

RK611/RK06

DRIVE COMPATIBILITY PROGRAM
CZR6QB0

AH-B162B-MC

COPYRIGHT © 77-78

FICHE 2 OF 2

MAR 1978

digital

MADE IN USA

TABLE OF CONTENTS

47	1.0 INTRODUCTION
48	2.0 HARDWARE REQUIREMENTS
49	3.0 PRELIMINARY PROGRAM REQUIREMENTS
50	4.0 GENERAL PROGRAM CONSIDERATIONS
51	4.1 SY3MAC
52	4.2 XXDP
53	4.3 ACT
54	4.4 APT
55	4.5 DUAL-ACCESS
56	4.6 MEMORY MANAGEMENT
57	4.7 MEMORY PARITY OPTION
58	4.8 BAD SECTORS
59	4.9 EXECUTION TIME
60	5.0 PROGRAM LOAD MEDIA
61	6.0 PROGRAM OPTIONS
62	6.1 STARTING ADDRESSES
63	6.2 SWITCH REGISTER OPTIONS USED
64	7.0 RUNNING THE PROGRAM
65	8.0 OPERATIONAL DIALOGUE
66	8.1 DIALOGUE FOR ADDRESS 200 START
67	8.2 DIALOGUE FOR ADDRESS 204 START
68	8.3 DIALOGUE FOR ADDRESS 220 START
69	8.4 PASS 1 DIALOGUE
70	8.5 PASS 2 DIALOGUE
71	9.0 DESCRIPTION OF TESTS
72	9.1 DESCRIPTION OF PASS 1 TESTS
73	9.2 DESCRIPTION OF PASS 2 TESTS
74	10.0 PRINTOUT OF TEST RESULTS
75	10.1 OVERWRITE AND DRIVE COMPATIBILITY DATA TEST RESULTS
76	11.0 ERROR REPORTING
77	11.1 COMMON ERRORS
78	11.2 ERROR HANDLING
79	11.3 ERROR PRINTOUT EXAMPLE
80	TABLE A - BASIC READ/WRITE TEST SECTORS
81	TABLE B - WORSE CASE DATA PATTERN
82	
83	
84	
85	
86	
87	
88	
89	
90	
91	
92	
93	
94	
95	
96	
97	
98	
99	
100	
101	
102	

103
104
105
106
107
108
109
110
111
112
113
114

TABLE C - CYLINDER BLOCK ASSIGNMENT FOR A GIVEN SURFACE
TABLE D - BASIC CYLINDER BLOCK LAYOUT EXAMPLE
TABLE E - OVERWRITE CYLINDERS
TABLE F - SELF-TEST CYLINDERS
TABLE G - PSEUDO-RANDOM DATA PATTERN

1.0 INTRODUCTION

THE PURPOSE OF THIS PROGRAM IS TO VERIFY THE COMPATIBILITY OF UP TO 16 RK06 OR RK07 DRIVES WHICH MAY RESIDE ON ONE OR MORE RK611/RK06-07 SUBSYSTEMS (INCLUDING SYSTEMS WITH MULTIPLE SUBSYSTEMS). COMPATIBILITY IS DEFINED HERE AS THE ABILITY OF A DRIVE TO WRITE DATA WHICH CAN BE READ SUCCESSFULLY BY ALL OTHER DRIVES, AND ADDITIONALLY THE ABILITY OF A DRIVE TO COMPLETELY OVER-WRITE DATA WRITTEN BY ALL OTHER DRIVES.

NOTE: RK06 AND RK07 DRIVES CANNOT BE MIXED FOR COMPATIBILITY TESTING.

THE PROGRAM IS DESIGNED TO DETECT THE FOLLOWING CONDITIONS WHICH MOST COMMONLY CAUSE INCOMPATIBILITY BETWEEN DRIVES:

1. HEAD MIS-ALIGNMENT
2. POSITIONER LATERAL MISALIGNMENT
3. SPINDLE-CARTRIDGE INTERFACE RUNOUT
4. IMPROPER LEVELS OF WRITE CURRENT
5. INCORRECT ADDRESSING OF READ/WRITE HEADS

THE TESTING IS DONE IN TWO PASSES. IN PASS 1, COMPATIBILITY DATA PATTERNS ARE WRITTEN BY ALL THE DRIVES UPON THE SAME DISK CARTRIDGE, AND THE BASIC READ/WRITE CAPABILITY OF EACH DRIVE IS DEMONSTRATED. IN PASS 2, THE COMPATIBILITY DATA FROM ALL DRIVES IS READ BY EACH DRIVE WITH INCREASING HEAD OFFSET, AND THIS IS COMPARED WITH EACH DRIVE'S ABILITY TO READ ITS OWN DATA. IN ADDITION, EACH DRIVE'S CAPABILITY TO OVERWRITE DATA WRITTEN BY ALL OTHER DRIVES IS TESTED ON THE SECOND PASS. (FOR THE REMAINDER OF THIS SPECIFICATION, THE ABOVE DEFINITIONS OF THE FIRST AND SECOND PASS SHALL APPLY).

IN BOTH PASSES, THE PROGRAM DIRECTS THE OPERATOR IN THE LOADING AND UNLOADING OF DRIVES AND THE MOVEMENT OF THE CARTRIDGE FROM DRIVE TO DRIVE, THROUGH MESSAGES AT THE CONSOLE TERMINAL. AT THE COMPLETION OF TESTING ON EACH DRIVE DURING THE SECOND PASS A SUMMARY IS PRINTED OF COMPATIBILITY TEST RESULTS FOR THAT DRIVE.

WITHIN THE VARIOUS TESTS OF BOTH PASSES, THE CAPABILITY IS PROVIDED TO LOOP ON CURRENT OPERATIONS, AND SWITCH REGISTER OPTIONS ARE PROVIDED FOR A VARIETY OF LOOPING, RUNNING, AND REPORTING MODES (SEE SECTION 6.2).

UNEXPECTED 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

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE RK06-07 DRIVE COMPATIBILITY PROGRAM, FOR EACH SUBSYSTEM WHICH MAY BE PRESENT:

PDP-11/04, (05,10 MFG. ONLY), 20,30,34,35,40,45,50,70, OR PDQ
16K MEMORY
CONSOLE TERMINAL
RK611 CONTROLLER
1 TO 8 RK06 OR RK07 DISK DRIVES PER CONTROLLER

IN ADDITION, A SINGLE RK06 DISK CARTRIDGE IS REQUIRED WHICH MAY BE FORMATTED IN EITHER 20 OR 22 SECTOR FORMAT, ON A RELIABLE WELL-ALIGNED RK06 DRIVE. THIS CARTRIDGE WILL BE MOVED FROM DRIVE TO DRIVE, (ON UP TO 16 DRIVES) ON EACH OF TWO PASSES.

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE RK06 DRIVE COMPATIBILITY PROGRAM, THE SUBSYSTEM(S) UNDER TEST SHOULD BE CAPABLE OF PASSING THE CONTROLLER DIAGNOSTICS CZR6A-CZR6E AND CZR6K, THE DRIVE DIAGNOSTICS CZR6H-CZR6J, AND THE SUBSYSTEM VERIFICATION PROGRAMS CZR6M-CZR6N. IN ADDITION, THE CARTRIDGE MUST BE FORMATTED IN 20 OR 22 SECTOR FORMAT USING THE PACK FORMATTER CZR6L, OR EQUIVALENT (PERFORMED ON KNOWN GOOD, ALIGNED DRIVE).

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, AND MAY BE RUN IN DUMP MODE ONLY. DUE TO MANUAL INTERVENTION AND LACK OF END-OF-PASS HOOKS, THE PROGRAM IS NOT XXDP CHAINABLE.

4.3 ACT

THIS PROGRAM MAY BE LOADED UNDER ACT AND MAY BE RUN IN DUMP MODE ONLY.
IT IS NOT CHAINABLE UNDER ACT.

