11683	057307	103	046131	047111	EM22:	.ASCIZ	/CYLINDER OVERFLOW/
11684	057314	042504	020122	053117			
11685	057322	051105	046106	053517			
11686	057330	000					
11687	057331	111	046114	043505	EM23:	.ASCIZ	/ILLEGAL CYLINDER ADDRESS/
11688	057336	046101	041440	046131			
11689	057344	047111	042504	020122			
11690	057352	042101	051104	051505			
11691	057360	000123					
11692	057362	051104	053111	020105	EM24:	.ASCIZ	/DRIVE OFF TRACK/
11693	057370	043117	020106	051124			
11694	057376	041501	000113				
11695	057402	051104	053111	020105	EM25:	.ASCIZ	/DRIVE TIMING ERROR/
11696	057410	044524	044515	043516			
11697	057416	042440	051122	051117			
11698	057424	000					
11699	057425	104	052101	020101	EM26:	.ASCIZ	/DATA LATE/
11700	057432	040514	042524	000			
11701	057437	103	047117	051124	EM27:	.ASCIZ	/CONTROLLER TIMEOUT/
11702	057444	046117	042514	020122			
11703	057452	044524	042515	052517			
11704	057460	000124					
11705	057462	050117	051105	052101	EM30:	.ASCIZ	/OPERATION INCOMPLETE/
11706	057470	047511	020116	047111			
11707	057476	047503	050115	042514			
11708	057504	042524	000				
11709	057507	110	040505	042504	EM31:	.ASCIZ	/HEADER VRC ERROR/
11710	057514	020122	051126	020103			
11711	057522	051105	047522	000122			
11712	057530	040504	040524	041440	EM32:	.ASCIZ	/DATA CHECK ERROR/
11713	057536	042510	045503	042440			
11714	057544	051122	051117	000			
11715	057551	127	044522	042524	EM33:	.ASCIZ	/WRITE CHECK ERROR/

11716	057556	041440	042510	045503			
11717	057564	042440	051122	051117			
11718	057572	000					
11719	057573	104	052101	020101	EM34:	.ASCIZ	'DATA MISCOMPARE/'
11720	057E00	044515	041523	046517			
11721	057606	040520	042522	000			
11722	057613	116	020117	051104	EM35:	.ASCIZ	'/NO DRIVE RESPONSE-UFE & NXD/'
11723	057620	053111	020105	042522			
11724	057626	050123	047117	042523			
11725	057634	052455	042506	023040			
11726	057642	047040	042130	000			
11727	057647	104	044522	042526	EM36:	.ASCIZ	'/DRIVE ERROR WILL NOT CLEAR/'
11728	057654	042440	051122	051117			
11729	057662	053440	046111	020114			
11730	057670	047516	020124	046103			
11731	057676	040505	000122				
11732	057702	051104	053111	020105	EM37:	.ASCIZ	'/DRIVE STATUS CHANGE WILL NOT CLEAR/'
11733	057710	052123	052101	051525			
11734	057716	041440	040510	043516			
11735	057724	020105	044527	046114			
11736	057732	047040	052117	041440			
11737	057740	042514	051101	000			
11738	057745	101	052124	047047	EM40:	.ASCIZ	'/ATT'N BUT NO STATUS CHANGE OR FAULT/'
11739	057752	041040	052125	047040			
11740	057760	020117	052123	052101			
11741	057766	051525	041440	040510			
11742	057774	043516	020105	051117			
11743	060002	043040	052501	052114			
11744	060010	000					
11745	060011	101	052124	047047	EM41:	.ASCIZ	'/ATT'N BUT DRIVE NOT AVAILABLE/'
11746	060016	041040	052125	042040			
11747	060024	044522	042526	047040			
11748	060032	052117	040440	040526			
11749	060040	046111	041101	042514			
11750	060046	000					
11751	060047	101	052124	047047	EM42:	.ASCIZ	'/ATT'N WHEN NOT EXPECTED/'
11752	060054	053440	042510	020116			
11753	060062	047516	020124	054105			
11754	060070	042520	052103	042105			
11755	060076	000					
11756	060077	105	051122	051117	EM43:	.ASCIZ	'/ERROR GATHERING DRIVE STATUS/'
11757	060104	043440	052101	042510			
11758	060112	044522	043516	042040			
11759	060120	044522	042526	051440			
11760	060126	040524	052524	000123			
11761	060134	052515	052114	050111	EM52:	.ASCIZ	'/MULTIPLE DRIVE SELECT/'
11762	060142	042514	042040	044522			
11763	060150	042526	051440	046105			
11764	060156	041505	000124				
11765	060162	042510	042101	051105	EM53:	.ASCIZ	'/HEADER COMPARE ERROR/'
11766	060170	041440	046517	040520			
11767	060176	042522	042440	051122			
11768	060204	051117	000				
11769	060207	123	041125	054523	EM56:	.ASCIZ	'/SUBSYS TIMEOUT/'
11770	060214	020123	044524	042515			
11771	060222	052517	000124				

11772	060226	051105	047522	020122	EM60:	.ASCIZ	/ERROR IN RECAL FOR RECOVERY/
11773	060234	047111	051040	041505			
11774	060242	046101	043040	051117			
11775	060250	051040	041505	053117			
11776	060256	051105	000131				
11777	060262	051120	043517	040522	EM61:	.ASCIZ	/PROGRAM ABORTING FATAL ERROR IN RETRY/
11778	060270	020115	041101	051117			
11779	060276	044524	043516	043040			
11780	060304	052101	046101	042440			
11781	060312	051122	051117	044440			
11782	060320	020116	042522	051124			
11783	060326	000131					
11784	060330	054503	044514	042116	EM62:	.ASCIZ	/CYLINDER MISCOMPARE/
11785	060336	051105	046440	051511			
11786	060344	047503	050115	051101			
11787	060352	000105					
11788	060354	046103	040505	020122	EM63:	.ASCIZ	/CLEAR CONTROLLER DID NOT CLEAR ERROR/
11789	060362	047503	052116	047522			
11790	060370	046114	051105	042040			
11791	060376	042111	047040	052117			
11792	060404	041440	042514	051101			
11793	060412	042440	051122	051117			
11794	060420	000					
11795	060421	116	020117	052101	EM64:	.ASCIZ	/NO ATT'N IN ATT'N SUMMARY REGISTER/
11796	060426	023524	020116	047111			
11797	060434	040440	052124	047047			
11798	060442	051440	046525	040515			
11799	060450	054522	051040	043505			
11800	060456	051511	042524	000122			
11801	060464	047125	047523	044514	EM65:	.ASCIZ	/UNSOLICITED ATTENTION/
11802	060472	044503	042524	020104			
11803	060500	052101	042524	052116			
11804	060506	047511	000116				
11805	060512	047125	054105	042520	EM66:	.ASCIZ	/UNEXPECTED DATA TYPE ERROR/
11806	060520	052103	042105	042040			
11807	060526	052101	020101	054524			
11808	060534	042520	042440	051122			
11809	060542	051117	000				
11810	060545	101	052124	047047	EM67:	.ASCIZ	/ATT'N DID NOT RESET WITH CLEAR/
11811	060552	042040	042111	047040			
11812	060560	052117	051040	051505			
11813	060566	052105	053440	052111			
11814	060574	020110	046103	040505			
11815	060602	000122					
11816	060604	052523	051502	051531	EM70:	.ASCIZ	/SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
11817	060612	041440	042514	051101			
11818	060620	042040	042111	047040			
11819	060626	052117	041440	042514			
11820	060634	051101	042040	044522			
11821	060642	042526	040440	052124			
11822	060650	047047	000				
11823	060653	104	052101	020101	EM71:	.ASCIZ	/DATA LATE WHEN UNLOADING HEADER/
11824	060660	040514	042524	053440			
11825	060666	042510	020116	047125			
11826	060674	047514	042101	047111			
11827	060702	020107	042510	042101			

11828	060710	051105	000			
11829	060713	103	047117	051124	EM72: .ASCIZ	/CONTROLLER ERROR WHEN DRIVER SERVICING/
11830	060720	046117	042514	020122		
11831	060726	051105	047522	020122		
11832	060734	044127	047105	042040		
11833	060742	044522	042526	020122		
11834	060750	042523	053122	041511		
11835	060756	047111	000107			
11836	060762	051104	053111	020105	EM73: .ASCIZ	/DRIVE DETECTED PARITY ERROR/
11837	060770	042504	042524	052103		
11838	060776	042105	050040	051101		
11839	061004	052111	020131	051105		
11840	061012	047522	000122			
11841	061016	047125	042504	044506	EM74: .ASCIZ	/UNDEFINED ERROR/
11842	061024	042516	020104	051105		
11843	061032	047522	000122			
11844	061036	040515	045522	047111	EM75: .ASCIZ	/MARKING THIS SECTOR BAD/
11845	061044	020107	044124	051511		
11846	061052	051440	041505	047524		
11847	061060	020122	040502	000104		
11848	061066	040502	020104	040504	EM76: .ASCIZ	/BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
11849	061074	040524	053040	051105		
11850	061102	043111	047047	053440		
11851	061110	052111	020110	042522		
11852	061116	042101	020056	041505		
11853	061124	020103	043117	046040		
11854	061132	051501	020124	042522		
11855	061140	051124	020131	051511		
11856	061146	000072				
11857	061150	042522	051124	020131	EM77: .ASCIZ	/RETRY SUCCESSFUL/
11858	061156	052523	041503	051505		
11859	061164	043123	046125	000		
11860	061171	122	052105	054522	EM100: .ASCIZ	/RETRY UNSUCCESSFUL/
11861	061176	052440	051516	041525		
11862	061204	042503	051523	052506		
11863	061212	000114				
11864	061214	040503	047116	052117	EM101: .ASCIZ	/CANNOT FIND A VALID HEADER IN TRACK JUST READ/
11865	061222	043040	047111	020104		
11866	061230	020101	040526	044514		
11867	061236	020104	042510	042101		
11868	061244	051105	044440	020116		
11869	061252	051124	041501	020112		
11870	061260	052512	052123	051040		
11871	061266	040505	000104			
11872	061272	040502	020104	042523	EM102: .ASCIZ	/BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
11873	061300	052103	051117	042440		
11874	061306	051122	051117	047440		
11875	061314	020116	042523	052103		
11876	061322	051117	047040	052117		
11877	061330	046040	051511	042524		
11878	061336	020104	040502	000104		
11879	061344	044524	042515	026504	EM103: .ASCIZ	/TIMED-OUT ON READ HDR/
11880	061352	052517	020124	047117		
11881	061360	051040	040505	020104		
11882	061366	042110	000122			
11883	061372	044524	042515	026504	EM104: .ASCIZ	/TIMED-OUT ON SEEK/

E04

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
DZR6N8.CMB TRAP TABLE

MACY11 27(732) 03-NOV-76 21:43 PAGE 251

11884	061400	052517	020124	047117	
11885	061406	051440	042505	000113	
11886	061414	051104	053111	020105	EM105: .ASCIZ /DRIVE SIEZED BY OTHER PORT/
11887	061422	044523	055105	042105	
11888	061430	041040	020131	052117	
11889	061436	042510	020122	047520	
11890	061444	052122	000		
11891	061447	104	052101	020101	EM106: .ASCIZ /DATA MISCMPR WHILE BAI SET/
11892	061454	044515	041523	050115	
11893	061462	020122	044127	046111	
11894	061470	020105	040502	020111	
11895	061476	042523	000124		
11896	061502	047516	047040	046505	EM107: .ASCIZ /NO NEM ERROR WHEN REF'ING LOC 760000/
11897	061510	042440	051122	051117	
11898	061516	053440	042510	020116	
11899	061524	042522	023506	047111	
11900	061532	020107	047514	020103	
11901	061540	033067	030060	030060	
11902	061546	000			
11903	061547	111	052116	050122	EM110: .ASCIZ /INTRPT WHEN CNTRLR NOT RDY/
11904	061554	020124	044127	047105	
11905	061562	041440	052116	046122	
11906	061570	020122	047516	020124	
11907	061576	042122	000131		
11908	061602	047516	040440	052124	EM111: .ASCIZ /NO ATT'N ON SEEK/
11909	061610	047047	047440	020116	
11910	061616	042523	045505	000	
11911	061623	104	044522	042526	EM112: .ASCIZ /DRIVE'S CYLINDER INCORRECT/
11912	061630	051447	041440	046131	
11913	061636	047111	042504	020122	
11914	061644	047111	047503	051122	
11915	061652	041505	000124		
11916	061656	041101	051117	026524	EM113: .ASCIZ /ABORT- CAN'T READ BSF/
11917	061664	041440	047101	052047	
11918	061672	051040	040505	020104	
11919	061700	051502	000106		
11920	061704	052113	030461	043040	EM114: .ASCIZ /KT11 FAILURE/
11921	061712	044501	052514	042522	
11922	061720	000			
11923	061721	115	046505	050040	EM115: .ASCIZ /MEM PARITY ERR/
11924	061726	051101	052111	020131	
11925	061734	051105	000122		
11926	061740	052503	051122	047105	DH100: .ASCIZ /CURRENT COMMAND :/
11927	061746	020124	047503	046515	
11928	061754	047101	020104	000072	
11929	061762	051120	053105	047511	DH105: .ASCIZ /PREVIOUS COMMAND :/
11930	061770	051525	041440	046517	
11931	061776	040515	042116	035040	
11932	062004	000			
11933	062005	122	033113	030461	DH200: .ASCIZ /RK611 REGISTERS :/
11934	062012	051040	043505	051511	
11935	062020	042524	051522	035040	
11936	062026	000			
11937	062027	122	046505	044501	DH500: .ASCIZ /REMAINING REGISTERS NOT VALID/
11938	062034	044516	043516	051040	
11939	062042	043505	051511	042524	