4.4 APT

THIS PROGRAM MAY BE LOADED BY THE APT SYSTEM, BUT MAY BE RUN IN
PROGRAM (DUMP) MODE ONLY. IT CANNOT BE RUN IN APT SCRIPT MODE.

4.5 DUAL-ACCESS

THIS PROGRAM DOES NOT UTILIZE THE DUAL-ACCESS OPTION IN ANY WAY, AND
ALL DRIVES UNDER TEST SHOULD BE DE-SELECTED THROUGH THE PORT WHICH IS
NOT IN USE.

4.6 MEMORY MANAGEMENT

MEMORY MANAGEMENT IS NOT UTILIZED IN THIS PROGRAM. IF IT IS
INSTALLED, IT IS DISABLED BY THE PROGRAM.

4.7 MEMORY PARITY OPTION

IF PARITY MEMORY IS INSTALLED, MEMORY PARITY TRAPS ARE DISABLED BY THE
PROGRAM.

4.8 BAD SECTORS

THE LIST OF BAD SECTORS ON THE CARTRIDGE IS OBTAINED FROM THE FIRST
DRIVE TO BE TESTED ON THE CURRENT SUBSYSTEM. ACCORDING TO A SWITCH
REGISTER OPTION (SEE SECTION 6.2) THIS LIST MAY BE TYPED AT THE
CONSOLE AT THE START OF THE FIRST PASS. ALL DATA ERRORS OCCURING IN
THE PROGRAM ARE IGNORED IF THEY OCCUR IN SECTORS DESIGNATED AS BAD.

4.9 EXECUTION TIME

THE TOTAL TIME REQUIRED TO RUN THE DRIVE COMPATIBILITY PROGRAM IS
DIRECTLY PROPORTIONAL TO THE NUMBER OF DRIVES TO BE TESTED AND
REQUIRES ABOUT 8-10 MINUTES PER DRIVE.

216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269

5.0 PROGRAM LOAD MEDIA

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

6.0 PROGRAM OPTIONS

6.1 STARTING ADDRESSES

200 - THIS IS THE STARTING ADDRESS FOR DEFAULT PARAMETERS AND RUNNING OF PASS 1 AND PASS 2 ON A SINGLE SUBSYSTEM. THE PROGRAM WILL USE DEFAULT RK611 BASE ADDRESS, INTERRUPT VECTOR AND PRIORITY, AND WILL AUTOMATICALLY DETERMINE WHICH DRIVES TO TEST. DRIVES MUST BE ON-LINE, POWERED UP, AND EITHER LOADED OR UNLOADED. THE PROGRAM WILL ASSUME ALL DRIVES TO BE TESTED RESIDE ON ONE RK611/RK06 SUBSYSTEM ONLY.

204 - THIS IS THE STARTING ADDRESS TO RUN PASS 1 ON ALL RK611/RK06 SUBSYSTEMS WHICH RESIDE ON THIS PDP-11 SYSTEM. THE PROGRAM WILL ASK FOR THE RK611 BASE ADDRESS, INTERRUPT VECTOR, AND PRIORITY FOR EACH SUBSYSTEM ON THIS SYSTEM, AND IT ASKS FOR THE LETTER NAMES (A THRU P) ASSIGNED TO ALL OTHER SUBSYSTEMS, AND THE DRIVE(S) WHICH WILL BE TESTED ON EACH.

220 - THIS IS THE STARTING ADDRESS TO RUN PASS 2 ON ALL RK611/RK06 SUBSYSTEMS WHICH RESIDE ON THIS PDP-11 SYSTEM. THE PROGRAM WILL ASK FOR THE RK611 BASE ADDRESS, INTERRUPT VECTOR, AND PRIORITY FOR EACH SUBSYSTEM ON THIS SYSTEM, AND IT ASKS FOR THE LETTER NAMES (A THRU P) ASSIGNED TO ALL OTHER SUBSYSTEMS, AND THE DRIVE(S) WHICH WILL BE TESTED ON EACH.

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 ARE SET TO 1:

BIT	OPTION
---	-----
15	HALT ON ERROR
14	LOOP ON CURRENT TEST

13 INHIBIT ERROR REPORTS
12 REPORT DESCRIPTION ONLY, ON ERRORS
11 UNUSED
10 BELL ON ERROR
09 LOOP ON ERROR
08 APPLY RANDOM STALL BETWEEN OPERATIONS
07 TYPE BAD SECTOR FILES (BSF'S) AT START
06-00 UNUSED

7.0 RUNNING THE PROGRAM

ONCE THE PROGRAM HAS BEEN LOADED INTO CORE (IN A GIVEN SYSTEM, IF THERE ARE MULTIPLE SYSTEMS) THE FOLLOWING STEPS MUST BE TAKEN TO RUN THE PROGRAM:

1. INSURE THAT ALL DRIVES TO BE TESTED ARE POWERED UP AND SINGLE PORT SELECTED.
2. LOAD THE DESIRED START ADDRESS.
3. SET ANY DESIRED BITS IN THE HARDWARE SWITCH REGISTER (IF PRESENT).
4. START THE PROGRAM.
5. FOLLOW ALL INSTRUCTIONS TYPED BY THE PROGRAM PERTAINING TO THE MANUAL INTERVENTION REQUIRED, AND THE ALTERNATE USE OF MULTIPLE SYSTEMS (IF THERE ARE ANY).

8.0 OPERATIONAL DIALOGUE

THIS SECTION DESCRIBES THE CONSOLE TERMINAL DIALOGUE THROUGH WHICH THE PROGRAM DIRECTS THE OPERATOR, IN THE SELECTION OF OPTIONS AND THE LOADING AND UNLOADING OF DRIVES, AND THE MOVEMENT OF THE TEST CARTRIDGE. THE EXACT DIALOGUE WHICH IS USED DEPENDS UPON THE STARTING ADDRESS WHICH WAS CHOSEN (SEE SECTION 6.1).

IN THE FOLLOWING DISCUSSION AND IN THE PRINTOUT OF TEST RESULTS, DRIVES TO BE TESTED WILL BE REFERRED TO BY A LETTER AND A NUMBER. THE LETTER IS THE SUBSYSTEM LETTER NAME (OPERATOR ASSIGNED), AND THE NUMBER IS THE DRIVE NUMBER ON THAT SUBSYSTEM. FOR EXAMPLE, DRIVE C6 REFERS TO DRIVE 6 ON SUBSYSTEM C.

381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
8.1 DIALOGUE FOR ADDRESS 200 START

THIS STARTING ADDRESS MAY BE USED FOR AUTOMATIC DRIVE SIZING AND DEFAULTING OF PARAMETERS, WHEN THE DRIVES RESIDE ON ONE SUBSYSTEM ONLY. THE PROGRAM FIRST IDENTIFIES ITSELF AS FOLLOWS:

CZR6Q80 - RK06-RK07 DRIVE COMPATIBILITY PROGRAM

THEN, THE DRIVES ON THE SYSTEM ARE AUTOMATICALLY SIZED, AND ALL EXISTING DRIVES WILL BE MARKED FOR TESTING. THE PROGRAM TYPES THE DRIVE LIST, AS IN THE FOLLOWING EXAMPLE:

DRIVE TYPE 6<CR> FOR RK06, 7<CR> FOR RK07:

THE OPERATOR MUST TYPE EITHER A 6<CR> OR A 7<CR>

DRIVES = 2,5,7

THE PROGRAM NOW PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE MOUNTING OF THE PACK, AS DESCRIBED IN SECTION 8.4.

PLEASE NOTE THAT THERE IS ONLY ONE SUBSYSTEM ON AN ADR. 200 START, AND IT IS NAMED SUBSYSTEM A. THE DRIVES IN THE ABOVE EXAMPLE WOULD BE REFERRED TO AS A2,A5,A7 IN THE TEST RESULTS PRINTOUT AT THE END OF PASS 2.