12052	063210	051123	000063						
12053	063214	052113	030461	051040	DH705:	.ASCIZ	/KT11 REGS :/		
12054	063222	043505	020123	000072					
12055	063230	042515	020115	042522	DH706:	.ASCIZ	/MEM REGS :/		
12056	063236	051507	035040	000					
12057	063243	120	004503	047514	DH707:	.ASCIZ	/PC LOERAD HIERAD MEMSYS/		
12058	063250	051105	042101	044011					
12059	063256	042511	040522	004504					
12060	063264	042515	051515	051531					
12061	063272	000							
12062	063273	120	004503	051503	DH710:	.ASCIZ	/PC CSR AD CSR/		
12063	063300	020122	042101	041411					
12064	063306	051123	000						
12065	063311	103	046131	047111	DH711:	.ASCIZ	/CYLINDERS :/		
12066	063316	042504	051522	035040					
12067	063324	000							
12068	063325	122	042113	020123	DH204:	.ASCIZ	/RKDS RKER RKMR2 RKMR3/		
12069	063332	020040	051040	042513					
12070	063340	020122	020040	051040					
12071	063346	046513	031122	020040					
12072	063354	051040	046513	031522					
12073	063362	000							
12074	063363	105	051122	051117	DH502:	.ASCIZ	/ERROR TRYING TO GET FAILURE INFO/		
12075	063370	052040	054522	047111					
12076	063376	020107	047524	043440					
12077	063404	052105	043040	044501					
12078	063412	052514	042522	044440					
12079	063420	043116	000117						
12080	063424	042523	047503	042116	DH503:	.ASCIZ	/SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/		
12081	063432	052040	046511	047505					
12082	063440	052125	047440	041503					
12083	063446	051125	042522	020104					
12084	063454	042507	052124	047111					
12085	063462	020107	051104	053111					
12086	063470	020105	052123	052101					
12087	063476	051525	000						
12088	063501	116	046525	042502	DH800:	.ASCIZ	/NUMBER OF RETRIES:/		
12089	063506	020122	043117	051040					
12090	063514	052105	044522	051505					
12091	063522	000072							
12092									
12093	063524	001116	005502	005500	DT100:	.EVEN .WORD	SERRPC, DRIVE, ERRCOM, \$REG0, \$REG1, \$REG2, \$REG3		
12094	063532	001162	001164	001166					
12095	063540	001170							
12096	063542	001256	001260		DT102:	.WORD	\$REG36, \$REG37		
12097	063546	001174	001176	001200	DT201:	.WORD	\$REG5, \$REG6, \$REG7, \$REG10, \$REG11, \$REG12, \$REG13		
12098	063554	001202	001204	001206					
12099	063562	001210							
12100	063564	001212	001214		DT202:	.WORD	\$REG14, \$REG15		
12101	063570	001216	001220	001222	DT203:	.WORD	\$REG16, \$REG17, \$REG20, \$REG21, \$REG22, \$REG23, \$REG24, \$REG25		
12102	063576	001224	001226	001230					
12103	063604	001232	001234						
12104	063610	001174	001176	001200	DT601:	.WORD	\$REG5, \$REG6, \$REG7		
12105	063616	001202	001204	001206	DT602:	.WORD	\$REG10, \$REG11, \$REG12, \$REG13, \$REG14, \$REG15, \$REG16		
12106	063624	001210	001212	001214					
12107	063632	001216							

12108	063634	005256	005254	DT103:	.WORD	PRMPHD,PRMPLO	
12109							
12110	063640	000006		DF01:	.WORD	6	;NUMBER OF HEADER LINES
12111	063642	000			.BYTE	0	;NUMBER OF COL FOR FIRST HDR
12112	063643	000			.BYTE	0	;ALL CHARACTERS OCTAL
12113	063644	062144			.WORD	DH101	;SECOND HDR LINE
12114	063646	007	000		.BYTE	7,0	;NUM OF COL-ALL OCTAL
12115	063650	062233			.WORD	DH102	
12116	063652	002	000		.BYTE	2,0	
12117	063654	062005			.WORD	DH200	
12118	063656	000	000		.BYTE	0,0	
12119	063660	062336			.WORD	DH201	
12120	063662	007	000		.BYTE	7,0	
12121	063664	062027			.WORD	DH500	
12122	063666	000	000		.BYTE	0,0	
12123							
12124	063670	000007		DF02:	.WORD	7	
12125	063672	000	000		.BYTE	0,0	
12126	063674	062144			.WORD	DH101	
12127	063676	007	000		.BYTE	7,0	
12128	063700	062233			.WORD	DH102	
12129	063702	002	000		.BYTE	2,0	
12130	063704	062005			.WORD	DH200	
12131	063706	000	000		.BYTE	0,0	
12132	063710	062336			.WORD	DH201	
12133	063712	007	000		.BYTE	7,0	
12134	063714	062425			.WORD	DH202	
12135	063716	002	000		.BYTE	2,0	
12136	063720	062442			.WORD	DH203	
12137	063722	010	000		.BYTE	10,0	
12138							
12139	063724	000010		DF03:	.WORD	10	
12140	063726	000	000		.BYTE	0,0	
12141	063730	062144			.WORD	DH101	
12142	063732	007	000		.BYTE	7,0	
12143	063734	062233			.WORD	DH102	
12144	063736	002	000		.BYTE	2,0	
12145	063740	062005			.WORD	DH200	
12146	063742	000	000		.BYTE	0,0	
12147	063744	062336			.WORD	DH201	
12148	063746	007	000		.BYTE	7,0	
12149	063750	062065			.WORD	DH501	
12150	063752	000	000		.BYTE	0,0	; "THE FOLLOWING REG DATA MB INCORRECT"
12151	063754	062425			.WORD	DH202	
12152	063756	002	000		.BYTE	2,0	
12153	063760	062442			.WORD	DH203	
12154	063762	010	000		.BYTE	10,0	
12155							
12156	063764	000003		DF04:	.WORD	3	
12157	063766	000	000		.BYTE	0,0	
12158	063770	062144			.WORD	DH101	
12159	063772	007	000		.BYTE	7,0	
12160	063774	062233			.WORD	DH102	
12161	063776	002	000		.BYTE	2,0	
12162							
12163	064000	000007		DF05:	.WORD	7	;OPI, HVRC

12164	064002	000	000	.BYTE	0,0
12165	064004	062144		.WORD	DH101
12166	064006	007	000	.BYTE	7,0
12167	064010	062233		.WORD	DH102
12168	064012	002	000	.BYTE	2,0
12169	064014	062536		.WORD	DH601
12170	064016	000	000	.BYTE	0,0
12171	064020	062777		.WORD	DH606
12172	064022	003	000	.BYTE	3,0
12173	064024	062560		.WORD	DH602
12174	064026	000	000	.BYTE	0,0
12175	064030	062777		.WORD	DH606
12176	064032	003	000	.BYTE	3,0
12177					
12178	064034	000007		DF07: .WORD	7
12179	064036	000	000	.BYTE	0,0
12180	064040	062144		.WORD	DH101
12181	064042	007	000	.BYTE	7,0
12182	064044	062233		.WORD	DH102
12183	064046	002	000	.BYTE	2,0
12184	064050	062625		.WORD	DH604
12185	064052	000	000	.BYTE	0,0
12186	064054	062660		.WORD	DH6041
12187	064056	003	000	.BYTE	3,0
12188	064060	062725		.WORD	DH605
12189	064062	000	000	.BYTE	0,0
12190	064064	062742		.WORD	DH6051
12191	064066	003	000	.BYTE	3,0
12192					
12193	064070	000005		DF10: .WORD	5
12194	064072	000	000	.BYTE	0,0
12195	064074	062144		.WORD	DH101
12196	064076	007	000	.BYTE	7,0
12197	064100	062233		.WORD	DH102
12198	064102	002	000	.BYTE	2,0
12199	064104	062625		.WORD	DH604
12200	064106	000	000	.BYTE	0,0
12201	064110	062660		.WORD	DH6041
12202	064112	003	000	.BYTE	3,0
12203					
12204	064114	000004		DF11: .WORD	4
12205	064116	000	000	.BYTE	0,0
12206	064120	062625		.WORD	DH604
12207	064122	000	000	.BYTE	0,0
12208	064124	062660		.WORD	DH6041
12209	064126	003	000	.BYTE	3,0
12210	064130	063045		.WORD	DH701
12211	064132	000	000	.BYTE	0,0
12212					
12213	064134	000006		DF12: .WORD	6
12214	064136	000	000	.BYTE	0,0
12215	064140	062144		.WORD	DH101
12216	064142	007	000	.BYTE	7,0
12217	064144	062233		.WORD	DH102
12218	064146	002	000	.BYTE	2,0
12219	064150	062005		.WORD	DH200

12220	064152	070	000	.BYTE	0,0	
12221	064154	062336		.WORD	DH201	
12222	064156	007	000	.BYTE	7,0	
12223	064160	063325		.WORD	DH204	
12224	064162	004	000	.BYTE	4,0	
12225						
12226	064164	000006		DF13: .WORD	6	;FORMAT FOR 2ND LEVEL ERROR
12227	064166	000	000	.BYTE	0,0	;IN HEADER COMPARE ERROR
12228	064170	062144		.WORD	DH101	;AND 2ND LEVEL HEADER
12229	064172	007	000	.BYTE	7,0	;VRC ERROR
12230	064174	062233		.WORD	DH102	
12231	064176	002	000	.BYTE	2,0	
12232	064200	062536		.WORD	DH601	
12233	064202	000	000	.BYTE	0,0	
12234	064204	062777		.WORD	DH606	
12235	064206	003	000	.BYTE	3,0	
12236	064210	063363		.WORD	DH502	
12237	064212	000	000	.BYTE	0,0	
12238						
12239	064214	000006		DF14: .WORD	6	;FORMAT FOR 2ND LEVEL ERROR
12240	064216	000	000	.BYTE	0,0	;IN OPERATION INCOMPLETE ERROR
12241	064220	062144		.WORD	DH101	
12242	064222	007	000	.BYTE	7,0	
12243	064224	062233		.WORD	DH102	
12244	064226	002	000	.BYTE	2,0	
12245	064230	062536		.WORD	DH601	
12246	064232	000	000	.BYTE	0,0	
12247	064234	062777		.WORD	DH606	
12248	064236	003	000	.BYTE	3,0	
12249	064240	063363		.WORD	DH502	
12250	064242	000	000	.BYTE	0,0	
12251						
12252	064244	000011		DF15: .WORD	11	
12253	064246	000	000	.BYTE	0,0	
12254	064250	062144		.WORD	DH101	
12255	064252	007	000	.BYTE	7,0	
12256	064254	062233		.WORD	DH102	
12257	064256	002	000	.BYTE	2,0	
12258	064260	062005		.WORD	DH200	
12259	064262	000	000	.BYTE	0,0	
12260	064264	062336		.WORD	DH201	
12261	064266	007	000	.BYTE	7,0	
12262	064270	062425		.WORD	DH202	
12263	064272	002	000	.BYTE	2,0	
12264	064274	063424		.WORD	DH503	
12265	064276	000	000	.BYTE	0,0	
12266	064300	062065		.WORD	DH501	
12267	064302	000	000	.BYTE	0,0	
12268	064304	062442		.WORD	DH203	
12269	064306	010	000	.BYTE	10,0	
12270						
12271	064310	000011		DF16: .WORD	11	
12272	064312	000	000	.BYTE	0,0	
12273	064314	062144		.WORD	DH101	
12274	064316	007	000	.BYTE	7,0	
12275	064320	062233		.WORD	DH102	

12276	064322	002	000		.BYTE	2,0
12277	064324	062005			.WORD	DH200
12278	064326	000	000		.BYTE	0,0
12279	064330	062336			.WORD	DH201
12280	064332	007	000		.BYTE	7,0
12281	064334	062425			.WORD	DH202
12282	064336	002	000		.BYTE	2,0
12283	064340	063363			.WORD	DH502
12284	064342	000	000		.BYTE	0,0
12285	064344	062065			.WORD	DH501
12286	064346	000	000		.BYTE	0,0
12287	064350	062442			.WORD	DH203
12288	064352	010	000		.BYTE	10,0
12289						
12290	064354	000005		DF17:	.WORD	5
12291	064356	000	000		.BYTE	0,0
12292	064360	062144			.WORD	DH101
12293	064362	007	000		.BYTE	7,0
12294	064364	062233			.WORD	DH102
12295	064366	002	000		.BYTE	2,0
12296	064370	063311			.WORD	DH711
12297	064372	000	000		.BYTE	0,0
12298	064374	063134			.WORD	DH702
12299	064376	002	000		.BYTE	2,0
12300						
12301	064400	000002		DF20:	.WORD	2
12302	064402	000	000		.BYTE	0,0
12303	064404	062274			.WORD	DH104
12304	064406	002	000		.BYTE	2,0
12305						
12306	064410	000002		DF21:	.WORD	2
12307	064412	000	000		.BYTE	0,0
12308	064414	062742			.WORD	DH6051
12309	064416	003	000		.BYTE	3,0
12310						
12311	064420	000006		DF22:	.WORD	6
12312	064422	000	000		.BYTE	0,0
12313	064424	061740			.WORD	DH100
12314	064426	000	000		.BYTE	0,0
12315	064430	062144			.WORD	DH101
12316	064432	007	000		.BYTE	7,0
12317	064434	062233			.WORD	DH102
12318	064436	002	000		.BYTE	2,0
12319	064440	062314			.WORD	DH106
12320	064442	000	000		.BYTE	0,0
12321	064444	062274			.WORD	DH104
12322	064446	002	000		.BYTE	2,0
12323						
12324	064450	000002		DF23:	.WORD	2
12325	064452	000	000		.BYTE	0,0
12326	064454	063501			.WORD	DH800
12327	064456	001	000		.BYTE	1,0
12328						
12329	064460	000001		DF24:	.WORD	1
12330	064462	002	000		.BYTE	2,0
12331						

12332	064464	000000		
12333	064466	007	000	
12334				
12335	064470	000002		
12336	064472	002	000	
12337	064474	062442		
12338	064476	010	000	
12339				
12340	064500	000005		
12341	064502	000	000	
12342	064504	062144		
12343	064506	007	000	
12344	064510	062233		
12345	064512	002	000	
12346	064514	063150		
12347	064516	000	000	
12348	064520	063134		
12349	064522	002	000	
12350				
12351	064524	000005		
12352	064526	000	000	
12353	064530	062144		
12354	064532	007	000	
12355	064534	062233		
12356	064536	002	000	
12357	064540	063214		
12358	064542	000	000	
12359	064544	063174		
12360	064546	004	000	
12361				
12362	064550	000005		
12363	064552	000	000	
12364	064554	062144		
12365	064556	007	000	
12366	064560	062233		
12367	064562	002	000	
12368	064564	063230		
12369	064566	000	000	
12370	064570	063273		
12371	064572	003	000	
12372				
12373	064574			
12374				
12375	064574	005015	020040	020040
12376	064602	020040	020040	020040
12377	064610	025052	020052	047516
12378	064616	042524	025040	025052
12379	064624	005015	012	
12380	064627	101	046114	042040
12381	064634	044522	042526	020123
12382	064642	047524	041040	020105
12383	064650	042524	052123	042105
12384	064656	046440	051525	020124
12385	064664	040510	042526	035040
12386	064672	005015	012	
12387	064675	061	020056	020101

DF25: .WORD 0
.BYTE 7,0

DF26: .WORD 2
.BYTE 2,0
.WORD DH203
.BYTE 10,0

DF27: .WORD 5
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
.WORD DH703
.BYTE 0,0
.WORD DH702
.BYTE 2,0

DF30: .WORD 5
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
.WORD DH705
.BYTE 0,0
.WORD DH704
.BYTE 4,0

DF31: .WORD 5
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
.WORD DH706
.BYTE 0,0
.WORD DH710
.BYTE 3,0