412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
8.2 DIALOGUE FOR ADDRESS 204 START

THIS STARTING ADDRESS MUST BE USED ON EACH SYSTEM, WHEN THERE IS MORE THAN ONE SUBSYSTEM, BUT IT MAY ALSO BE USED WHEN THERE IS JUST ONE SUBSYSTEM (TOTAL), TO SPECIFY DRIVES TO TEST AND NON-DEFAULT PARAMETER VALUES, FOR PASS 1.

THE PROGRAM IDENTIFIES ITSELF AS FOLLOWS:

CZR6Q80 - RK06-RK07 DRIVE COMPATIBILITY PROGRAM

THEN, THE PROGRAM ASKS THE OPERATOR FOR THE DRIVES TO BE TESTED ON EACH OF THE POSSIBLE SUBSYSTEMS (STARTING WITH A - THE NAMES RANGE FROM SUBSYS A TO SUBSYS P. THERE COULD BE UP TO 16 SUBSYSTEMS, WITH A DRIVE ON EACH) :

DRIVE TYPE 6<CR> FOR RK06, 7<CR> FOR RK07:

THE OPERATOR MUST TYPE IN 6<CR> OR 7<CR>

SUBSYS A DRIVE(S) =

THE OPERATOR THEN TYPES THE DESIRED DRIVE NUMBERS, AS IN THE FOLLOWING EXAMPLE:

K01

CZR6QBO RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 11
CZR6Q8.P11 02-DEC-77 10:46

SEQ 0010

437
438
439
440
441
442
443

SUBSYS A DRIVE(S) = 2,5,7

THE PROGRAM THEN VERIFIES THE DRIVE NOS. BY TYPING :

WILL TEST DRIVE(S) 2,5,7 ON SUBSYS A.

NEXT, THE PROGRAM ASKS THE FOLLOWING QUESTION:

IS THERE ANOTHER SUBSYS (Y OR N <CR>)?

THE OPERATOR TYPES Y<CR> OR N<CR>. (IF JUST <CR> IS TYPED, THE PROGRAM ASSUMES THAT N<CR> WAS TYPED). IF THE OPERATOR TYPED N, THE PROGRAM PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE MOUNTING OF THE PACK, AS DESCRIBED IN SECTION 8.4. IF Y WAS TYPED, THE PROGRAM ASKS FOR THE NUMBERS OF THE DRIVES TO BE TESTED ON THE NEXT SUBSYSTEM (SUBSYS B) AS FOLLOWS:

SUBSYS B DRIVE(S) =

THE OPERATOR TYPES THE DRIVE NUMBERS, AS IN THE FOLLOWING EXAMPLE:

SUBSYS B DRIVE(S) = 2,3

THE PROGRAM THEN VERIFIES THE DRIVE NOS. BY TYPING :

WILL TEST DRIVE(S) 2,3 ON SUBSYS B.

NEXT, THE PROGRAM WILL ASK :

IS THERE ANOTHER SUBSYS (Y OR N <CR>)?

AND IN THE SAME MANNER, THE OPERATOR SPECIFIES THE DRIVES ON EACH OF THE REMAINING SUBSYSTEMS, UNTIL ALL HAVE BEEN SPECIFIED.

NEXT, THE PROGRAM ASKS FOR THE LETTER NAME ASSIGNED TO THE SUBSYSTEM TO BE TESTED NEXT :

TYPE NAME OF NEXT SUBSYS TO TEST (A-P) :

THE OPERATOR RESPONDS, AS IN THE FOLLOWING EXAMPLE :

TYPE NAME OF NEXT SUBSYS TO TEST (A-P) : A

ALL SUBSYSTEMS MUST BE TESTED IN THE ORDER IN WHICH THE LETTERS ARE ASSIGNED (A THRU P). NEXT, THE PROGRAM ALLOWS THE OPERATOR TO ALTER THE RK611 BUS ADDRESS, VECTOR ADDRESS, AND INTERRUPT PRIORITY FOR THIS SUBSYSTEM. FOR EACH PARAMETER THE CURRENT VALUE IS TYPED, AND THE OPERATOR IS GIVEN THE OPPORTUNITY TO TYPE IN A NEW VALUE, PLUS <CR>. IF JUST <CR> IS TYPED, THE PARAMETER IS NOT CHANGED. WHEN THE PROGRAM IS FIRST LOADED, THE FOLLOWING DEFAULT VALUES ARE ASSIGNED: RK611 BUS ADDRESS = 177440, VECTOR ADDRESS = 210, AND PRIORITY = 5. THE FOLLOWING EXAMPLE SHOWS A PRINTOUT IN WHICH ONLY THE VECTOR WAS CHANGED:

RK611 BUS ADR	=	177440	NEW =
RK611 VEC ADR	=	210	NEW = 240
RK611 PRIORITY	=	5	NEW =

THEN THE PROGRAM PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE MOUNTING OF THE PACK, AS DESCRIBED IN SECTION 8.4. AT THE COMPLETION OF PASS 1 ON THE SUBSYSTEM, THE PROGRAM WILL INFORM THE OPERATOR HOW

500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555

TO PERFORM PASS 1 ON THE NEXT SUBSYSTEM.

8.3 DIALOGUE FOR ADDRESS 220 START

THIS STARTING ADDRESS MUST BE USED ON EACH SYSTEM, WHEN THERE IS MORE THAN 1 SUBSYSTEM, BUT IT MAY ALSO BE USED WHEN THERE IS JUST ONE SUBSYSTEM (TOTAL) TO SPECIFY DRIVES TO TEST AND NON-DEFAULT PARAMETER VALUES, FOR PASS 2. THE PROGRAM IDENTIFIES ITSELF, AS FOLLOWS :

CZR6Q80 - RK06-RK07 DRIVE COMPATIBILITY PROGRAM

THE DIALOGUE FOR 220 START IS IDENTICAL TO THE DIALOGUE FOR THE 204 START DESCRIBED ABOVE (SECTION 8.2), FOR THE SELECTION OF SUBSYSTEM PARAMETERS AND THE SPECIFICATION OF ALL DRIVES TO BE TESTED ON THE VARIOUS SUBSYSTEMS. HOWEVER, AFTER THIS DIALOGUE IS COMPLETED, THE PROGRAM PROCEEDS WITH PASS 2, AND DIRECTS THE OPERATOR IN THE MOVEMENT OF THE PACK, AS DESCRIBED IN SECTION 8.5.

NOTE THAT SINCE THE APPROPRIATE PROCESSOR MUST BE STARTED AT THE STARTING ADDRESS FOR EACH SUBSYSTEM TO BE TESTED, THE COMPATIBILITY TEST MAY BE PERFORMED IN STEPS, AT VARIOUS TIMES AND BETWEEN VARIOUS DISTANT LOCATIONS, BY MOVING THE TEST PACK AND SAVING THE PRINTOUT FROM EACH PASS ON EACH PDP-11 SYSTEM INVOLVED.

8.4 PASS 1 DIALOGUE

AFTER THE SELECTION OF PARAMETERS AND DRIVES HAS BEEN COMPLETED ON THE CURRENT SUBSYSTEM (SECTIONS 8.1-8.2), THE PROGRAM INDICATES THE START OF PASS 1 AS FOLLOWS:

** STARTING PASS 1 **

NEXT, THE PROGRAM SELECTS THE FIRST DRIVE TO BE TESTED ON THIS SUBSYSTEM, AND INSTRUCTS THE OPERATOR TO MOUNT THE TEST CARTRIDGE AND LOAD THE HEADS ON THAT DRIVE, AS IN THE FOLLOWING EXAMPLE:

MOUNT PACK ON DRIVE A2 AND LOAD.
TYPE R<CR> WHEN DRIVE READY:

THE OPERATOR PERFORMS THIS TASK AND TYPES R<CR> WHEN THE DRIVE READY LIGHT IS ON. THE PROGRAM PERFORMS PASS 1 FUNCTIONS ON THIS DRIVE (SEE SECTION 9.1) AND THEN INSTRUCTS THE OPERATOR TO UNLOAD THE DRIVE AND REMOVE THE PACK AS FOLLOWS:

UNLOAD DRIVE A2 AND REMOVE PACK.
TYPE R<CR> WHEN DONE:

THE OPERATOR PERFORMS THESE FUNCTIONS AND TYPES R<CR> AFTER THE PACK HAS BEEN REMOVED.