RWBUF: ;READ/WRITE DATA BUF <AT LEAST 512(DEC) WORDS>

NOTMSG: .ASCII <15><12>/ *** NOTE ***/<15><12><12>

.ASCII /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>

.ASCII /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>

12388	064702	030062	047440	020122	
12389	064710	031062	051440	041505	
12390	064716	047524	020122	047506	
12391	064724	046522	052101	042524	
12392	064732	020104	040503	052122	
12393	064740	044522	043504	006505	
12394	064746	012			
12395	064747	062	020056	042510	.ASCII /2. HEADS MANUALLY LOADED/<15><12>
12396	064754	042101	020123	040515	
12397	064762	052516	046101	054514	
12398	064770	046040	040517	042504	
12399	064776	006504	012		
12400	065001	063	020056	042504	.ASCII /3. DESIRED PORT SELECTED/<15><12>
12401	065006	044523	042522	020104	
12402	065014	047520	052122	051440	
12403	065022	046105	041505	042524	
12404	065030	006504	012		
12405	065033	064	020056	051127	.ASCII /4. WRITE LOCK DISABLED/<15><12>
12406	065040	052111	020105	047514	
12407	065046	045503	042040	051511	
12408	065054	041101	042514	006504	
12409	065062	012			
12410	065063	065	020056	051104	.ASCII /5. DRIVE READY LIGHT ON/<15><12><12>
12411	065070	053111	020105	042522	
12412	065076	042101	020131	044514	
12413	065104	044107	020124	047117	
12414	065112	005015	012		
12415	065115	104	044522	042526	.ASCII /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12416	065122	020123	047516	020124	
12417	065130	047524	041040	020105	
12418	065136	042524	052123	042105	
12419	065144	046440	051525	020124	
12420	065152	040510	042526	041040	
12421	065160	052117	006510	012	
12422	065165	120	051117	051524	.ASCIZ /PORTS DE-SELECTED./<15><12><12>
12423	065172	042040	026505	042523	
12424	065200	042514	052103	042105	
12425	065206	006456	005012	000	
12426					
12427	065213	015	044412	051516	HLPFIL: .ASCII <15><12>/INSTRUCTIONS FOR USING RK06 HEAD ALIGNMENT AID :/<15><12>
12428	065220	051124	041525	044524	
12429	065226	047117	020123	047506	
12430	065234	020122	051525	047111	
12431	065242	020107	045522	033060	
12432	065250	044040	040505	020104	
12433	065256	046101	043511	046516	
12434	065264	047105	020124	044501	
12435	065272	020104	006472	012	
12436	065277	052	025052	025052	.ASCII /*****/<15><12>
12437	065304	025052	025052	025052	
12438	065312	025052	025052	025052	
12439	065320	025052	025052	025052	
12440	065326	025052	025052	025052	
12441	065334	025052	025052	025052	
12442	065342	025052	025052	025052	
12443	065350	025052	025052	006452	

805

12444	065356	012			
12445	065357	115	052517	052116	.ASCII /MOUNT AN RK06 ALIGNMENT CARTRIDGE ON THE DESIRED /<15><12>
12446	065364	040440	020116	045522	
12447	065372	033060	040440	044514	
12448	065400	047107	042515	052116	
12449	065406	041440	051101	051124	
12450	065414	042111	042507	047440	
12451	065422	020116	044124	020105	
12452	065430	042504	044523	042522	
12453	065436	020104	005015		
12454	065442	051104	053111	026105	.ASCII /DRIVE, AND INSURE THAT THE DRIVE IS WRITE-LOCKED. /<15><12>
12455	065450	040440	042116	044440	
12456	065456	051516	051125	020105	
12457	065464	044124	052101	052040	
12458	065472	042510	042040	044522	
12459	065500	042526	044440	020123	
12460	065506	051127	052111	026505	
12461	065514	047514	045503	042105	
12462	065522	020056	005015		
12463	065526	047503	047116	041505	.ASCII /CONNECT THE ALIGNMENT INDICATOR TO THE DESIRED DRIVE, /<15><12>
12464	065534	020124	044124	020105	
12465	065542	046101	043511	046516	
12466	065550	047105	020124	047111	
12467	065556	044504	040503	047524	
12468	065564	020122	047524	052040	
12469	065572	042510	042040	051505	
12470	065600	051111	042105	042040	
12471	065606	044522	042526	006454	
12472	065614	012			
12473	065615	126	040511	052040	.ASCII /VIA THE HEAD ALIGNMENT CABLE ONLY, AND CYCLE UP THE /<15><12>
12474	065622	042510	044040	040505	
12475	065630	020104	046101	043511	
12476	065636	046516	047105	020124	
12477	065644	040503	046102	020105	
12478	065652	047117	054514	020054	
12479	065660	047101	020104	054503	
12480	065666	046103	020105	050125	
12481	065674	052040	042510	005015	
12482	065702	051104	053111	027105	.ASCII /DRIVE. /<15><12>
12483	065710	005015			
12484	065712	042522	050123	047117	.ASCII /RESPOND TO ALL REQUESTS FOR PARAMETERS, BY ENTERING /<15><12>
12485	065720	020104	047524	040440	
12486	065726	046114	051040	050505	
12487	065734	042525	052123	020123	
12488	065742	047506	020122	040520	
12489	065750	040522	042515	042524	
12490	065756	051522	020054	054502	
12491	065764	042440	052116	051105	
12492	065772	047111	020107	005015	
12493	066000	044124	020105	042504	.ASCII /THE DESIRED PARAMETER VALUE (NO <CR> NEEDED). /<15><12>
12494	066006	044523	042522	020104	
12495	066014	040520	040522	042515	
12496	066022	042524	020122	040526	
12497	066030	052514	020105	047050	
12498	066036	020117	041474	037122	
12499	066044	047040	042505	042504	

C05

12500	066052	024504	006456	012	
12501	066057	124	042510	042522	.ASCII /THERE ARE TWO MODES OF OPERATION : MANUAL MODE /<15><12>
12502	066064	040440	042522	052040	
12503	066072	047527	046440	042117	
12504	066100	051505	047440	020106	
12505	066106	050117	051105	052101	
12506	066114	047511	020116	020072	
12507	066122	046440	047101	040525	
12508	066130	020114	047515	042504	
12509	066136	006440	012		
12510	066141	101	046114	053517	.ASCII /ALLOWS SELECTION OF DRIVES AND HEADS BY TTY INPUT,/<15><12>
12511	066146	020123	042523	042514	
12512	066154	052103	047511	020116	
12513	066162	043117	042040	044522	
12514	066170	042526	020123	047101	
12515	066176	020104	042510	042101	
12516	066204	020123	054502	052040	
12517	066212	054524	044440	050116	
12518	066220	052125	006454	012	
12519	066225	101	042116	040440	.ASCII /AND AUTO MODE ALLOWS DRIVES AND HEADS TO BE SELECTED/<15><12>
12520	066232	052125	020117	047515	
12521	066240	042504	040440	046114	
12522	066246	053517	020123	051104	
12523	066254	053111	051505	040440	
12524	066262	042116	044040	040505	
12525	066270	051504	052040	020117	
12526	066276	042502	051440	046105	
12527	066304	041505	042524	006504	
12528	066312	012			
12529	066313	102	020131	043117	.ASCII /BY OFF-ON OPERATION OF DRIVE PORT SELECT SWITCHES./<15><12>
12530	066320	026506	047117	047440	
12531	066326	042520	040522	044524	
12532	066334	047117	047440	020106	
12533	066342	051104	053111	020105	
12534	066350	047520	052122	051440	
12535	066356	046105	041505	020124	
12536	066364	053523	052111	044103	
12537	066372	051505	006456	012	
12538	066377	111	020116	044505	.ASCII /IN EITHER MODE, UP TO 5 MINUTES OF SEEK EXERCISES/<15><12>
12539	066404	044124	051105	046440	
12540	066412	042117	026105	052440	
12541	066420	020120	047524	032440	
12542	066426	046440	047111	052125	
12543	066434	051505	047440	020106	
12544	066442	042523	045505	042440	
12545	066450	042530	041522	051511	
12546	066456	051505	005015		
12547	066462	040515	020131	042502	.ASCII /MAY BE REQUESTED FOR EACH DRIVE. /<15><12>
12548	066470	051040	050505	042525	
12549	066476	052123	042105	043040	
12550	066504	051117	042440	041501	
12551	066512	020110	051104	053111	
12552	066520	027105	006440	012	
12553	066525	101	051514	020117	.ASCII /ALSO IN EITHER MODE, A VERIFY OPTION ALLOWS HEAD/<15><12>
12554	066532	047111	042440	052111	
12555	066540	042510	020122	047515	

12556	066546	042504	020054	020101	
12557	066554	042526	044522	054506	
12558	066562	047440	052120	047511	
12559	066570	020116	046101	047514	
12560	066576	051527	044040	040505	
12561	066604	006504	012		
12562	066607	123	046105	041505	.ASCII /SELECTION WITHOUT THE UNLOADING AND LOADING OF /<15><12>
12563	066614	044524	047117	053440	
12564	066622	052111	047510	052125	
12565	066630	052040	042510	052440	
12566	066636	046116	040517	044504	
12567	066644	043516	040440	042116	
12568	066652	046040	040517	041504	
12569	066660	043516	047440	020106	
12570	066666	005015			
12571	066670	044124	020105	051104	.ASCII /THE DRIVE BY THE PROGRAM, WHICH OTHERWISE OCCURS/<15><12>
12572	066676	053111	020105	054502	
12573	066704	052040	042510	050040	
12574	066712	047522	051107	046501	
12575	066720	020054	044127	041511	
12576	066726	020110	052117	042510	
12577	066734	053522	051511	020105	
12578	066742	041517	052503	051522	
12579	066750	005015			
12580	066752	047524	040440	046114	.ASCII /TO ALLOW MANIPULATION OF THE ALIGNMENT TOOL./<15><12>
12581	066760	053517	046440	047101	
12582	066766	050111	046125	052101	
12583	066774	047511	020116	043117	
12584	067002	052040	042510	040440	
12585	067010	044514	047107	042515	
12586	067016	052116	052040	047517	
12587	067024	027114	005015		
12588	067030	047524	051040	051505	.ASCII /TO RESTART EITHER MODE, TYPE <↑>./<15><12>
12589	067036	040524	052122	042440	
12590	067044	052111	042510	020122	
12591	067052	047515	042504	020054	
12592	067060	054524	042520	036040	
12593	067066	055136	027076	005015	
12594	067074	047524	051040	051505	.ASCII /TO RESTART ALIGNMENT AID, TYPE <↑R>./<15><12>
12595	067102	040524	052122	040440	
12596	067110	044514	047107	042515	
12597	067116	052116	040440	042111	
12598	067124	020054	054524	042520	
12599	067132	036040	051136	027076	
12600	067140	005015			
12601	067142	047524	051440	046105	.ASCII /TO SELECT NEW DRIVES IN MANUAL MODE, TYPE <↑C>./<15><12><12>
12602	067150	041505	020124	042516	
12603	067156	020127	051104	053111	
12604	067164	051505	044440	020116	
12605	067172	040515	052516	046101	
12606	067200	046440	042117	026105	
12607	067206	052040	050131	020105	
12608	067214	057074	037103	006456	
12609	067222	005012			
12610	067224	047506	020122	042510	.ASCII /FOR HEAD ALIGNMENT PROCEDURE, REFER TO FIELD /<15><12>
12611	067232	042101	040440	044514	

EOS

12612	067240	047107	042515	052116
12613	067246	050040	047522	042503
12614	067254	052504	042522	020054
12615	067262	042522	042506	020122
12616	067270	047524	043040	042511
12617	067276	042114	006440	012
12618	067303	124	051505	020124
12619	067310	047502	020130	051050
12620	067316	030113	052066	026101
12621	067324	051040	030113	052066
12622	067332	024502	047440	042520
12623	067340	040522	047524	023522
12624	067346	020123	040515	052516
12625	067354	046101	006456	000012
12626				
12627				
12628				
12629	000001			

.ASCIZ /TEST BOX (RK06TA, RK06TB) OPERATOR'S MANUAL./<15><12>

.END

APTSIZ=	000200	4745	6243	11456#															
APTSP0=	000100	10794	11417	11458#															
ASKDRV	024356	6308#	6323	6327	6342	6374	6444												
ASKHED	024576	6360#	6388	6400	6407														
ASKMOE	016124	5139#	5145	5146	5151	5166	5171	5176	5201	5215	5226	5235	5251	5270					
		5279	5305	5328	5400	7674													
ASKMO0	024270	6277	6282	6287#	6301														
ASKTMD	015360	5021#	5024	5025	5030	5054	5062	5068	5084	5095	5104	5129							
ASKTST	015354	4935	5020#																
ASTART	023602	2211	6273#																
ASWREG=	000000	2340	2353																
AS2SP2	056640	8945	11625#																
AATESTN=	000000	2340	2344																
AUNIT =	000000	2340	2347																
AUSMR =	000000	2340	2354																
AUTAL	025414	6470	6477	6484#	6565	6586													
AUTALN	013054	4632#	6489																
AUTEX	026410	6473	6625#	6646	6659														
AUTEXR	013172	4647#	6628																
AUTO	025300	6299	6457#	6505	6636	6682	6694												
AUTODR	013304	4660#	6663#																
AUTOEX	013263	4657#	6664																
AUTVRF	013112	4637#	6487																
AVECT1=	120210	2247#	2340	2379	4796														
AVECT2=	000000	2340	2380																
A.ABNL	003036	3570#	5568#	5571*	6517*	6546*	6649*	6654*	7133*	7196*	7200*	7275*	7279*	8214*					
		8510#	8527*	8536*	9159*	9425*	9438*	9440*	10360										
A.CONT	003040	3571#	10370																
A.NORM	003034	3569#	6102*	6151*	7134*	9160*	9426*	10364											
BADDRV	007362	4277#	7203*	7204*	7205	7214*	7215*	7216	7222*	7223*	7224	7242*	7243*	7244					
BADSEC=	001000	4228#	8123	8346	8799	8807	9371	9470											
BAD632	013310	4662#	7239																
BAI =	000020	3248#	10529																
BA16 =	000400	3224#																	
BA17 =	001000	3225#																	
BDSCHD	037540	8510	8527	8541#															
BDSECT	011245	4457#	8560																
BDSRCK	040612	8121	8344	8789#	9370														
BGNRTY	044434	9083	9436	9442	9463#														
BIT0 =	000001	2141#	3221	3238	3264	3286	3434	3578	3671	4882	4974	6940	7713	7997					
		8014	8029	8068	8085	8139	8181	8286	8301	8332	8368	8412	9042	10644					
BIT00 =	000001	2131#	2141																
BIT01 =	000002	2130#	2140																
BIT02 =	000004	2129#	2139																
BIT03 =	000010	2128#	2138																
BIT04 =	000020	2127#	2137																
BIT05 =	000040	2126#	2136																
BIT06 =	000100	2125#	2135																
BIT07 =	000200	2124#	2134																
BIT08 =	000400	2123#	2133	7838	7870														
BIT09 =	001000	2122#	2132	11248	11379														
BIT1 =	000002	2140#	3239	3266	3435	3579	3672	4220	5545	5593	8256	9046							
BIT10 =	002000	2121#	3226	3255	3277	3312	3326	3339	3354	3368	3447	3588	4229	9075					
		11226																	
BIT11 =	004000	2120#	3227	3256	3278	3297	3313	3327	3340	3355	3369	3448	4230	11386					
BIT12 =	010000	2119#	3229	3257	3279	3314	3328	3341	3342	3356	3370	3449	4231	8957					

K05

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 03-NOV-76 21:43 PAGE 271

DMD = 000040	3307#													
DONE 003113	3649#	8450*	8457	8835*	8842	8846*	8852	8897*	9039*	9153*	9231*	9496*	9499*	
DOTIM 003123	3657#													
DRA = 000001	3286#	9783	10566											
DRDY = 000200	3295#													
DRIEXR 012544	4594#	6416*												
DRISEL 013145	4642#	6584												
DRIV 011167	4447#	7776												
DRIVE 005502	3712#	4990*	5417*	5424*	5425	5428	5456	6050	6089	6100	6117	6320*	6414	
	6515*	6520	6523	6525*	6547*	6552*	6579	6581	6589	6602	6647*	6650	6661	
	6689*	6690	6707	7141	7142	7186	7203	7214	7222	7242	7273	7282	7290	
	7307	9228	12093											
DRNAFG 003127	3661#	8836*	8844	8847*	9052	9225*								
DRNAVL= 010000	4231#													
DRNRDY 011673	4516#	7292*												
DROPDR 011077	4437#	8928												
DROT = 000040	3292#	9298												
DRPAR = 000010	3269#	9238												
DRPDRV= 002000	3447#	10428												
DRVCAL 040716	5437	5443	5563	5566	5570	5573	5641	5984	6053	6071	6103	6107	6124	
	6152	6368	6373	6378	6423	6431	6438	6443	6448	6504	6511	6522	6564	
	6574	6578	6593	6595	6599	6608	6653	6670	6676	6681	6688	7140	7198	
	7229	7236	7249	7277	7301	7314	7316	7328	7341	7343	7355	7371	7775	
	7817	7903	8217	8258	8261	8512	8529	8717	8832*	9439	9481	9491	9526	
	9568													
DRVDSC= 000040	3439#	9406	9868	10034	10096									
DRVERS 003120	3654#	4755*	5422*	8921*	8925	8927*								
DRVHRD= 000020	3438#	9401	9875	10070	10201									
DRVLST 005610	3746#	4883	4908	4955	4963*	4975*	4978*	4988	4999*	5429	6048	6087	6115	
DRVLUP 025472	6496#	6532	6544	6600	6621									
DRVMSK= 000007	3245#	9777	9818	9969	10414									
DRVNED 012037	4535#	7284*												
DRVNO 011163	4446#	5456*	5457*											
DRVPDT= 000004	3436#	9976	10044											
DRVPOS= 000002	3435#	9976	10003	10465										
DRVSEL 013154	4644#	6583*												
DRVSEQ 007344	4274#	4879												
DRVSER 033524	5461	6328	7772#											
DRVSZD= 010000	3449#	9219	9806	10568										
DRVTST 014520	4696	4845	4861#	4867	4868	4869	4877	4928	4929	5023	5051	5067	5094	
	5144	5198	5223	5250	5278	5327	5399	7671	8206					
	3434#	10428												
DRVUSE= 000001	3294#													
DRY = 000200	3217#	9856	10019	10061	10085	10199								
DR.CLR= 000005	3216#	9926	10005	10320	10327	10334	10341	10545						
DR.SEL= 000001	4314#	7193												
DROXOP 007667	3299#													
DS: = 040000	2062#	2277	4732	6230										
DSMR = 177570	3452#	5956	6069	10428	10527									
DTBAII= 100000	3279#	5656	8541	9303	10196									
DTE = 010000	3271#	9266	10196											
DTYE = 000040	2404	2410	2416	2422	2428	2434	2440	2446	2452	2458	2464	2470	2476	
DT100 063524	2482	2488	2494	2500	2506	2512	2518	2524	2530	2536	2542	2548	2554	
	2560	2566	2572	2590	2596	2602	2608	2614	2620	2626	2632	2638	2644	
	2650	2656	2662	2668	2674	2680	2698	2710	2716	2722	2728	2734	2740	
	2746	2752	2758	2764	2806	2812	2818	2824	2830	2842	2848	2872	2878	

OFSHED	011274	4465#	5547																	
OFST =	000004	3288#																		
ONLINE	005530	3723#	6493*	6531	6541	6542*														
OPCOMP	003112	3648#																		
OPCONT	003121	3655#																		
OPI =	020000	3280#	5656	8541	9312	9794	9960	10196												
OPIERR=	000010	4222#	9314	9463																
OPMLST	016610	5157#	5247#	5252	5259															
OR =	000200	3252#																		
OREQSW=	000020	3675#																		
OVL0DR	007155	4251#																		
O.WAIT	003044	3592#	9694	9698*	9808*	9812	9828*	9894*	9907*	9922*	10466*	10526*	10588*	10605*						
PACK =	000103	3191#	7228	7315																
PARMDE	010365	4375#	5141																	
PARMD	002620	3481#	3636	7136	7369	8686	8699	9424	9489	9540	9571									
PARMI	002704	3513#	7142*	9158	9524	9558														
PART =	000001	1990#	2007	2013	2035	2044	2210	2225	3697	3737	3761	3772	3787	3789						
		3793	3794	3796	3802	3804	3808	3809	3811	3820	3824	3832	3834	3838						
		3845	3847	3851	3852	3854	4234	4463	4464	4701	4757	4785	5177	5467						
		5470	6202	7053	7255	7828	7875	8188	8256	8267	8813	12373	12422	12427						
PART1	007115	4244#																		
PART2	007121	4245#	4759	6251																
PAT =	000020	3306#																		
PATRN	005536	3726#	5699*	5776*	5865*	5867*	5889	7990	8061	8168	8692									
PAT00	006026	3859#	7992	8007	8063	8078														
PAT01	006066	3881#																		
PAT02	006126	3903#																		
PAT03	006166	3925#																		
PAT04	006226	3947#																		
PAT05	006266	3969#																		
PAT06	006326	3990#																		
PAT07	006366	4012#																		
PAT08	006426	4033#																		
PAT09	006466	4055#																		
PAT10	006526	4077#																		
PAT11	006566	4099#																		
PAT12	006626	4121#																		
PAT13	006666	4143#	5971	6068																
PAT14	006726	4165#	7919																	
PAT15	006766	4185#	5968	7574	7652*	7658*														
PBLKT	003102	3636#	9691	9778	9804	9972	9997	10415*												
PBSVAL=	001000	3444#	10353																	
PC =%	000007	2075#	4754*	4782*	4785*	4788*	4813*	4818*	4827*	4864*	4866*	4912*	4923*	4927*						
		4991#	5022*	5044*	5045*	5055*	5066*	5089*	5093*	5116*	5140*	5143*	5191*	5192*						
		5202#	5203*	5227*	5228*	5229*	5249*	5273*	5277*	5290*	5295*	5298*	5326*	5370*						
		5374#	5377*	5398*	5410*	5435*	5437*	5443*	5454*	5455*	5461*	5462*	5465*	5500*						
		5501#	5504*	5505*	5519*	5523*	5529*	5563*	5566*	5570*	5573*	5641*	5686*	5687*						
		5690#	5691*	5693*	5696*	5701*	5702*	5719*	5722*	5723*	5724*	5726*	5728*	5730*						
		5731#	5734*	5736*	5763*	5764*	5767*	5768*	5770*	5773*	5779*	5781*	5798*	5802*						
		5804#	5806*	5808*	5810*	5812*	5816*	5818*	5824*	5851*	5852*	5855*	5856*	5858*						
		5861#	5864*	5869*	5870*	5887*	5897*	5898*	5899*	5901*	5903*	5905*	5906*	5910*						
		5912#	5952*	5953*	5956*	5957*	5978*	5984*	5989*	5994*	6035*	6036*	6039*	6040*						
		6053#	6061*	6071*	6076*	6093*	6094*	6103*	6107*	6121*	6124*	6135*	6144*	6146*						
		6152#	6177*	6180*	6190*	6195	6288*	6300*	6309*	6322*	6324*	6326*	6328*	6333*						
		6351#	6361*	6368*	6373*	6378*	6387*	6392*	6394*	6413*	6420*	6421*	6423*	6431*						
		6468#	6443*	6448*	6450*	6458*	6462*	6478*	6490*	6504*	6511*	6513*	6514*	6522*						

P.TRCK= 000005	9559*	9934	10470	10512	10614									
	3424*	5513*	5527	5531*	5532	5537*	5548*	5596	5635*	5636	5960*	6393*	6614*	
P.WC = 000012	7232*	7814*	7897*	7899	7904*	7905	7960	8423*	8502*	8604	8658*	9095		
	3429*	5705*	5709*	5784*	5788*	5873*	5877*	5962*	5965*	6064*	6067*	7234*	7816*	
	8253*	8477	8508*	8662*	8688	8725	9099	10510						
P.WCR = 000022	3461*	9109	10052*	10282*										
QNEWSM= 000000	3180													
Q.INIT= 040000	3451*													
RCDASM= 000010	3674*													
RCLREQ= 004000	4230*	9236	9241	9286	9291	9296	9301	9435						
RDALHD= 000164	3209*	9824	10603											
RDCHR = 104405	6279	7436	11618*											
RDCHRS 032214	4866	4912	4927	5022	5066	5093	5143	5249	5277	5326	5398	6842	7430*	
	7608	7626												
RDDATH= 000121	3198*	5729	5809	5904	5988	7230	7812	8500						
RDGATE= !00000	3317*													
RDHDO 045004	9323	9348	9549*											
RDHEAD= 000125	3200*	5436	9566	9938	10619									
ROSTAT= 000141	3208*	6110	6573	6594	7195	7274	7342	7774	9525	10539				
ROY = 000200	3223*	6142	9431	10496	10549									
RECAL = 000113	3195*	5640	6367	6372	6377	6430	6437	6442	6447	6503	6510	6675	6680	
	6687	7248	7313	8216	9437	10482								
RECODE 005476	3710*	5981	6104	7237	7367*	8123	8346	8718	8721	8799*	8807*	8898	8908*	
	9155*	9156	9183*	9188*	9209	9211	9236*	9241*	9281*	9286*	9291*	9296*	9301*	
	9306*	9314*	9318	9324	9339*	9343	9349	9364*	9371	9374*	9378*	9381*	9394*	
	9395*	9404*	9409*	9414*	9419*	9427	9433*	9435	9441	9443*	9463	9465*	9470	
	9477*	9497*	9529*											
RED8SF 037276	5455	8497*												
REFNEM 037126	5978	5989	8448*											
REISSU 003130	3662*	9054*												
RELEAS= 000140	3207*	10586												
REPLPK 007476	4292*	4910												
REPSUP 042130	5994	6121	6135	6146	6915	6958	8060	8326	8518	9089*	9161			
RESREG= 104407	6826	6907	7031	7087	7739	7797	7884	7908	7925	7972	8041	8122	8183	
	8262	8302	8345	8413	8488	8593	8618	8676	8704	8800	8885	9025	9140	
	9492	9541	9572	9631	10966	11018	11620*							
	2145*													
RESVEC= 000010	9159	9438	9496*											
RETAML 044606	5660	6715	8545	9160	9499*									
RETAML 044624	3176*	9518	9834	9904	9984	10239	10287	10476*	10501					
RKASOF= 000016	3171*	9517	10235	10283	10509*									
RKBAB 007244	4261*	4812												
RKBAS 003026	3566*	4793*	4797*	4811	4814*	7135	9326	9351	9430	9479	9510	9687	9756	
	10420													
RKCS1 = 000000	3169*	4921*	6560*	7208*	7295*	9226*	9327*	9331	9352*	9356	9512	9779	9831	
	9889	9939*	9948	10130*	10131	10140*	10179*	10180	10204*	10232	10493*	10495	10534*	
	10546*	10548	10578*	10600*	10622*	10631*	10641	10651	10666*					
RKCS2 = 000010	3173*	9513	9757	9917	10175*	10233	10281	10464*	10530*	10541*	10591*	10615*	10640*	
RKDA = 000006	3172*	9515	9934*	10236	10284	10470*	10512*	10614*						
RKDB = 000024	3179*	5447	5448	9328	9329	9330	9353	9354	9355	9913	9914	9915		
RKDC = 000020	3177*	9514												
RKDCYL= 000020	3178*	9933*	10240	10289	10469*	10511*	10613*							
RKDS = 000012	3174*	9431	9519	9782	10237	10285								
RKECPS= 000030	3184*	9848	10244											
RKECPT= 000032	3186*	9847	10245											
RKER = 000014	3175*	9520	9791	9957	10238	10286								