IN THE SAME MANNER, THE PROGRAM INSTRUCTS THE OPERATOR IN THE MOVEMENT OF THE PACK THROUGHOUT THE REST OF THE DRIVES ON THE CURRENT SUBSYSTEM. WHEN THIS HAS BEEN COMPLETED, THE PROGRAM DOES ONE OF THREE THINGS: (1) IF THERE IS ONLY ONE SUBSYSTEM (FROM ADR 200 START) THE PROGRAM BEGINS PASS 2 (SEE SECTION 8.5). (2) IF THERE IS ANOTHER SUBSYSTEM, THE PROGRAM DIRECTS THE OPERATOR TO PERFORM PASS 1 ON THE NEXT SUBSYS AS FOLLOWS (AND THEN HALTS) :

START AT ADR 204 FOR PASS 1 ON SUBSYS B

(3) IF THERE ARE NO MORE DRIVES TO TEST IN PASS 1 ON ANY SUBSYS, THE PROGRAM DIRECTS THE OPERATOR TO BEGIN PASS 2 ON THE FIRST SUBSYS (SEE SECT. 8.5) AS FOLLOWS (AND THEN HALTS) :

START AT ADR 220 FOR PASS 2 ON SUBSYS A

8.5 PASS 2 DIALOGUE

THE OPERATOR RETURNS TO THE FIRST SUBSYSTEM TO PERFORM PASS 2 EITHER THROUGH THE DIALOGUE OF THE ADR 200 START, OR AFTER THE SELECTION OF PARAMETERS AND DRIVES HAS BEEN COMPLETED IN ACCORDANCE WITH THE DIALOGUE OF THE ADR 220 START (SEE SECTION 8.3). IN EITHER CASE, THE PROGRAM INDICATES THE START OF PASS 2 BY TYPING :

** STARTING PASS 2 **

THE PROGRAM THEN DIRECTS THE OPERATOR IN THE UNLOADING, PACK MOVEMENT, AND LOADING OF ALL DRIVES ON THE FIRST SUBSYSTEM, IN THE SAME MANNER AS DESCRIBED FOR PASS 1 (SEE SECTION 8.4).

HOWEVER, AFTER PASS 2 TESTING (SEE SECTION 9.2) IS COMPLETED ON A GIVEN DRIVE, THE ENTIRE TEST RESULTS FOR THAT DRIVE ARE TYPED. THE DETAILS OF THIS PRINTOUT ARE DESCRIBED IN SECTION 10, AFTER THE DETAILS OF THE TESTING ARE DESCRIBED.

WHEN PASS 2 HAS BEEN COMPLETED FOR ALL DRIVES ON THE FIRST SUBSYSTEM, THE PROGRAM DOES ONE OF TWO THINGS: (1) IF THERE IS ONLY ONE SUBSYSTEM (FROM ADR 200 START) OR IF ALL DRIVES ON ALL SUBSYSTEMS HAVE BEEN TESTED IN PASS 2 (FROM ADR 220 START), THE ENTIRE TESTING AND REPORTING HAVE BEEN COMPLETED, AND THE PROGRAM TYPES:

** END OF TESTING **

CZR6080 Rk6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 15
CZR608.P11 02-DEC-77 10:46

SEQ 0014

612
613
614
615
616
617
618
619

(2) IF THERE IS ANOTHER SUBSYSTEM, HOWEVER, THE PROGRAM DIRECTS THE
OPERATOR TO PERFORM PASS 2 ON THE NEXT SUBSYSTEM AS FOLLOWS (AND THEN
HALTS):

START AT ADR 220 FOR PASS 2 ON SUBSYS B

9.0 DESCRIPTION OF TESTS

THE MAIN FUNCTIONAL BLOCKS OF CODE IN THE PROGRAM ARE ASSIGNED TEST NUMBERS, FOR THE PURPOSE OF IDENTIFICATION IN ERROR PRINTOUTS. TEST 0 REFERS TO THE OPERATOR INPUT DIALOGUE ROUTINES DESCRIBED IN SECTIONS 8.1-8.3. THE OTHER TEST NUMBERS ARE ASSIGNED BELOW, IN THE DESCRIPTION OF PASS 1 AND PASS 2 TESTING.

IN THE FOLLOWING SECTIONS, TABLES A-G ARE REFERRED TO. IN THESE TABLES, DRIVES ARE NAMED FROM 0-17 FOR ILLUSTRATIVE PURPOSES, ALTHOUGH THE DRIVES ARE NAMED THE FOLLOWING WAY IN AN ACTUAL SITUATION : A0,A1, A2,...B0,B1,B2,...C0,C1,C2,... ETC. (SEE SECTION 8.0).

9.1 DESCRIPTION OF PASS 1 TESTS

IN PASS 1, THE BASIC READ/WRITE CAPABILITY OF EACH DRIVE IS DEMONSTRATED, AND COMPATIBILITY DATA PATTERNS ARE WRITTEN BY ALL DRIVES UPON THE SAME TEST CARTRIDGE.

THE SEQUENCE OF OPERATIONS PERFORMED ON EACH DRIVE IS AS FOLLOWS:

1. TEST 1 - MOUNTING OF TEST CARTRIDGE FOR PASS 1 - THE OPERATOR MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.4).
2. TEST 2 - BASIC READ/WRITE DATA TEST - THE PROGRAM PERFORMS A WRITE AND WRITE CHECK OPERATION USING A "WORST CASE" DATA PATTERN, AT THE APPROPRIATE SECTOR FOR THIS DRIVE (SEE TABLE A) ON ALL SURFACES. THE PURPOSE OF THIS OPERATION IS TO VERIFY THE BASIC READ/WRITE CAPABILITY OF THE DRIVE ON PASS 1. THE ENTIRE SECTOR IS WRITTEN WITH THE REPETITION OF THE DATA PATTERN SHOWN IN TABLE B. THIS PATTERN CONSISTS OF A MIXTURE OF HIGH AND LOW FREQUENCY ELEMENTS AND BIT STRINGS REQUIRING VARIOUS DEGREES OF PRECOMPENSATION, WHEN ENCODED.
3. TEST 3 - WRITE OVERWRITE AND COMPATIBILITY CYLINDER BLOCKS - NEXT, THE PROGRAM WRITES ALL SECTORS FOR THIS DRIVE WITHIN THE CYLINDER BLOCKS SHOWN IN TABLE C ON ALL SURFACES USING A SINGLE REPEATED WORD OF THE PATTERN IN TABLE G. LOGICAL DRIVE 0 USES WORD 0, LOGICAL DRIVE 1 USES WORD 1, LOGICAL DRIVE 10 USES WORD 10, ETC. THUS, THE DATA FROM EACH DRIVE IS UNIQUE. TABLE C HAS BEEN DETERMINED AS FOLLOWS:

IN EACH OF THE SEVEN WRITE CURRENT ZONES ON EACH SURFACE SECTORS ARE WRITTEN WITHIN TWO DISTINCT CYLINDER BLOCKS. IN THE FIRST 16 CYLINDERS OF EACH WRITE CURRENT ZONE DATA IS WRITTEN TO BE LATER OVERWRITTEN IN PASS 2, DURING THE OVERWRITE TEST. IN THE LAST 16 CYLINDERS OF EACH CURRENT

676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715

ZONE (EXCEPT THE INNERMOST ZONE) COMPATIBILITY DATA IS WRITTEN TO BE READ WITH OFFSET IN PASS 2. WITHIN EACH CURRENT ZONE BOTH THE OVERWRITE AND COMPATIBILITY CYLINDER BLOCKS ARE IDENTICALLY WRITTEN BUT FROM ZONE TO ZONE THE DATA WRITTEN BY EACH DRIVE IS ROTATED SEVERAL SECTORS SO THAT OVER ALL THE ZONES THE DATA APPEARS AT VARIOUS ANGULAR POSITIONS ON THE PACK.