RKMR1 = 000026	3180#	9925*	10004*	10241	10318*	10325*	10332*	10339*	10440*						
RKMR2 = 000034	3181#	10242	10552												
RKMR3 = 000036	3182#	10243	10553												
RKPAT = 000032	3185#														
RKPOS = 000030	3183#														
RKPRI 003032	3568#	4771	4802*	4821	4837*	9686	10410								
RKPRTY 007304	4267#	4826													
RKVADR 007264	4264#	4817													
RKVEC 003030	3567#	4769	4798*	4799*	4816	4819*	6256								
RKWC = 000002	3170#	9516	10234	10282	10510*										
RLS = 000010	3247#	10590													
RNCADR 034044	6076	6421	6668	7865#											
RNDLNG= 016514	4216#	6418	6643												
RNDSHT= 002734	4215#	6639													
RUNTST 017330	5177	5395#													
RMBUF 064574	3739	5552	5694	5771	5859	5986	5987*	5992	5996	6971	7233	7240	7815		
	7820	7880	7882	7994	8065	8246	8252	8427	8690	12373#					
RWNRDY 012751	4618#	7346													
R.ABNL 050774	9699	9809	9884	9901	10079	10099	10359#	10569							
R.CONT 051020	9761	9799	9863	9872	9943	9965	9988	9994	10027	10067	10091	10136	10188		
	10211	10349	10367#	10583	10637	10646	10659								
R.NORM 051006	9837	9930	10039	10363#	10506	10579	10667								
RO =.000000	2066#	4777*	4779*	4792*	4796*	4798	4800*	4801*	4802	4828*	4829	4831	4833*		
	4834*	4835*	4836*	4837	4870	4881*	4885	4891*	4896	4916	4931	4937	4948		
	4960*	4961	4964*	4965	4969*	4975*	4978*	4980*	4981	4987*	4988	4990	4995*		
	4996	4999*	5026	5070	5078*	5080	5097	5108	5110*	5111*	5113	5118*	5119		
	5121	5147	5253	5261*	5263	5281	5287	5330	5335	5357	5362	5364*	5365		
	5402	5404	5428*	5429	5433	5552*	5554*	5558*	5564	5576	5577*	5582	5584*		
	5587	5708*	5709	5787*	5788	5876*	5877	5968*	5971*	5974	5992*	5995	6044*		
	6046	6048	6050	6052	6055*	6075*	6077*	6079*	6084*	6085	6087	6089	6091		
	6096*	6112*	6113	6115	6117	6119	6123	6137*	6187*	6190	6270*	6272*	6418*		
	6427*	6516*	6518	6527*	6534*	6535*	6536	6549*	6550	6554*	6556	6666*	6685*		
	6703	6705*	6710*	6712	6714*	6803*	6807*	6812*	6814	6846	6849	6858*	6861		
	7006*	7007*	7009	7011*	7013*	7016*	7022*	7024*	7025*	7076*	7082*	7084	7091*		
	7093*	7096*	7155	7157*	7159*	7160	7163*	7433*	7465	7472*	7473	7487*	7488*		
	7489	7492*	7497*	7597	7612	7630	7639	7641*	7642*	7644	7649*	7652	7658*		
	7731	7748*	7750*	7756*	7757*	7761*	7788*	7789*	7790*	7791	7836	7841*	7842*		
	7843*	7845*	7850*	7853*	7881	7899*	7900*	7919*	7921*	7922*	7949*	7952*	7953		
	7955	7992*	7996*	7999*	8002*	8003*	8004*	8005*	8006*	8007*	8017	8032	8063*		
	8067*	8070*	8073*	8074*	8075*	8076*	8077*	8078*	8088	8105	8118*	8206*	8219		
	8222*	8228*	8246*	8248*	8341*	8477*	8481*	8483*	8485	8558*	8563	8570	8573		
	8579	8583	8590*	8603*	8607	8640*	8653*	8657	8670*	8671*	8672	8686*	8698*		
	8701	8760	8863*	8865	8866	8867	8868	8869	8870	8933*	8936*	8937*	8938*		
	8939*	8940*	8941*	8942	8948	8987	8989*	8990	8991	8992	9009	9010	9021		
	9023*	9093*	9094*	9095*	9096*	9097*	9098*	9099*	9100*	9105*	9106*	9107*	9108*		
	9109*	9111*	9112*	9113*	9114*	9115*	9116*	9117*	9118*	9119*	9120*	9121*	9122*		
	9125*	9126	9128	9130	9131	9133	9135	9136	9139	9228*	9229	9365*	9482*		
	9483	9484	9485	9486	9487	9488	9521*	9552*	9553*	9554*	9555*	9556*	9557*		
	9560	9755	9849*	9854*	9857*	9899*	9927*	10006*	10020*	10059*	10062*	10086*	10101*		
	10137*	10138*	10142	10143*	10202*	10206	10207*	10213*	10214*	10321*	10328*	10335*	10342*		
	10350*	10351*	10354	10355*	10402	10430*	10431*	10435*	10436	10562*	10572*	10627*	10670*		
	10694	10697*	10700	10722*	10729*	10745	10748*	10751*	10755*	10758*	10761	10765*	10790		
	10791*	10796	10801	10804*	10862*	10865*	10871*	10872*	10898*	10899*	10905*	10910*	10912*		
	10915*	10917*	10927*	10928*	10956*	10960	10970*	10972*	10974*	10994*	10995	11013*	11017*		
	11051	11052*	11053	11055	11057*	11058	11061*	11154	11164*	11168	11184	11185	11193*		
	11411	11419*	11423	11424	11426*	11427*	11428	11450*	11476	11482*	11530	11538*	11560		

K06

DZR6N-B - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2
 DZR6NB.CMB CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 03-NOV-76 21:43 PAGE 284

R3	= %000003	10591*	10600*	10613*	10614*	10615*	10622*	10631*	10640*	10641	10651	10666*	10672*	10696
		10699*	10709*	10711*	10713*	10719	10720*	10727*	10747	10750*	10753*	10763*	10867*	10903*
		10958*	10959	10960*	10975*	10976*	10993*	10996	10997*	11003*	11004*	11009*	11010*	11156
		11167*	11171*	11174	11181*	11182*	11183	11188*	11196*	11478	11499*	11501*	11502*	11505*
		11506*	11514*	11515	11517*	11522*	11524*	11529*	11534	11536*	11562	11583*		
		2069*	4882*	4893*	4970*	4973*	4976	5518*	5520*	5727*	5603*	5807*	5815*	5902*
		5973*	6125*	6126*	6127*	6128*	6129*	6130*	6131	6134	6533*	6538*	6548*	6550
		6556*	6896*	6900*	7009*	7024	7026*	7075*	7082	7084*	7104	7112*	7781*	7792*
		8017*	8019	8032*	8034	8088*	8090	8105*	8107	8154	8248	8275*	8287	8290*
		8317*	8333	8383	8400*	8459	8503*	8507*	8516	8522	8523*	8526*	8533	8573*
		8574*	8575*	8576	8579*	8580	8606*	8613*	8616*	8617	8660*	8661*	8662	8664*
		8666	8751	8756*	8773*	8777	8779*	8793*	8804*	8808*	8991*	8993	9000*	9101*
		9102*	9103	9136*	9137*	9138	9678	9702*	9752	10104*	10405	10673*	10868*	10902*
		10921*	10924*	10930*	10955*	10968*	10975	11001*	11006*	11012*	11013	11097	11106*	11112*
		11113*	11116*	11121*	11122*	11123	11132*	11157	11165*	11166*	11180*	11183*	11192*	11193*
11195*	11479	11500*	11504*	11535*	11563	11582*								
R4	= %000004	2070*	5557*	5578*	5585*	5589	5591	5780*	5803	5811*	5815	5817*	5888*	5889
		5891*	5894	7010*	7025	7027*	7065*	7071*	7080	7102	7779*	7780	7791*	7794*
		8018*	8026*	8033*	8037*	8089*	8101*	8106*	8175*	8274*	8278*	8288	8316*	8320*
		8398	8459*	8461*	8557*	8587	8589*	8628*	8631*	8633	8663*	8665*	8668	8714
		8715*	8723*	8727	8734*	8753	8754	8781*	8791*	9179*	9238	9254	9262	9266
		9270	9274	9278	9283	9288	9293	9303	9312	9337	9362	9376	9379	9677
		9703*	9751	9776*	9777*	9817*	9818*	9821	9968*	9969*	9998*	10088	10105*	10404
		10413*	10414*	10416	10417	10439	10674*	10872	10892	10917	10928	10954*	10961*	10999*
		11002	11008	11010	11098	11100*	11101*	11102*	11103	11104*	11118	11120*	11128*	11131*
		11564	11581*											
		2071*	5436*	5438	5440*	5442*	5511*	5512*	5513*	5521*	5524*	5525	5527	5530*
		5531*	5532	5534	5536*	5537*	5539	5541*	5542*	5548*	5549*	5550*	5562*	5564*
		5565*	5567*	5572*	5596	5602	5607	5626	5628*	5630	5632*	5633*	5635*	5636
		5638*	5640*	5653	5656	5705*	5709*	5710*	5713*	5725*	5729*	5784*	5788*	5789*
		5792*	5805*	5809*	5873*	5877*	5878*	5881*	5900*	5904*	5959*	5960*	5961*	5962*
5965*	5966*	5974*	5977*	5983*	5986*	5988*	6042*	6045*	6052*	6062*	6063*	6064*		
6067*	6068*	6069*	6070*	6083	6091*	6092*	6099*	6100*	6101*	6106*	6110*	6123*		
6125	6145*	6150*	6367*	6372*	6377*	6393*	6419*	6422*	6430*	6437*	6442*	6447*		
6498*	6503*	6510*	6520*	6521*	6563*	6572*	6573*	6575	6577*	6589*	6592*	6594*		
6596	6598*	6602*	6607*	6614*	6650*	6651*	6665*	6669*	6675*	6680*	6687*	6700		
7029*	7030*	7104*	7106	7110*	7112	7136*	7139*	7141*	7143*	7156	7157	7158*		
7160	7162*	7186*	7195*	7212	7220	7228*	7230*	7231*	7232*	7233*	7234*	7235*		
7248*	7273*	7274*	7288	7300*	7305	7313*	7315*	7326*	7327*	7329*	7330	7340*		
7342*	7344	7354*	7369*	7370*	7774*	7778	7808*	7811*	7812*	7813*	7814*	7815*		
7816*	7896	7897*	7899	7902*	7904*	7905	7907*	7946	7960	7969	8216*	8252*		
8253*	8255*	8260*	8421*	8422*	8423*	8426*	8452	8477	8499*	8500*	8501*	8502*		
8506*	8508*	8509*	8515*	8516	8521*	8522*	8532*	8533	8541	8603	8604	8605		
8607	8609	8611	8653	8654	8655	8657*	8658*	8659*	8662*	8669*	8672*	8688		
8725	8838	8848	8879	8880	8881	8882	8883	8884	8934*	8935*	8984*	8995		
9013*	9015*	9092	9094	9095	9097	9099	9101	9104	9105	9106	9107	9108		
9109	9111	9112	9113	9114	9115	9116	9117	9118	9119	9120	9121	9122		
9158*	9179	9180	9185	9190	9192	9198	9200	9206	9219	9233	9243	9247		
9251	9258	9298	9308	9365	9366	9368	9388	9389	9392	9401	9406	9411		
9416	9424*	9480*	9483*	9484*	9485*	9486*	9487*	9488*	9489*	9524*	9525*	9527		
9530*	9531	9532	9533	9534	9535	9536	9537	9538	9540*	9551	9552	9558*		
9559*	9560*	9563*	9565*	9566*	9569*	9570*	9571*	9580	9583	9584	9590	9612		
9613	9619	9623	9676	9691*	9693*	9694	9704*	9750	9778*	9804*	9806*	9812*		
9821	9824	9840*	9845*	9847*	9848*	9851	9868*	9875	9879*	9897*	9904	9929*		
9933	9934	9935*	9936*	9938*	9939	9972*	9976*	9997*	9998	9999	10003*	10008*		
10009	10013*	10014	10016*	10022*	10023	10030*	10031	10034*	10044*	10048*	10050*	10051*		

	10052*	10053*	10054*	10055*	10056*	10057*	10058*	10070	10074*	10084*	10096*	10106*	10174
	10176	10201*	10277*	10280*	10281*	10282*	10283*	10284*	10285*	10286*	10287*	10289*	10323*
	10324*	10330*	10331*	10337*	10338*	10344*	10345*	10353*	10403	10411*	10413	10415	10428*
	10430	10432	10439*	10441	10445	10457	10464	10465*	10467	10469	10470	10473	10475*
	10476	10479	10482	10485*	10486*	10488*	10489*	10490	10492*	10493	10509	10510	10511
	10512	10513	10515	10517*	10520*	10521*	10522*	10523*	10526	10527	10529*	10530	10531
	10533*	10534	10539	10541	10542*	10543*	10545*	10546	10548*	10549	10552*	10553*	10554
	10558	10560	10564	10566	10568*	10576	10586	10588	10590*	10591	10593*	10594*	10596*
	10597	10599*	10600	10603	10605	10606	10607	10613	10614	10615	10616*	10617*	10619*
	10620	10622	10625	10629	10634	10641*	10642	10651*	10652	10663	10675*	10870	10893
	10914	10926	10953*	10959*	10964*	10965	11000*	11004	11011	11099	11105*	11107*	11109*
	11110*	11111*	11112	11130*	11158	11160*	11162*	11169*	11173*	11188	11194*	11565	11580*
R6 =%000006	2072#	2074	4708*	4709*	4710	6206*	6207*	6208					
R7 =%000007	2073#	2075											
SAVPAR 003200	3695#	7018*	8013	8084	8285	8331							
SAVPRS 005552	3731#	8687	8698										
SAVREG= 104406	6794	6893	7004	7064	7067	7695	7772	7879	7895	7917	7940	7988	8055
	8117	8244	8273	8277	8315	8319	8340	8475	8555	8602	8651	8685	8697
	8789	8861	8920	9038	9090	9152	9509	9550	9611	10951	10992	11619#	
SAVWRD 003202	3696#	6119	6148*	6149*									
SAVXDP 030474	5696	5773	5861	7064#									
SCLR = 000040	3250#	10640											
SCNDRV 031046	4991	7185#											
SCOPE= 104410	5732	5907	11621#										
SCOPE1 055402	11336#	11621											
SCRACH 005534	3725#	4888*	4889*	4890	6495*	6498	6501	6508	6570	6572	6579*	6639*	6643*
	6666	7471*	7473*	7474	7484*	7485*	7486	8059*	8125	8138*	8141	8323*	8354
	8367*	8370											
SECNL1 010473	4389#	5165											
SECNL2 010523	4394#	5169											
SEEK = 000117	3197#	5562	6042	6070	6419	6665	7326	10457	10467				
SELDRV= 000101	3190#	6521	6651	10596									
SELECT 005526	3722#	5778*	5799	5821*	6492*	6567	6580*						
SELP15 010720	4417#	7620											
SERNM 011205	4451#	7777	7819										
SETPRM 017074	5160	5323#	5329	5334	5378								
SETTUP 032060	5500	5686	5722	5763	5851	5897	5952	6035	7365#				
SETUPM 037022	5701	5779	5869	8421#									
SIZMEM 027302	4785	6794#											
SKI = 000002	3266#	9283											
SOFTBS 011233	4455#	8591											
SP =%000006	2074#	4712*	4730*	4738*	4742	4811*	4814	4816*	4819	4821*	4822*	4823*	4824*
	4825*	4828	4862*	5115*	5118	5289*	5292	5369*	5372	5409*	5412	5421*	5597*
	5602*	5608*	5615*	5622*	6184*	6210*	6228*	6236*	6240	6280	6703*	6704*	6713
	6714	6722*	6735	6821*	6837*	6853*	6855*	6858	6861*	6875*	6878	6923	6934*
	6935*	6936*	6942	6945	6946	6947	6955*	6956*	6957*	6959	6976	6977	6978
	6982	6990	7155*	7156*	7162	7163	7209*	7250*	7302*	7318*	7365	7366*	7387*
	7401*	7402	7431*	7432*	7437	7442*	7448*	7454*	7498*	7499	7500	7516*	7517*
	7518*	7523*	7548*	7568*	7569*	7574*	7597*	7598*	7599*	7646*	7649	7666	7667
	7668	7671*	7674*	7702*	7733*	7737*	7738*	7780*	7820*	7836*	7837*	7852	7853
	7896*	7907	7941*	7942*	7943*	7944*	7945*	7946*	7947*	7948*	7953	7955*	7956*
	7958	7959*	7960*	7961*	7962*	7963	7965*	7966*	7968	7969*	7970	7971	7993*
	8008*	8018	8033	8040	8064*	8079*	8089	8106	8182	8215*	8466	8570*	8576*
	8580*	8714*	8734	8752	8774*	8775*	8777*	8780*	8793	8802	8810	8952*	8965*
	8975*	8978*	8998*	9676*	9677*	9678*	9679*	9680*	9700	9701	9702	9703	9704
	9750*	9751*	9752*	9753*	9754*	9755*	10101	10102	10103	10104	10105	10106	10172*

SW03 = 000010	2100#	2110				
SW04 = 000020	2099#	2109				
SW05 = 000040	2098#	2108				
SW06 = 000100	2097#	2107				
SW07 = 000200	2096#	2106				
SW08 = 000400	2095#	2105				
SW09 = 001000	2094#	2104				
SW1 = J00002	2112#					
SW10 = 002000	2093#					
SW11 = 004000	2092#					
SW12 = 010000	2091#					
SW13 = 020000	2090#	8930				
SW14 = 040000	2089#					
SW15 = 100000	2088#					
SW2 = 000004	2111#					
SW3 = 000010	2110#					
SW4 = 000020	2109#					
SW5 = 000040	2108#					
SW6 = 000100	2107#					
SW7 = 000200	2106#					
SW8 = 000400	2105#					
SW9 = 001000	2104#					
S.ACLO= 000100	3335#					
S.BRHM= 000100	3350#					
S.BRKE= 040000	3372#					
S.CART= 000400	3352#					
S.DCLO= 010000	3342#					
S.DIB = 002000	3368#					
S.DOOR= 000200	3351#					
S.DRA = 000040	3321#					
S.DROT= 020000	3343#					
S.DRY = 000200	3323#	7212	7288			
S.DSC = 040000	3330#	9866	10014	10031	10094	
S.FLT = 000200	3336#	10009				
S.FORM= 001000	3325#					
S.FWD = 002000	3354#					
S.HDFL= 000200	3365#					
S.HDHM= 000040	3349#	6575	6596	7344		
S.ICYL= 000040	3334#					
S.ILF = 000400	3337#					
S.LIMD= 020000	3371#					
S.LOAD= 010000	3356#					
S.MHD = 000400	3366#					
S.NMOV= 010000	3370#					
S.OFF = 002000	3326#					
S.PAR = 001000	3338#	9869	10346			
S.PIP = 020000	3329#	9877	10072			
S.PLO = 004000	3369#					
S.REV = 004000	3355#					
S.RTZ = 020000	3357#					
S.SECT= 000020	3362#					
S.SKI = 002000	3339#					
S.SPIN= 010000	3328#					
S.SPLS= 010000	3341#					
S.SPOK= 001000	3353#					
S.TYPE= 000400	3324#					

ADC	6813	7027	7758	8161	8390	8486	8667	10756	10759	10871	10898	10910	10911	10915	10916
ADD	10918	10927	11009												
	4964	5056	5058	5099	5101	5126	5131	5206	5207	5209	5231	5285	5300	5302	5341
	5343	5368	5524	5578	6812	6899	6991	7026	7047	7116	7158	7250	7318	7401	7448
	7454	7498	7632	7659	7737	7738	7757	7900	7947	7961	7969	8007	8022	8078	8096
	8160	8162	8295	8389	8391	8405	8430	8432	8485	8487	8515	8532	8661	8662	8666
	8668	8755	8757	8762	8941	9023	9530	9621	10412	10431	10433	10715	10723	10755	10757
	10758	10760	10805	10870	10872	10897	10899	10909	10912	10914	10917	10926	10928	10931	10934
ASL	10976	11008	11010	11092	11102	11173	11420	11432	11444	11503	11514	11523	11524		
	4822	4823	4824	4893	4979	5891	6710	7398	7697	8002	8003	8004	8005	8006	8073
	8074	8075	8076	8077	8938	8939	8940	10708	10710	10712	10751	11599			
ASLB	11178														
ASR	4834	4835	4836	7517	7569	7842	7843	8000	8071	9554	9555	9556	9557	9615	9616
	9617	10968	11427												
BCC	10869	11179													
BCE	10900	10925													
BEQ	4746	4765	4787	4871	4884	4886	4909	4917	4949	4977	4984	4989	5048	5071	5073
	5106	5123	5195	5220	5254	5256	5340	5350	5403	5413	5430	5439	5449	5464	5528
	5535	5540	5588	5627	5631	5637	5645	5738	5822	5826	5892	5914	5964	5993	6000
	6047	6049	6051	6066	6086	6088	6090	6114	6116	6118	6132	6188	6244	6282	6290
	6293	6311	6314	6335	6344	6363	6464	6470	6486	6497	6502	6509	6524	6532	6544
	6557	6568	6571	6576	6597	6630	6672	6708	6817	6904	7020	7041	7070	7073	7114
	7202	7221	7238	7281	7345	7349	7351	7400	7466	7477	7482	7605	7613	7615	7634
	7637	7654	7657	7699	7701	7760	7810	7839	7846	7951	8021	8025	8036	8091	8095
	8128	8148	8169	8174	8201	8203	8208	8294	8334	8357	8377	8404	8408	8429	8460
	8505	8514	8525	8531	8612	8630	8693	8719	8722	8728	8761	8770	8796	8803	8809
	8843	8845	8853	8872	8899	8903	8924	8931	8943	8949	8988	8994	9001	9005	9043
	9047	9051	9053	9058	9062	9066	9070	9073	9076	9079	9157	9181	9186	9191	9193
	9199	9201	9207	9220	9234	9239	9244	9248	9252	9255	9259	9263	9267	9271	9275
	9279	9284	9289	9294	9299	9304	9309	9313	9316	9319	9338	9341	9344	9363	9377
	9380	9393	9402	9407	9412	9417	9436	9442	9464	9528	9562	9587	9591	9596	9624
	9685	9759	9765	9775	9796	9822	9860	9867	9870	9878	9892	9906	9951	9962	10000
	10010	10024	10033	10065	10073	10089	10095	10124	10184	10186	10347	10442	10491	10497	10503
	10516	10528	10532	10550	10555	10577	10598	106	10630	10635	10643	10653	10701	10703	10795
	10808	10843	10907	10920	10963	11054	11056	11114	11224	11227	11249	11252	11337	11339	11360
	11376	11380	11414	11418	11438	11440									
BGE	4938	5331	5358	5363	5405	7869	8764	11390							
BGT	4832	4946	5336	5366	6179	6318	6383	6726	10705	10923	10962	11126	11187	11294	
BHI	5120	7708	7721	7724	7954	7964	10452	11378	11516						
BHIS	6972	7045													
BIC	4801	4921	4962	5061	5214	5234	5579	5712	5791	5880	6130	6172	6176	6319	6384
	6429	6541	6556	6560	6692	6712	6724	6729	6734	6816	7007	7019	7029	7100	7115
	7208	7295	7789	7867	7870	8015	8023	8029	8086	8097	8181	8283	8296	8301	8329
	8406	8412	8431	8671	8768	8775	8799	9102	9137	9327	9352	9367	9369	9374	9394
	9553	9614	9620	9628	9629	9777	9818	9936	9969	9976	10003	10044	10048	10049	10277
	10279	10414	10428	10486	10489	10492	10521	10533	10543	10594	10599	10617	10714	10975	11116
	11281	11287	11295	11520											
BICB	5440	7030	9563	9569	10177	10523									
BIS	4889	5966	6069	6415	6518	6582	6662	6767	7101	7790	8014	8016	8085	8087	8284
	8286	8330	8332	8807	9183	9188	9236	9241	9281	9286	9291	9296	9301	9306	9314
	9339	9364	9378	9381	9395	9404	9409	9414	9419	9433	9443	9465	9477	9497	9529
	9622	9625	9630	9693	9760	9798	9806	9840	9861	9868	9871	9879	9942	9964	9987
	9993	10016	10025	10034	10050	10066	10074	10084	10096	10135	10187	10201	10209	10280	10348
	10353	10465	10529	10568	10582	10590	10636	10644	10657	11012	11121	11122	11181	11182	11497
BISB	5442	5450	5457	5713	5792	5881	6397	6618	7204	7215	7223	7243	7283	7291	7308
	8672	8897	9039	9153	9496	9499	9565	9570							

BIT	4976	5212	5448	5463	5545	5582	5587	5593	5653	5656	5889	5981	6104	6142	6550
	6575	6596	6700	7113	7212	7220	7237	7288	7305	7344	7604	7713	7838	7997	8020
	8068	8094	8123	8139	8166	8256	8293	8346	8368	8395	8403	8541	8718	8721	8898
	8922	8930	8957	8985	9042	9046	9050	9057	9061	9065	9069	9072	9075	9078	9156
	9180	9185	9190	9192	9198	9200	9206	9209	9211	9219	9233	9238	9243	9247	9251
	9254	9258	9262	9266	9270	9274	9278	9283	9288	9293	9298	9303	9308	9312	9318
	9324	9331	9337	9343	9349	9356	9362	9371	9376	9379	9392	9401	9406	9411	9416
	9427	9431	9435	9441	9463	9470	9527	9590	9623	9758	9773	9783	9788	9794	9832
	9851	9866	9869	9875	9877	9891	9918	9950	9954	9960	9980	9999	10009	10014	10031
	10070	10072	10094	10132	10183	10185	10192	10194	10196	10346	10490	10496	10498	10515	10527
	10531	10549	10554	10558	10560	10564	10566	10576	10597	10620	10629	10642	10652	10663	11226
	11233	11248	11338	11361	11379	11386									
BITB	4745	4794	4971	4983	5438	6119	6243	9561	9835	9859	9905	9991	10023	10064	10088
	10441	10502	10607	10794	10799	10831	11417								
BLE	5164	5168	5173	6850	8425	8534	8634	8638	9008						
BLO	6993	7706	7716	7719	10458										
BLOS	10437														
BLT	4830	4944	5114	5288	6316	6381	6728	7645	8142	8371	8586	8926	10707	10822	11005
	11127	11170	11186	11292											
BMI	5522	5704	5783	5872	5970	6798	7079	7753	8349	8759	9766	11177			
BNE	4697	4711	4735	4761	4780	4791	4795	4873	4895	4897	4904	4907	4919	4932	4934
	4951	4966	4972	4982	4997	5027	5032	5035	5038	5050	5053	5060	5082	5084	5098
	5103	5109	5128	5148	5153	5156	5159	5162	5197	5200	5205	5211	5213	5222	5225
	5233	5265	5267	5282	5284	5304	5345	5352	5355	5360	5426	5460	5508	5516	5526
	5533	5546	5556	5575	5581	5583	5594	5654	5657	5707	5786	5800	5866	5875	5890
	5980	5982	5991	6058	6078	6081	6105	6120	6143	6209	6233	6273	6277	6295	6298
	6338	6341	6346	6349	6366	6371	6376	6391	6426	6428	6436	6441	6446	6467	6472
	6475	6500	6507	6537	6539	6551	6591	6604	6606	6610	6632	6635	6638	6642	6656
	6674	6679	6686	6691	6701	6732	6738	6742	6746	6810	6847	6860	6874	6901	6944
	6970	6974	7081	7086	7103	7118	7161	7188	7190	7192	7213	7241	7289	7306	7331
	7389	7391	7393	7440	7446	7452	7458	7464	7468	7490	7546	7602	7631	7640	7651
	7714	7717	7722	7793	7849	7851	7883	7906	7924	7991	7998	8027	8038	8062	8069
	8099	8102	8108	8124	8126	8140	8144	8146	8167	8176	8199	8205	8221	8225	8231
	8250	8257	8289	8298	8347	8355	8369	8373	8375	8396	8399	8458	8462	8478	8517
	8542	8566	8588	8608	8610	8703	8726	8794	8805	8958	8969	8986	8996	9011	9016
	9022	9210	9212	9222	9325	9332	9350	9357	9372	9386	9397	9428	9432	9469	9471
	9473	9682	9689	9696	9785	9790	9813	9825	9833	9836	9853	9876	9919	9921	9956
	9981	9986	9992	10015	10071	10193	10195	10197	10200	10468	10474	10480	10483	10499	10514
	10540	10559	10561	10565	10567	10587	10604	10621	10626	10655	10664	10754	10793	10800	10802
	10810	10818	10832	10839	10874	11016	11117	11175	11234	11239	11256	11283	11289	11316	11362
	11387	11416	11422	11425	11442	11527									
BPL	5612	5619	5715	5794	5883	6986	8011	8082	8171	8180	8282	8292	8300	8328	8402
	8411	8564	8674	8732	10181	10787	10836	11115	11161	11191	11246	11279	11285	11494	
BR	4699	4737	4839	4842	4877	4941	4953	4967	5000	5030	5076	5132	5135	5151	5166
	5171	5176	5259	5286	5308	5334	5346	5378	5408	5441	5538	5543	5586	5590	5592
	5614	5621	5629	5639	5735	5813	5819	5823	5893	5908	5911	6056	6097	6138	6235
	6301	6321	6323	6352	6386	6452	6477	6479	6488	6528	6555	6612	6640	6644	6646
	6711	6740	6744	6748	6751	6801	6922	6941	6989	6994	7043	7066	7083	7108	7119
	7194	7218	7226	7246	7286	7311	7353	7395	7443	7449	7455	7461	7475	7495	7618
	7635	7660	7664	7672	7675	7703	7711	7844	7957	7967	7995	8001	8028	8030	8039
	8066	8072	8093	8103	8133	8223	8229	8276	8318	8352	8362	8465	8467	8480	8535
	8569	8592	8615	8696	8765	8772	8776	8798	8806	8811	8901	9003	9014	9024	9045
	9049	9056	9060	9064	9068	9375	9383	9434	9466	9478	9564	9589	9593	9598	9768
	9843	10011	10017	10035	10040	10092	10189	10471	10477	10518	10610	10632	10638	10647	10660
	10716	10789	10815	10825	10834	10841	10913	10977	11007	11093	11108	11129	11172	11189	11244
	11290	11311	11364	11370	11373	11382	11385	11408	11430	11509	11519				