WITHIN EACH CYLINDER BLOCK UP TO 16 SECTORS ARE WRITTEN (DEPENDING ON THE NUMBER OF DRIVES BEING TESTED) ON EACH CYLINDER. THESE SECTORS ARE ALWAYS SECTORS 0,1,2,3,5,6,7,10,12,13,14,15,17,20,21,22(OCTAL). OF THE REMAINING SECTORS, 4,11,16, AND 23 ARE USED FOR DRIVE SELF-TESTING IN PASS 2 (SEE SECTION 9.2).

THE BASIC LAYOUT OF A TYPICAL CYLINDER BLOCK IS SHOWN IN TABLE D, WHERE THE BLOCK SHOWN IS THE COMPATIBILITY BLOCK FOR ZONE 1, WHICH STARTS ON CYLINDER 60, AND HAS THE ROTATING STARTING SECTOR = SECTOR 0. EACH NUMBER INSIDE THE BLOCK IS THE NUMBER OF THE DRIVE WHICH WRITES THAT SECTOR. TABLE D SHOWS 16 BLOCKS WRITTEN BY EACH OF 16 DRIVES. IF ANY OF THE DRIVES SHOWN ARE NOT PRESENT, HOWEVER, THE BLOCKS RESERVED FOR THE MISSING DRIVES ARE SIMPLY NOT WRITTEN.

THE ABOVE PATTERN OF SECTOR WRITES INSURES THAT DATA FROM EACH DRIVE IS WRITTEN ON ADJACENT CYLINDERS TO DATA FROM EVERY OTHER DRIVE, IN BOTH DIRECTIONS. IN ADDITION, THE ROTATION OF THE ABOVE SECTORS FROM CURRENT ZONE TO CURRENT ZONE INSURES THAT OVERWRITE AND DATA COMPATIBILITY TESTING IS DONE AT SEVERAL DIFFERENT ANGULAR POSITIONS WITH RESPECT TO THE CARTRIDGE.

4. TEST 4 - DISMOUNTING OF TEST CARTRIDGE IN PASS 1 - THE OPERATOR UNLOADS THE DRIVE AND DISMOUNTS THE PACK, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.4), TO PROCEED WITH THE ABOVE STEPS ON THE NEXT DRIVE.

9.2 DESCRIPTION OF PASS 2 TESTS

IN PASS 2, THE ABILITY OF EACH DRIVE TO COMPLETELY OVERWRITE DATA WRITTEN BY ALL OTHER DRIVES AND TO READ DATA WRITTEN BY ALL OTHER DRIVES, IS TESTED.

THE SEQUENCE OF OPERATIONS PERFORMED BY EACH DRIVE IS AS FOLLOWS:

1. TEST 5 - MOUNTING OF TEST CARTRIDGE FOR PASS 2 - THE OPERATOR MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.5).
2. TEST 6 - OVERWRITE TEST - NEXT, THE PROGRAM PROCEEDS TO TEST THIS DRIVE'S OVERWRITE CAPABILITY. FIRST, THE APPROPRIATE CYLINDERS IN TABLE E FOR THIS DRIVE ARE OVERWRITTEN, ON EACH SURFACE. THE DATA USED IS A REPETITION OF A SINGLE WORD OF THE PATTERN IN TABLE G. LOGICAL DRIVE 0 USES WORD 0, LOGICAL DRIVE 1 USES WORD 1, LOGICAL DRIVE 10 USES WORD 10, ETC.

THEN, EACH CYLINDER OVERWRITTEN IS READ BACK BY THIS DRIVE WITH THE FOLLOWING RANGE OF OFFSET VALUES: 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION (+ AND -). THE PROGRAM SCANS FOR READ ERRORS (DCK, HVRC, ETC.) DURING THIS READ, AND IF ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA HAS NOT BEEN CORRECTLY OVERWRITTEN, AND A SCORE FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS PERFORMING THE OVERWRITES. FOR EACH TRACK (0, 1, 2), SCORES ARE AVERAGED OVER ALL CYLS TESTED, IN EACH OFFSET DIRECTION. AT THE COMPLETION OF THE OVERWRITE TEST ON THIS DRIVE, THE SCORES OF ALL THE DRIVES ARE CONVERTED AND STORED, FOR PRINTING AT THE END OF PASS 2 (AS DESCRIBED IN SECTION 10.2). EACH SCORE IS PROPORTIONAL TO THE DEGREE OF OFFSET WHICH COULD BE APPLIED IN A GIVEN OFFSET DIRECTION BY THE CURRENT DRIVE WHILE SUCCESSFULLY READING THE DATA IT WROTE OVER ONE OF THE OTHER DRIVE'S DATA. THUS, THE PRINTOUT REVEALS WHICH DRIVES ARE INVOLVED, IN A SITUATION IN WHICH A DRIVE CANNOT OVERWRITE ONE OR SEVERAL OTHER DRIVE'S DATA.

3. TEST 7 - DRIVE SELF-TEST - THE PROGRAM NEXT EVALUATES THE DRIVE'S ABILITY TO WRITE AND READ ITS OWN DATA, AT VARIOUS POSITIONS ON THE PACK. FIRST, SECTORS 4, 11, 16, AND 23 OF THE APPROPRIATE CYLINDERS SHOWN IN TABLE F FOR THIS DRIVE ARE WRITTEN WITH THE DATA PATTERN SHOWN IN TABLE B, FOR ALL SURFACES. THEN, THE SECTORS ARE READ WITH THE FOLLOWING OFFSETS APPLIED: 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION. THE PROGRAM SCANS FOR READ ERRORS DURING EACH READ, AND IT COMPUTES A SCORE WHICH IS PROPORTIONAL TO THE FAILING OFFSET

772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813

MAGNITUDE. THEN, THE SCORES FOR ALL 4 SECTORS READ IN THIS CYLINDER BLOCK ARE AVERAGED, TO COME UP WITH A DRIVE SELF-TEST SCORE FOR EACH SURFACE FOR EACH OFFSET DIRECTION. THIS SCORE IS SAVED FOR LATER USE, TO BECOME THE STANDARD FOR THE COMPATIBILITY DATA READS WHICH ARE TO FOLLOW.

4. TEST 10 - COMPATIBILITY DATA READ TEST - HAVING ESTABLISHED A SELF-TEST SCORE FOR THIS DRIVE, THE PROGRAM PROCEEDS TO PERFORM THE COMPATIBILITY DATA READS OF THE PATTERNS WRITTEN BY ALL THE DRIVES IN EACH CYLINDER BLOCK (ON EACH SURFACE). EACH COMPATIBILITY CYLINDER BLOCK SHOWN IN TABLE C IS READ, A CYLINDER AT A TIME, FOR THE FOLLOWING OFFSET VALUES: 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION. THE PROGRAM SCANS FOR READ ERRORS DURING EACH READ AND IF ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA WAS BEING READ AT THAT INSTANT AND A SCORE FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS READING THE COMPATIBILITY DATA. THEN, EACH SCORE IS APPROPRIATELY ADJUSTED TO REFLECT THE SELF-TEST SCORE FOR THE CURRENT DRIVE AT THAT PARTICULAR CYLINDER BLOCK. THE SCORES ARE THEN AVERAGED OVER ALL CYLINDER BLOCKS. EACH SCORE IS PROPORTIONAL TO THE CAPABILITY OF THE CURRENT DRIVE TO SUCCESSFULLY READ THE DATA WRITTEN BY ONE OF THE OTHER DRIVES, AND SCORES ARE COMPUTED SEPARATELY FOR EACH SURFACE (TRACK), FOR EACH OFFSET DIRECTION. THUS, THE PRINTOUT REVEALS WHICH DRIVES ARE INVOLVED IN A SITUATION IN WHICH A PARTICULAR DRIVE HAS DIFFICULTY IN READING THE DATA OF ONE OR SEVERAL OTHER DRIVES.
5. TEST 11 - TYPE TEST SCORES AND DISMOUNT PACK IN PASS 2 - THE OVERWRITE AND COMPATIBILITY DATA READ TEST SCORES FOR THIS DRIVE ARE CONVERTED AND TYPED. THEN, THE OPERATOR UNLOADS THE DRIVE AND DISMOUNTS THE PACK AS DIRECTED BY THE PROGRAM (SEE SECTION 8.5), TO PROCEED WITH THE ABOVE STEPS ON THE NEXT DRIVE.