	5226	5235	5269	5270	5305	5427	5502	5634	5649	5658	5660	5688	5765	5853	5954
	6001	6037	6059	6153	6170	6195	6296	6299	6339	6342	6347	6350	6369	6374	6379
	6388	6400	6407	6433	6439	6444	6449	6468	6473	6505	6512	6565	6600	6621	6633
	6636	6677	6682	6694	6702	6715	6927	6975	8100	8109	8172	8177	8178	8219	8409
	8520	8543	8545	8929	8932	9083	9184	9189	9195	9197	9203	9205	9214	9216	9218
	9224	9237	9242	9246	9250	9257	9261	9265	9269	9273	9277	9292	9287	9292	9297
	9302	9307	9311	9317	9321	9334	9336	9342	9346	9359	9361	9391	9445	9454	9762
	9771	9780	9800	9810	9815	9826	9838	9864	9873	9882	9885	9902	9910	9931	9940
	9944	9966	9978	9982	9989	9995	10001	10028	10068	10077	10080	10443	10507	10535	10570
JSR	10580	10584	10601	10623	11323										
	4754	4782	4785	4788	4813	4818	4827	4864	4866	4912	4923	4927	4991	5022	5044
	5045	5055	5066	5089	5093	5116	5140	5143	5191	5192	5202	52J3	5227	5228	5229
	5249	5273	5277	5290	5295	5298	5326	5370	5374	5377	5398	5410	5435	5437	5443
	5454	5455	5461	5462	5465	5500	5501	5504	5505	5519	5523	5529	5563	5566	5570
	5573	5641	5686	5687	5690	5691	5693	5696	5701	5702	5719	5722	5723	5724	5726
	5728	5730	5731	5734	5736	5763	5764	5767	5768	5770	5773	5779	5781	5798	5802
	5804	5806	5808	5810	5812	5816	5818	5824	5851	5852	5855	5856	5858	5861	5864
	5869	5870	5887	5897	5898	5899	5901	5903	5905	5906	5910	5912	5952	5953	5956
	5957	5978	5984	5989	5994	6035	6036	6039	6040	6053	6061	6071	6076	6093	6094
	6103	6107	6121	6124	6135	6144	6146	6152	6190	6288	6300	6309	6322	6324	6326
	6328	6333	6351	6361	6366	6373	6378	6387	6392	6394	6413	6420	6421	6423	6431
	6438	6443	6448	6450	6458	6462	6478	6490	6504	6511	6513	6514	6522	6564	6574
	6578	6585	6593	6595	6599	6608	6615	6626	6627	6645	6653	6658	6667	6668	6670
	6676	6681	6683	6684	6688	6796	6822	6823	6842	6856	6877	6915	6958	6979	7005
	7090	7138	7140	7185	7198	7229	7236	7249	7277	7301	7314	7316	7328	7341	7343
	7347	7352	7355	7371	7460	7549	7550	7608	7622	7626	7647	7729	7734	7735	7773
	7775	7795	7807	7817	7821	7822	7840	7865	7901	7903	7920	8060	8116	8121	8209
	8217	8232	8233	8258	8261	8326	8339	8344	8449	8451	8454	8456	8498	8512	8518
	8529	8614	8652	8656	8675	8717	8724	8729	8730	8733	8791	8833	8834	8837	8839
	8841	8849	8851	9041	9161	9208	9322	9323	9347	9348	9370	9387	9398	9439	9481
	9491	9526	9568	9699	9761	9799	9809	9837	9842	9846	9849	9854	9857	9863	9872
	9884	9898	9899	9901	9927	9930	9943	9965	9988	9994	10006	10020	10027	10039	10059
	10062	10067	10079	10086	10091	10099	10136	10182	10188	10202	10211	10220	10328	10335	10342
	10349	10360	10364	10370	10494	10500	10506	10547	10551	10562	10569	10572	10579	10583	10627
	10637	10646	10659	10667	10797	10816	10823	10830	11235	11241	11434				
MOV	4698	4705	4708	4712	4714	4715	4716	4717	4718	4719	4720	4721	4722	4726	4727
	4730	4731	4732	4733	4738	4740	4741	4742	4747	4750	4752	4753	4767	4768	4769
	4770	4771	4772	4773	4775	4776	4777	4783	4784	4792	4793	4796	4797	4802	4811
	4814	4816	4819	4821	4828	4837	4845	4862	4882	4888	4955	4970	4973	4974	4990
	5057	5078	5079	5080	5100	5115	5118	5121	5125	5130	5208	5261	5262	5263	5268
	5289	5292	5293	5301	5342	5348	5369	5372	5373	5409	5412	5417	5420	5421	5428
	5433	5445	5446	5451	5452	5453	5499	5510	5511	5514	5518	5520	5541	5549	5552
	5553	5554	5559	5560	5568	5571	5576	5584	5589	5591	5597	5602	5608	5615	5622
	5632	5638	5642	5643	5646	5647	5685	5694	5699	5709	5710	5711	5717	5718	5721
	5762	5771	5776	5780	5788	5789	5790	5796	5797	5803	5815	5850	5859	5865	5867
	5877	5878	5879	5835	5886	5888	5894	5896	5951	5959	5962	5965	5968	5971	5972
	5973	5974	5986	5995	5996	6034	6062	6064	6067	6068	6074	6075	6077	6082	6083
	6092	6101	6102	6111	6125	6133	6134	6145	6148	6151	6180	6184	6187	6203	6206
	6210	6212	6213	6214	6215	6216	6217	6218	6219	6220	6224	6225	6228	6229	6230
	6231	6236	6238	6239	6240	6245	6252	6254	6255	6256	6257	6258	6259	6260	6261
	6264	6265	6269	6270	6291	6312	6320	6336	6364	6385	6396	6414	6418	6422	6434
	6465	6493	6494	6495	6516	6517	6529	6533	6534	6542	6546	6548	6549	6580	6581
	6588	6617	6639	6643	6649	6652	6654	6661	6666	6669	6703	6704	6705	6713	6714
	6722	6723	6730	6735	6795	6799	6803	6805	6814	6815	6821	6837	685	6858	6861
	6875	6878	6895	6896	6897	6898	6902	6905	6906	6916	6917	6918	6919	6920	6921
	6934	6935	6936	6937	6938	6940	6945	6946	6947	6955	6956	6957	6959	6960	6961

6963	6964	6965	6966	6976	6977	6978	6983	6984	6985	6987	7006	7008	7009	7010
7018	7022	7023	7024	7025	7042	7046	7071	7074	7075	7076	7077	7082	7084	7098
7099	7104	7106	7110	7112	7133	7134	7135	7136	7155	7156	7157	7162	7163	7196
7197	7200	7209	7231	7233	7234	7275	7276	7279	7302	7365	7366	7369	7372	7387
7394	7396	7399	7402	7431	7432	7442	7499	7500	7516	7523	7541	7543	7547	7548
7568	7574	7597	7598	7599	7646	7649	7652	7658	7666	7667	7658	7671	7674	7696
7702	7710	7726	7727	7731	7732	7733	7748	7749	7762	7778	7779	7780	7781	7798
7813	7815	7816	7820	7836	7837	7841	7845	7847	7852	7853	7866	7880	7881	7896
7907	7918	7919	7921	7922	7941	7949	7952	7970	7989	7992	7993	7994	8008	8009
8013	8017	8018	8019	8032	8033	8034	8057	8058	8063	8064	8065	8079	8090	8099
8088	8089	8092	8105	8106	8111	8113	8115	8118	8129	8130	8135	8149	8154	8155
8156	8163	8206	8214	8215	8222	8228	8246	8247	8248	8252	8253	8275	8278	8279
8280	8285	8287	8317	8320	8321	8322	8325	8331	8336	8338	8341	8351	8353	8358
8359	8364	8378	8383	8384	8385	8392	8421	8426	8427	8452	8453	8459	8468	8477
8501	8503	8508	8509	8510	8521	8523	8526	8527	8536	8558	8570	8573	8576	8580
8590	8603	8617	8628	8631	8653	8657	8660	8669	8670	8686	8687	8698	8690	8694
8695	8698	8699	8700	8701	8714	8734	8750	8751	8752	8753	8754	8767	8774	8777
8778	8779	8780	8790	8793	8797	8838	8848	8862	8863	8865	8866	8867	8869	8869
8870	8873	8874	8875	8876	8878	8879	8880	8881	8882	8883	8884	8942	8948	8952
8963	8964	8965	8975	8978	8987	8989	8990	8998	9009	9093	9094	9099	9100	9103
9104	9105	9106	9107	9108	9109	9111	9112	9113	9114	9115	9116	9117	9118	9119
9120	9121	9122	9124	9125	9130	9135	9138	9139	9158	9159	9160	9179	9188	9189
9326	9328	9329	9330	9351	9353	9354	9355	9365	9368	9384	9388	9389	9424	9425
9426	9430	9438	9440	9479	9482	9483	9484	9485	9486	9487	9488	9489	9490	9510
9511	9512	9513	9514	9515	9516	9517	9518	9519	9520	9524	9531	9532	9533	9534
9535	9536	9537	9538	9540	9551	9552	9558	9559	9560	9567	9571	9580	9592	9594
9599	9612	9613	9619	9626	9627	9676	9677	9678	9679	9680	9683	9686	9687	9691
9700	9701	9702	9703	9704	9750	9751	9752	9753	9754	9755	9756	9757	9776	9778
9779	9782	9791	9804	9812	9817	9831	9834	9845	9847	9848	9881	9889	9897	9904
9912	9913	9914	9915	9916	9917	9925	9929	9933	9934	9939	9948	9957	9968	9972
9984	9997	10008	10013	10022	10030	10051	10052	10053	10054	10055	10056	10057	10058	10076
10090	10101	10102	10103	10104	10105	10106	10130	10131	10137	10140	10172	10175	10179	10204
10213	10232	10233	10234	10235	10236	10237	10238	10239	10240	10241	10242	10243	10244	10245
10281	10282	10283	10284	10285	10286	10287	10289	10318	10323	10324	10325	10330	10331	10332
10337	10338	10344	10345	10350	10403	10404	10405	10406	10407	10408	10409	10410	10411	10413
10415	10418	10420	10430	10432	10439	10464	10469	10470	10476	10481	10484	10493	10495	10501
10509	10510	10511	10512	10517	10526	10530	10534	10541	10546	10548	10552	10553	10588	10591
10600	10605	10606	10609	10612	10613	10614	10615	10622	10640	10641	10649	10651	10666	10669
10670	10671	10672	10673	10674	10675	10694	10695	10696	10697	10718	10719	10720	10721	10722
10727	10728	10729	10730	10745	10746	10747	10748	10749	10750	10761	10762	10763	10764	10765
10790	10791	10796	10804	10819	10864	10891	10892	10893	10952	10953	10954	10955	10956	10957
10960	10965	10993	10994	10995	10996	10997	10998	10999	11000	11051	11052	11058	11061	11062
11089	11097	11098	11099	11105	11112	11130	11131	11132	11133	11134	11154	11155	11156	11157
11158	11159	11160	11165	11168	11188	11194	11195	11196	11197	11198	11200	11201	11225	11230
11250	11253	11276	11277	11309	11317	11318	11319	11320	11340	11366	11367	11369	11372	11381
11391	11392	11395	11396	11399	11400	11411	11412	11419	11423	11428	11429	11431	11433	11443
11449	11450	11476	11477	11478	11479	11480	11481	11482	11484	11485	11487	11488	11490	11491
11495	11499	11500	11501	11502	11505	11507	11508	11512	11517	11521	11530	11531	11532	11533
11534	11535	11536	11537	11538	11560	11561	11562	11563	11564	11565	11566	11567	11568	11569
11576	11577	11578	11579	11580	11581	11582	11583	11584	11585	11595	11596	11600		
4702	4725	4756	4798	4961	4963	4978	5416	5436	5456	5512	5536	5542	5548	5550
5562	5564	5565	5567	5572	5596	5607	5628	5633	5640	5697	5698	5725	5729	5774
5775	5805	5809	5862	5863	5900	5904	5960	5976	5977	5983	5988	6042	6052	6070
6091	6099	6100	6106	6109	6110	6123	6150	6223	6280	6367	6372	6377	6393	6398
6405	6416	6419	6430	6437	6442	6447	6476	6498	6503	6510	6520	6521	6563	6572
6573	6577	6579	6583	6587	6589	6592	6594	6598	6602	6607	6614	6619	6650	6651

MOVE

	6663	6665	6675	6680	6687	6818	6924	6967	7139	7141	7142	7143	7186	7195	7203
	7214	7222	7228	7230	7232	7242	7248	7273	7274	7282	7284	7290	7292	7300	7307
	7309	7313	7315	7326	7340	7342	7354	7370	7437	7473	7485	7487	7492	7497	7774
	7791	7812	7814	7899	7902	7944	7946	7956	7960	7968	8056	8119	8120	8150	8151
	8216	8255	8260	8324	8342	8343	8379	8380	8422	8423	8500	8502	8522	8579	8604
	8605	8654	8655	8658	8659	8720	8905	8936	8991	9092	9095	9097	9101	9126	9128
	9131	9133	9136	9228	9229	9230	9366	9437	9475	9480	9525	9566	9583	9584	9856
	9880	9926	9935	9938	9998	10005	10019	10061	10075	10085	10141	10174	10176	10320	10327
	10334	10341	10416	10417	10475	10485	10488	10520	10522	10542	10545	10592	10593	10596	10616
	10619	10665	10700	10801	10829	10837	10959	11013	11090	11091	11094	11095	11096	11100	11103
	11104	11123	11163	11166	11180	11183	11192	11232	11240	11280	11286	11394	11398	11406	11407
	11409	11598													
NEG	5708	5787	5876	7942	8153	8382	8481	8689	10894	10895	11101	11162			
NOP	6191	6192	6193												
RESET	4749	6189	6248	8210	9453	11322									
ROL	6527	6554	6807	6808	7011	7012	7013	7014	7016	7091	7092	7093	7094	7096	7782
	7783	7784	7785	7796	7787	8158	8159	8387	8388	8483	8484	8664	8665	10709	10711
ROR	10713	10752	10902	10903	10904	10905	10930	11107	11109	11110	11111	11113			
	6126	6127	6128	6129	7755	7756	10865	10866	10867	10868	10924	10969	10970	10971	10972
	10973	10974													
RORB	7017	7097													
RTI	4739	6237	6736	6980	10107	10806	11135	11202	11259	11296	11341	11401	11489	11570	11586
RTS	6768	6783	6827	6848	6854	6862	6879	6908	6948	7032	7049	7088	7144	7164	7210
	7251	7303	7319	7332	7356	7373	7403	7501	7527	7551	7578	7669	7740	7763	7798
	7824	7854	7871	7885	7909	7926	7973	8042	8184	8234	8263	8303	8414	8434	8469
	8489	8537	8594	8619	8641	8677	8705	8735	8781	8801	8854	8886	8909	9026	9141
	9232	9493	9498	9500	9542	9573	9601	9632	9705	10138	10143	10207	10214	10246	10290
	10351	10355	10361	10365	10371	10676	10724	10731	10766	10846	10876	10933	10936	10967	11019
	11063	11451	11539	11601											
SBC	7751	10896	11003												
SEV	10935														
SOB	7028	11504													
SLB	5364	6853	7048	7750	7752	7955	7965	8114	8164	8337	8393	11002	11004	11169	11231
	11426	11528	11529												
SWAB	4800	4825	4833	6149	7015	7095	8575	8616							
TRAP	11496	11603	11612	11613	11614	11615	11618	11619	11620	11621					
TST	4870	4885	4896	4916	4931	5026	5047	5070	5097	5122	5147	5194	5219	5253	5281
	5402	5447	5459	5611	5618	5703	5706	5714	5782	5785	5793	5799	5871	5874	5882
	5969	6057	6289	6310	6334	6362	6425	6463	6496	6536	6543	6629	6671	6797	6846
	6859	6969	7040	7078	7080	7102	7348	7392	7465	7467	7476	7612	7630	7650	7653
	7700	7759	7971	7990	8010	8040	8061	8081	8125	8168	8170	8179	8182	8200	8281
	8288	8291	8299	8327	8348	8354	8398	8401	8410	8463	8563	8565	8583	8587	8673
	8692	8725	8727	8731	8758	8766	8771	8810	8993	8995	9004	9585	9595	10142	10205
	10206	10212	10354	10656	10662	10717	10726	10803	10811	10833	10875	10906	10919	11118	11174
	11184	11245	11251	11359	11368	11421	11439	11441	11496	11513	11525	11597			
TSTB	4760	4764	4786	4790	4883	4903	4908	4933	4965	4988	5083	5266	5429	5507	5574
	5644	5737	5825	5913	5963	5979	5990	6048	6065	6087	6115	6276	6390	6485	6590
	6605	6903	6973	7069	7072	7191	7388	7390	7809	7950	8127	8202	8207	8356	8428
	8457	8504	8513	8524	8530	8629	8842	8844	8852	8871	8902	8992	9010	9021	9052
	9221	9315	9340	9385	9396	9468	9684	9764	9985	10180	10786	10835	11053	11176	11190
	11278	11284	11336	11375	11413	11424	11437	11492							
.ASCII	2333	2334	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	4302	4328
	4332	4359	4363	4367	4389	4443	4455	4465	4472	4514	4516	4533	4549	4551	4585
	4592	4623	4642	4647	4657	4676	4677	4678	4679	12375	12380	12387	12395	12400	12405
	12410	12415	12427	12436	12445	12454	12463	12473	12482	12484	12493	12501	12510	12519	12529
	12538	12547	12553	12562	12571	12580	12588	12594	12601	12610					