10.0 PRINTOUT OF TEST RESULTS

THE TEST RESULTS ARE PRINTED AT THE END OF PASS 2 ON EACH DRIVE BEING TESTED. THESE RESULTS PERTAIN TO THE OVERWRITE TEST AND THE COMPATIBILITY DATA READ TEST.

10.1 OVERWRITE AND DRIVE COMPATIBILITY DATA TEST RESULTS

THE RESULTS OF BOTH THE OVERWRITE AND THE COMPATIBILITY DATA READ TESTS ARE ALWAYS PRINTED, REGARDLESS OF DEGREE OF SUCCESS. COMPLETE RESULTS ARE PRINTED, SEPARATELY FOR EACH DRIVE, AT THE END OF PASS 2 ON THAT DRIVE. THE TEST RESULTS ARE TABULAR IN FORM, AND CONSIST OF READ SCORES FOR THE CURRENT DRIVE, USING DATA WRITTEN BY ALL OTHER DRIVES. THE SCORES ARE CONVERTED ON THE BASIS OF 12(DEC), (WITH A PERFECT SCORE = 12). ALSO, THE SCORES ARE LISTED SEPARATELY FOR EACH TRACK (SURFACE), FOR EACH OFFSET DIRECTION, AND AN OVERALL AVERAGE FOR ALL THREE TRACKS, ON THE CURRENT DRIVE, IS PRINTED.

IN THE FOLLOWING EXAMPLE, THERE ARE 2 SYSTEMS, AND THE DRIVES BEING TESTED ARE A0,A1,A2,B0, AND B5. THE TEST RESULTS FOR DRIVE A1 ARE SHOWN BELOW:

SCORES FOR DRIVE A1 (0-12 DEC.) :

TRACK NO.	DRIVE READ	OVWRT SCORE OFST-	OVWRT SCORE OFST+	READ SCORE OFST-	READ SCORE OFST+
0	A0	12	12	12	12
0	SELF	12	12	12	12
0	A2	* 7	* 6	12	12
0	B0	11	12	11	11
0	B5	9	12	11	11
1	A0	12	11	12	11
1	SELF	11	10	12	12
1	A2	12	11	11	12
1	B0	12	12	9	9
1	B5	12	12	11	12
2	A0	12	12	12	12
2	SELF	12	12	12	12
2	A2	11	9	11	11
2	B0	12	12	12	12
2	B5	12	12	12	12

DRIVE A1 OVWRT AVG = 12
DRIVE A1 READ AVG = 10

THE ABOVE EXAMPLE REVEALS A POSSIBLE COMPATIBILITY PROBLEM BETWEEN DRIVES A1 AND A2. NOTICE THAT ON TRACK 0, THAT THE OVERWRITE SCORES

H02

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 21
CZR6Q3.P11 02-DEC-77 10:46

SEQ 0020

870
871
872
873
874

WERE UNACCEPTABLY LOW (7 OR LESS), AND THE PROGRAM NOTED THESE BAD
SCORES WITH AN ASTERISK (*).

875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928

11.0 ERROR REPORTING

11.1 COMMON ERRORS

THE FOLLOWING IS A LIST OF COMMON ERROR MESSAGES WHICH ACCOMPANY ERROR TYPEOUTS FROM THE RK06 DRIVE COMPATIBILITY 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

929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983

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

11.2 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, POSSIBLY 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. OTHER ERROR REPORTS MAY CONSIST OF A SINGLE DESCRIPTIVE LINE.

11.3 ERROR PRINTOUT EXAMPLE

** WRITE CHECK ERROR

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					

PACK ADDRESS OF ERROR(S):

CYLNR	TRACK	SECTOR
000016	000001	000006

TABLE A

BASIC READ/WRITE TEST SECTORS

ADDRESS OF SECTOR ON EACH SURFACE (IN OCTAL)

DRIVE NO. -----	CYLINDER RK06/RK07 -----	SECTOR -----
0	620/1444	0
1	620/1444	20
2	620/1444	4
3	620/1444	6
4	620/1444	10
5	620/1444	12
6	620/1444	14
7	620/1444	16
10	622/1446	0
11	622/1446	2
12	622/1446	4
13	622/1446	6
14	622/1446	10
15	622/1446	12
16	622/1446	14
17	622/1446	16

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
1050
1051
1052
1053
1054
1055
1056
1057
1058

1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093

TABLE B

WORST CASE DATA PATTERN (REPEATS EVERY 16 WORDS)

WORD NO.	DATA (OCTAL)
0	072307
1	135143
2	156461
3	167230
4	073514
5	035646
6	016723
7	107351
10	143564
11	061672
12	030735
13	114356
14	046167
15	123073
16	151453
17	164616

1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130

TABLE C

CYLINDER BLOCK ASSIGNMENT FOR A GIVEN SURFACE

CURRENT ZONE - RANGE	OVERWRITE CYL BLK RANGE (OCT)	COMPATIBILITY CYL BLK RANGE (OCT)	ROTATING STARTING SECTOR	
1 - CYL 0-77	CYL 0-17	CYL 60-77	SECT 0	RK06
2 - 100-177	100-117	160-177	3	RK06
3 - 200-277	200-217	260-277	7	RK06
4 - 300-377	300-317	360-377	13	RK06
5 - 400-477	400-417	460-477	17	RK06
6 - 500-577	500-517	560-577	22	RK06
7 - 600-632	600-617	---	0	RK06
1 - CYL 0-177	CYL 0-17	CYL 160-177	SECT 0	RK07
2 - 200-377	200-217	360-377	3	RK07
3 - 400-577	400-417	560-577	7	RK07
4 - 600-777	600-617	760-777	13	RK07
5 - 1000-1177	1000-1017	1160-1177	17	RK07
6 - 1200-1377	1200-1217	1360-1377	22	RK07
7 - 1400-1456	1400-1417	---	0	RK07

TABLE D

BASIC CYLINDER BLOCK LAYOUT EXAMPLE

(SHOWN FOR RK06, USE CYL 160-177 FOR RK07)

CYLINDER NUMBERS (OCTAL)	SECTOR NUMBERS (OCTAL)															
	0	1	2	3	5	6	7	10	12	13	14	15	17	20	21	22
60	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
61	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	0
62	2	3	4	5	6	7	10	11	12	13	14	15	16	17	0	1
63	3	4	5	6	7	10	11	12	13	14	15	16	17	0	1	2
64	4	5	6	7	10	11	12	13	14	15	16	17	0	1	2	3
65	5	6	7	10	11	12	13	14	15	16	17	0	1	2	3	4
66	6	7	10	11	12	13	14	15	16	17	0	1	2	3	4	5
67	7	10	11	12	13	14	15	16	17	0	1	2	3	4	5	6
70	10	11	12	13	14	15	16	17	0	1	2	3	4	5	6	7
71	11	12	13	14	15	16	17	0	1	2	3	4	5	6	7	10
72	12	13	14	15	16	17	0	1	2	3	4	5	6	7	10	11
73	13	14	15	16	17	0	1	2	3	4	5	6	7	10	11	12
74	14	15	16	17	0	1	2	3	4	5	6	7	10	11	12	13
75	15	16	17	0	1	2	3	4	5	6	7	10	11	12	13	14
76	16	17	0	1	2	3	4	5	6	7	10	11	12	13	14	15
77	17	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16