.ASCIZ	2332	2335	4234	4236	4244	4245	4246	4251	4256	4261	4264	4267	4270	4272	4274
	4277	4279	4292	4284	4287	4292	4307	4311	4314	4319	4336	4342	4346	4354	4372
	4375	4379	4385	4391	4394	4396	4398	4403	4412	4417	4421	4430	4437	4446	4447
	4449	4451	4453	4457	4460	4478	4486	4493	4498	4504	4509	4524	4535	4545	4560
	4566	4577	4594	4595	4601	4606	4613	4618	4624	4627	4632	4637	4644	4652	4660
	4662	4668	4669	4670	4671	4672	4673	4674	4675	4680	4682	4693	4684	4685	4686
	6198	11297	11298	11299	11301	11325	11623	11625	11626	11630	11634	11638	11641	11644	11648
	11653	11655	11657	11660	11664	11668	11671	11674	11677	11680	11683	11687	11692	11695	11699
	11701	11705	11709	11712	11715	11719	11722	11727	11732	11738	11745	11751	11756	11761	11765
	11769	11772	11777	11784	11788	11795	11801	11805	11810	11816	11823	11829	11836	11841	11844
	11848	11857	11860	11864	11872	11879	11883	11886	11891	11896	11903	11908	11911	11916	11920
	11923	11926	11929	11933	11937	11943	11951	11961	11964	11968	11971	11974	11984	11987	11997
	12000	12007	12012	12016	12019	12022	12027	12031	12034	12044	12046	12050	12053	12055	12057
	12062	12065	12068	12074	12080	12088	12422	12618							
.BLKB	10978	11040													
.BLKW	3697	3698	3699	3700	3701	3706	3728	3731	3741	11207					
.BYTE	2259	2260	2265	2266	2274	2275	2283	2284	2285	2286	2351	2352	2362	2363	2370
	2371	2373	2374	2376	2377	3481	3482	3484	3485	3486	3487	3513	3514	3516	3517
	3518	3519	3615	3616	3617	3621	3625	3626	3627	3628	3629	3630	3631	3632	3644
	3645	3646	3647	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659
	3660	3661	3662	3663	3664	3665	3666	3667	3669	3670	3734	3735	3747	3748	3749
	3750	3751	3752	3753	3754	4484	4485	5599	5600	5604	5605	6197	6839	6840	7520
	7521	7525	7526	7571	7572	7576	7577	8954	8955	11136	11137	11138	11139	11242	11243
	11452	11453	11454	12111	12112	12114	12116	12118	12120	12122	12125	12127	12129	12131	12133
	12135	12137	12140	12142	12144	12146	12148	12150	12152	12154	12157	12159	12161	12164	12166
	12168	12170	12172	12174	12176	12179	12181	12183	12185	12187	12189	12191	12194	12196	12198
	12200	12202	12205	12207	12209	12211	12214	12216	12218	12220	12222	12224	12227	12229	12231
	12233	12235	12237	12240	12242	12244	12246	12248	12250	12253	12255	12257	12259	12261	12263
	12265	12267	12269	12272	12274	12276	12278	12280	12282	12284	12286	12288	12291	12293	12295
	12297	12299	12302	12304	12307	12309	12312	12314	12316	12318	12320	12322	12325	12327	12330
	12333	12336	12338	12341	12343	12345	12347	12349	12352	12354	12356	12358	12360	12363	12365
	12367	12369	12371												
.DSABL	11265														
.ENABL	1997	11263													
.END	12629														
.ENDC	2005	2007	2010	2011	2013	2018	2023	2031	2033	2034	2035	2038	2044	2045	2050
	2142	2156	2168	2179	2190	2209	2210	2212	2217	2221	2223	2225	2228	2230	2237
	2247	2253	2257	2259	2287	2321	2330	2331	2332	2333	2337	2340	2362	2370	2373
	2376	2379	2380	2381	2382	2385	3181	3482	3514	3612	3643	3697	3698	3741	3761
	3767	3772	3778	3787	3789	3793	3794	3796	3802	3804	3808	3809	3811	3820	3824
	3832	3834	3836	3838	3845	3847	3851	3852	3854	4234	4236	4463	4662	4689	4701
	4712	4713	4716	4718	4720	4722	4723	4724	4726	4728	4749	4757	4760	4785	4852
	4860	5004	5018	5177	5178	5467	5470	5471	5472	5497	5498	5499	5500	5503	5650
	5665	5666	5683	5684	5685	5686	5689	5745	5746	5760	5761	5762	5763	5766	5833
	5834	5848	5849	5850	5851	5854	5921	5922	5949	5950	5951	5952	5955	6007	6008
	6032	6033	6034	6035	6038	6159	6162	6163	6164	6166	6173	6179	6182	6183	6187
	6189	6195	6197	6198	6201	6210	6211	6214	6216	6218	6220	6221	6222	6224	6226
	6247	6695	6719	6721	6756	6764	6773	6780	6788	6794	6832	6837	6867	6873	6884
	6893	6913	6915	6932	6934	6953	6955	6999	7004	7037	7040	7053	7054	7064	7124
	7132	7149	7154	7169	7184	7256	7272	7324	7326	7337	7340	7360	7361	7365	7378
	7408	7429	7506	7514	7531	7540	7556	7565	7583	7595	7680	7694	7745	7748	7768
	7772	7803	7807	7828	7829	7835	7859	7865	7875	7876	7879	7890	7895	7914	7917
	7931	7940	7978	7987	8047	8055	8188	8189	8198	8239	8244	8258	8268	8273	8308
	8315	8419	8421	8439	8448	8472	8473	8475	8494	8497	8550	8555	8599	8602	8624
	8628	8646	8651	8682	8685	8710	8714	8740	8750	8783	8789	8813	8816	8832	8859
	8861	8891	8897	8911	8920	9028	9038	9085	9143	9151	9504	9508	9547	9549	9578

	9590	9606	9611	9635	9644	9675	9700	9709	9749	10112	10129	10147	10171	10218	10231
	10250	10276	10295	10317	10375	10402	10677	10690	10693	10736	10772	10801	10852	10862	10880
	10891	10943	10982	11044	11067	11144	11211	11214	11223	11230	11235	11236	11237	11245	11255
	11259	11260	11263	11264	11265	11269	11297	11303	11333	11336	11348	11351	11355	11361	11363
	11374	11375	11377	11379	11386	11388	11393	11395	11399	11402	11403	11406	11407	11410	11437
	11452	11463	11484	11489	11498	11542	11545	11590	11596	11599	11611	11612	11613	11614	11615
	11616	11617	11618	11619	11620	11621	12373	12374	12422	12426	12627				
.EQUIV	2050	2051	2059	2074	2075	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113
.EVEN	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141					
.IF	2340	3676	4681	4688	11328	11455	12092								
	2004	2007	2010	2013	2014	2030	2032	2033	2034	2035	2044	2048	2114	2142	2168
	2179	2190	2206	2210	2216	2219	2221	2225	2227	2229	2236	2252	2256	2258	2287
	2321	2330	2331	2332	2336	2337	2339	2362	2370	2373	2376	2379	2380	2381	2382
	2383	2385	3180	3478	3510	3594	3618	3643	3697	3737	3761	3772	3787	3789	3793
	3794	3796	3802	3804	3808	3809	3811	3820	3824	3832	3834	3838	3845	3847	3851
	3852	3854	4234	4463	4464	4689	4701	4707	4712	4714	4716	4718	4720	4722	4723
	4724	4726	4744	4757	4785	4851	4859	5003	5017	5177	5467	5470	5472	5497	5499
	5502	5649	5664	5666	5683	5685	5688	5744	5746	5760	5762	5765	5832	5834	5848
	5850	5853	5920	5922	5949	5951	5954	6006	6008	6032	6034	6037	6161	6162	6163
	6164	6165	6166	6168	6178	6181	6183	6187	6189	6195	6197	6198	6202	6205	6210
	6212	6214	6216	6218	6220	6221	6222	6224	6242	6718	6720	6755	6763	6772	6779
	6787	6793	6831	6836	6866	6872	6883	6892	6912	6914	6931	6933	6952	6954	6998
	7003	7036	7039	7053	7063	7123	7131	7148	7153	7168	7183	7255	7271	7323	7325
	7336	7339	7360	7364	7377	7407	7428	7505	7513	7530	7539	7555	7564	7582	7594
	7679	7692	7744	7747	7767	7771	7802	7806	7828	7834	7858	7864	7875	7878	7889
	7894	7913	7916	7930	7939	7977	7986	8046	8054	8188	8197	8238	8243	8256	8267
	8272	8307	8314	8418	8420	8438	8447	8472	8474	8493	8496	8549	8554	8598	8601
	8623	8627	8645	8650	8681	8684	8709	8713	8739	8749	8782	8788	8813	8815	8831
	8858	8860	8890	8896	8910	8919	9027	9037	9084	9142	9150	9503	9507	9546	9548
	9577	9579	9605	9610	9634	9643	9674	9684	9708	9748	9761	10111	10128	10146	10170
	10217	10230	10249	10275	10294	10316	10374	10401	10679	10692	10735	10771	10792	10851	10861
	10879	10890	10942	10981	11043	11066	11143	11210	11213	11223	11226	11233	11235	11236	11238
	11245	11248	11255	11259	11260	11262	11264	11265	11268	11269	11297	11303	11332	11335	11347
	11350	11355	11359	11361	11373	11375	11376	11377	11386	11388	11394	11396	11401	11402	11403
	11405	11407	11410	11437	11452	11462	11466	11484	11487	11498	11544	11589	11595	11599	11603
	11612	11613	11614	11615	11616	11617	11618	11619	11620	11621	12373	12422	12427		
.IFF	2005	2011	2030	2033	2034	2048	2217	2221	2228	2230	2237	2253	2256	2259	2287
	2337	2340	3180	3482	3514	3594	3643	4712	4852	4860	5004	5018	5471	5472	5498
	5499	5502	5649	5665	5666	5674	5685	5688	5745	5746	5761	5762	5765	5833	5834
	5849	5850	5853	5921	5922	5950	5951	5954	6007	6008	6033	6034	6037	6162	6165
	6168	6179	6182	6197	6210	6719	6721	6756	6764	6773	6780	6788	6794	6832	6837
	6867	6873	6884	6893	6913	6915	6932	6934	6953	6955	6999	7004	7037	7040	7054
	7064	7124	7132	7149	7154	7169	7184	7256	7272	7324	7326	7337	7340	7361	7365
	7378	7408	7429	7506	7514	7531	7540	7556	7565	7583	7595	7680	7694	7745	7748
	7768	7772	7803	7807	7829	7835	7859	7865	7876	7879	7890	7895	7914	7917	7931
	7940	7978	7987	8047	8055	8189	8198	8239	8244	8268	8273	8308	8315	8419	8421
	8439	8448	8473	8475	8494	8497	8550	8555	8599	8602	8624	8628	8646	8651	8682
	8685	8710	8714	8740	8750	8783	8789	8816	8832	8859	8861	8891	8897	8911	8920
	9028	9038	9085	9143	9151	9504	9508	9547	9549	9578	9580	9606	9611	9635	9644
	9675	9693	9700	9709	9749	9762	9779	9800	9806	9810	9821	9831	9836	9838	9860
	9864	9873	9882	9885	9897	9902	9906	9910	9925	9931	9944	9966	9973	9982	9989
	9999	10024	10028	10041	10048	10065	10068	10077	10080	10092	10100	10112	10129	10137	10147
	10171	10189	10212	10218	10231	10250	10276	10295	10317	10350	10375	10402	10419	10482	10485
	10508	10560	10570	10580	10584	10638	10647	10660	10668	10677	10680	10693	10736	10772	10852
	10862	10880	10891	10943	10982	11044	11067	11144	11211	11213	11226	11255	11260	11263	11265
	11269	11271	11276	11297	11333	11336	11348	11374	11375	11377	11402	11403	11406	11463	11475

3504	3505	3506	3507	3508	3509	3515	3520	3521	3522	3523	3524	3525	3526	3527
3528	3529	3530	3531	3532	3533	3534	3535	3536	3537	3538	3539	3540	3541	3545
3547	3549	3550	3551	3552	3553	3555	3556	3557	3558	3559	3560	3561	3562	3566
3567	3568	3572	3592	3594	3595	3608	3610	3611	3612	3613	3642	3678	3679	3680
3681	3682	3683	3684	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695	3696
3702	3703	3704	3705	3707	3708	3709	3710	3711	3712	3713	3714	3715	3716	3717
3718	3719	3720	3721	3722	3723	3724	3725	3726	3727	3729	3730	3732	3733	3736
3737	3738	3739	3740	3761	3762	3763	3764	3765	3766	3772	3773	3774	3775	3776
3777	3785	3786	3787	3788	3789	3790	3791	3792	3793	3794	3795	3800	3801	3802
3803	3804	3805	3806	3807	3808	3809	3810	3816	3817	3818	3819	3820	3821	3822
3823	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837
4166	4167	4168	4169	4170	4171	4172	4173	4174	4175	4176	4177	4178	4179	4180
4181	6095	6178	6181	6196	8455	8792	8840	8850	8947	8982	9019	10732	10767	10768
10798	10845	11015	11060	11140	11324	11435	11493	11540	11541	12093	12096	12097	12100	12101
12104	12105	12108	12110	12113	12115	12117	12119	12121	12124	12126	12128	12130	12132	12134
12136	12139	12141	12143	12145	12147	12149	12151	12153	12156	12158	12160	12163	12165	12167
12169	12171	12173	12175	12178	12180	12182	12184	12186	12188	12190	12193	12195	12197	12199
12201	12204	12206	12208	12210	12213	12215	12217	12219	12221	12223	12226	12228	12230	12232
12234	12236	12239	12241	12243	12245	12247	12249	12252	12254	12256	12258	12260	12262	12264
12266	12268	12271	12273	12275	12277	12279	12281	12283	12285	12287	12290	12292	12294	12296
12298	12301	12303	12306	12308	12311	12313	12315	12317	12319	12321	12324	12326	12329	12332
12335	12337	12340	12342	12344	12346	12348	12351	12353	12355	12357	12359	12362	12364	12366
12368	12370													

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*, DZR6NB.SEQ/SOL/CRF/NL:TOC=DRIVE8.P11/EQ:QNEWSW,DZR6NB.CMB
 RUN-TIME: 158 129 19 SECONDS
 RUN-TIME RATIO: 2271/308=7.3
 CORE USED: 50K (99 PAGES)