TABLE E

OVERWRITE CYLINDERS

DRIVE #	CYLINDERS OVERWRITTEN (OCTAL)
0	0, 100, 200, 300, 400, 500, 600
1	1, 101, 201, 301, 401, 501, 601
2	2, 102, 202, 302, 402, 502, 602
3	3, 103, 203, 303, 403, 503, 603
4	4, 104, 204, 304, 404, 504, 604
5	5, 105, 205, 305, 405, 505, 605
6	6, 106, 206, 306, 406, 506, 606
7	7, 107, 207, 307, 407, 507, 607
10	10, 110, 210, 310, 410, 510, 610
11	11, 111, 211, 311, 411, 511, 611
12	12, 112, 212, 312, 412, 512, 612
13	13, 113, 213, 313, 413, 513, 613
14	14, 114, 214, 314, 414, 514, 614
15	15, 115, 215, 315, 415, 515, 615
16	16, 116, 216, 316, 416, 516, 616
17	17, 117, 217, 317, 417, 517, 617

NOTE: THE ABOVE TABLE GIVES CYLINDER NUMBERS FOR THE RK06.
REFER FOR TABLE C TO OBTAIN CORRESPONDING CYLINDER
NUMBERS FOR AN RK07.

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

TABLE F

SELF-TEST CYLINDERS

DRIVE #	CYLINDERS (WHERE SECTORS 4,11,16,23 ARE TESTED)
0	60,160,260,360,460,560
1	61,161,261,361,461,561
2	62,162,262,362,462,562
3	63,163,263,363,463,563
4	64,164,264,364,464,564
5	65,165,265,365,465,565
6	66,166,266,366,466,566
7	67,167,267,367,467,567
10	70,170,270,370,470,570
11	71,171,271,371,471,571
12	72,172,272,372,472,572
13	73,173,273,373,473,573
14	74,174,274,374,474,574
15	75,175,275,375,475,575
16	76,176,276,376,476,576
17	77,177,277,377,477,577

NOTE: THE ABOVE TABLE GIVES CYLINDER NUMBERS FOR THE RK06.
REFER TO TABLE C TO OBTAIN CORRESPONDING CYLINDER
NUMBERS FOR THE RK07.

1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248

1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284

TABLE G

PSEUDO-RANDOM DATA PATTERN

WORD #	DATA (OCTAL)
0	040135
1	177070
2	070414
3	064531
4	174473
5	062422
6	114352
7	036620
10	010031
11	052336
12	017310
13	011347
14	102367
15	152567
16	001246
17	160073

2

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
1340

.NLIST MC,MD,CND
.LIST ME
.ENABL ABS,AMA

*** REV 003 ***

SBTTL STARTING ADDRESSES

* 200 DEFAULT PARAMETERS AND RUN PASS 1 AND PASS 2
* 204 SELECT PARAMETERS AND RUN PASS 1
* 220 SELECT PARAMETERS AND RUN PASS 2
* 224 MEMORY DUMP ROUTINE

000001

STN=1
.TITLE CZR6Q80 RK6 DR CPT PROG
.*COPYRIGHT (C) 1977
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY DAVE HOFFMAN
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
.*
\$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

163000

\$SWR=163000
SBTTL OPERATIONAL SWITCH SETTINGS

* SWITCH USE
* -----
* 15 HALT ON ERROR
* 14 LOOP ON TEST
* 13 INHIBIT ERROR TYPEOUTS
* 12 REPORT DESCRIPTION ONLY, ON ERRORS
* 10 BELL ON ERROR
* 9 LOOP ON ERROR
* 8 APPLY RANDOM STALL BETWEEN OPERATIONS
* 7 TYPE BAD SECTOR FILES (BSF'S) AT START

SBTTL BASIC DEFINITIONS

001100

.*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

1341	000011	HT=	11	:: CODE FOR HORIZONTAL TAB
1342	000012	LF=	12	:: CODE FOR LINE FEED
1343	000015	CR=	15	:: CODE FOR CARRIAGE RETURN
1344	000200	CRLF=	200	:: CODE FOR CARRIAGE RETURN-LINE FEED
1345	177776	PS=	177776	:: PROCESSOR STATUS WORD
1346		.EQUIV	PS,PSW	
1347	177774	STKLMT=	177774	:: STACK LIMIT REGISTER
1348	177772	PIRQ=	177772	:: PROGRAM INTERRUPT REQUEST REGISTER
1349	177570	DSWR=	177570	:: HARDWARE SWITCH REGISTER
1350	177570	DDISP=	177570	:: HARDWARE DISPLAY REGISTER
1351				
1352		.*GENERAL PURPOSE REGISTER DEFINITIONS		
1353	000000	R0=	%0	:: GENERAL REGISTER
1354	000001	R1=	%1	:: GENERAL REGISTER
1355	000002	R2=	%2	:: GENERAL REGISTER
1356	000003	R3=	%3	:: GENERAL REGISTER
1357	000004	R4=	%4	:: GENERAL REGISTER
1358	000005	R5=	%5	:: GENERAL REGISTER
1359	000006	R6=	%6	:: GENERAL REGISTER
1360	000007	R7=	%7	:: GENERAL REGISTER
1361	000006	SP=	%6	:: STACK POINTER
1362	000007	PC=	%7	:: PROGRAM COUNTER
1363				
1364		.*PRIORITY LEVEL DEFINITIONS		
1365	000000	PR0=	0	:: PRIORITY LEVEL 0
1366	000040	PR1=	40	:: PRIORITY LEVEL 1
1367	000100	PR2=	100	:: PRIORITY LEVEL 2
1368	000140	PR3=	140	:: PRIORITY LEVEL 3
1369	000200	PR4=	200	:: PRIORITY LEVEL 4
1370	000240	PR5=	240	:: PRIORITY LEVEL 5
1371	000300	PR6=	300	:: PRIORITY LEVEL 6
1372	000340	PR7=	340	:: PRIORITY LEVEL 7
1373				
1374		.*"SWITCH REGISTER" SWITCH DEFINITIONS		
1375	100000	SW15=	100000	
1376	040000	SW14=	40000	
1377	020000	SW13=	20000	
1378	010000	SW12=	10000	
1379	004000	SW11=	4000	
1380	002000	SW10=	2000	
1381	001000	SW09=	1000	
1382	000400	SW08=	400	
1383	000200	SW07=	200	
1384	000100	SW06=	100	
1385	000040	SW05=	40	
1386	000020	SW04=	20	
1387	000010	SW03=	10	
1388	000004	SW02=	4	
1389	000002	SW01=	2	
1390	000001	SW00=	1	
1391		.EQUIV	SW09,SW9	
1392		.EQUIV	SW08,SW8	
1393		.EQUIV	SW07,SW7	
1394		.EQUIV	SW06,SW6	
1395		.EQUIV	SW05,SW5	
1396		.EQUIV	SW04,SW4	

```

1397      .EQUIV SW03,SW3
1398      .EQUIV SW02,SW2
1399      .EQUIV SW01,SW1
1400      .EQUIV SW00,SW0
1401
1402      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1403      BIT15= 100000
1404      BIT14= 40000
1405      BIT13= 20000
1406      BIT12= 10000
1407      BIT11= 4000
1408      BIT10= 2000
1409      BIT09= 1000
1410      BIT08= 400
1411      BIT07= 200
1412      BIT06= 100
1413      BIT05= 40
1414      BIT04= 20
1415      BIT03= 10
1416      BIT02= 4
1417      BIT01= 2
1418      BIT00= 1
1419      .EQUIV BIT09,BIT9
1420      .EQUIV BIT08,BIT8
1421      .EQUIV BIT07,BIT7
1422      .EQUIV BIT06,BIT6
1423      .EQUIV BIT05,BIT5
1424      .EQUIV BIT04,BIT4
1425      .EQUIV BIT03,BIT3
1426      .EQUIV BIT02,BIT2
1427      .EQUIV BIT01,BIT1
1428      .EQUIV BIT00,BIT0
1429
1430      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1431      ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
1432      RESVEC= 10     ;; RESERVED AND ILLEGAL INSTRUCTIONS
1433      TBITVEC= 14    ;; "T" BIT
1434      TRTVEC= 14     ;; TRACE TRAP
1435      BPTVEC= 14     ;; BREAKPOINT TRAP (BPT)
1436      IOTVEC= 20     ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
1437      PWRVEC= 24     ;; POWER FAIL
1438      EMTVEC= 30     ;; EMULATOR TRAP (EMT) **ERROR**
1439      TRAPVEC= 34    ;; "TRAP" TRAP
1440      TKVEC= 60      ;; TTY KEYBOARD VECTOR
1441      TPVEC= 64      ;; TTY PRINTER VECTOR
1442      PIRQVEC= 240   ;; PROGRAM INTERRUPT REQUEST VECTOR
1443
1444      .SBTTL TRAP CATCHER
1445
1446      .=0
1447      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1448      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1449      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1450      .=174
1451      000174 000000  DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
1452      000176 000000  SWREG: .WORD 0      ;; SOFTWARE SWITCH REGISTER

```

CZR6080 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 35
 CZR608.P11 02-DEC-77 10:46 STARTING ADDRESS(ES)

SEQ 0034

```

1453 .SBTTL STARTING ADDRESS(ES)
1454 000200 000137 012176 JMP 2#DFSTRT ;;JUMP TO STARTING ADDRESS OF PROGRAM
1455 000204 000204 =204
1456 000204 000137 012204 JMP 2#P1STRT
1457 000220 000220 =220
1458 000220 000137 012222 JMP 2#P2STRT
1459
1460 000224 000700 ..LOW: .WORD 700
1461 000226 001100 ..HIGH: .WORD 1100
1462
1463 .SBTTL ACT11 HOOKS
1464
1465 ;*****
1466 ;HOOKS REQUIRED BY ACT11
1467 000230 $SVPC=.; SAVE PC
1468 000046 =46
1469 000046 017264 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1470 000052 000052 =52
1471 000052 140000 .WORD 140000 ;;2)SET LOC.52 TO 140000
1472 000230 = $SVPC ;; RESTORE PC
1473 001000 =1000
1474 .SBTTL APT PARAMETER BLOCK
1475
1476 ;*****
1477 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1478 ;*****
1479 001000 . $X=.; SAVE CURRENT LOCATION
1480 000024 =24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1481 000024 000200 200 ;;FOR APT START UP
1482 000044 =44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1483 000044 001000 $APTHDR ;;POINT TO APT HEADER BLOCK
1484 001000 =. $X ;;RESET LOCATION COUNTER
1485 ;*****
1486 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1487 ;INTERFACE SPEC.
1488
1489 001000 $APTHD:
1490 001000 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1491 001002 001330 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1492 001004 001320 $TSTM: .WORD 1320 ;;RUN TIM OF LONGEST TEST
1493 001006 001546 $PASTM: .WORD 1546 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1494 001010 001546 $UNITM: .WORD 1546 ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1495 001012 000030 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1496 120210 AVECT1=120210 ;;RKVEC=210, RKPRI=5
1497 177440 ABASE=177440 ;;RKBAS ADR$
1498 000377 ADEVM=000377 ;;SET DEVICES 0-7 IN MAP

```

```

1499      .SBTTL  COMMON TAGS
1500
1501      ;;*****
1502      ;;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1503      ;;USED IN THE PROGRAM.
1504
1505      001100      .=1100
1506      001100      $CMTAG:      .WORD      0      ;;START OF COMMON TAGS
1507      001100      000000      .BYTE      0      ;;CONTAINS THE TEST NUMBER
1508      001102      000      .BYTE      0      ;;CONTAINS ERROR FLAG
1509      001103      000      .WORD      0      ;;CONTAINS SUBTEST ITERATION COUNT
1510      001104      000000      .WORD      0      ;;CONTAINS SCOPE LOOP ADDRESS
1511      001106      000000      .WORD      0      ;;CONTAINS SCOPE RETURN FOR ERRORS
1512      001110      000000      .WORD      0      ;;CONTAINS TOTAL ERRORS DETECTED
1513      001112      000000      .BYTE      0      ;;CONTAINS ITEM CONTROL BYTE
1514      001114      000      .BYTE      1      ;;CONTAINS MAX. ERRORS PER TEST
1515      001115      001      .WORD      0      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1516      001116      000000      .WORD      0      ;;CONTAINS ADDRESS OF 'GOOD' DATA
1517      001120      000000      .WORD      0      ;;CONTAINS ADDRESS OF 'BAD' DATA
1518      001122      000000      .WORD      0      ;;CONTAINS 'GOOD' DATA
1519      001124      000000      .WORD      0      ;;CONTAINS 'BAD' DATA
1520      001126      000000      .WORD      0      ;;RESERVED--NOT TO BE USED
1521      001130      000000      .WORD      0
1522      001132      000000      .WORD      0
1523      001134      000      .WORD      0      ;;AUTOMATIC MODE INDICATOR
1524      001135      000      .BYTE      0      ;;INTERRUPT MODE INDICATOR
1525      001136      000000      .WORD      0
1526      001140      177570      .WORD      0      ;;ADDRESS OF SWITCH REGISTER
1527      001142      177570      .WORD      0      ;;ADDRESS OF DISPLAY REGISTER
1528      001144      177560      .WORD      0      ;;TTY KBD STATUS
1529      001146      177562      .WORD      0      ;;TTY KBD BUFFER
1530      001150      177564      .WORD      0      ;;TTY PRINTER STATUS REG. ADDRESS
1531      001152      177566      .WORD      0      ;;TTY PRINTER BUFFER REG. ADDRESS
1532      001154      000      .BYTE      0      ;;CONTAINS NULL CHARACTER FOR FILLS
1533      001155      002      .BYTE      2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1534      001156      012      .BYTE      12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
1535      001157      000      .BYTE      0      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1536      001160      000000      .WORD      0      ;;CONTAINS THE ADDRESS FROM
1537      001162      000000      .WORD      0      ;;WHICH ($REGAD) WAS OBTAINED
1538      001164      000000      .WORD      0      ;;CONTAINS (($REGAD)+0)
1539      001166      000000      .WORD      0      ;;CONTAINS (($REGAD)+2)
1540      001170      000000      .WORD      0      ;;CONTAINS (($REGAD)+4)
1541      001172      000000      .WORD      0      ;;CONTAINS (($REGAD)+6)
1542      001174      000000      .WORD      0      ;;CONTAINS (($REGAD)+10)
1543      001176      000000      .WORD      0      ;;CONTAINS (($REGAD)+12)
1544      001200      000000      .WORD      0      ;;CONTAINS (($REGAD)+14)
1545      001202      000000      .WORD      0      ;;CONTAINS (($REGAD)+16)
1546      001204      000000      .WORD      0      ;;CONTAINS (($REGAD)+20)
1547      001206      000000      .WORD      0      ;;CONTAINS (($REGAD)+22)
1548      001210      000000      .WORD      0      ;;CONTAINS (($REGAD)+24)
1549      001212      000000      .WORD      0      ;;CONTAINS (($REGAD)+26)
1550      001214      000000      .WORD      0      ;;CONTAINS (($REGAD)+30)
1551      001216      000000      .WORD      0      ;;CONTAINS (($REGAD)+32)
1552      001220      000000      .WORD      0      ;;CONTAINS (($REGAD)+34)
1553      001222      000000      .WORD      0      ;;CONTAINS (($REGAD)+36)
1554      001224      000000      .WORD      0      ;;CONTAINS (($REGAD)+40)

```

K03

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 37
CZR6QB.P11 02-DEC-77 10:46 COMMON TAGS

SEQ 0036

```

1555 001224 000000 $REG21: .WORD 0 ; CONTAINS (($REGAD)+42)
1556 001226 000000 $REG22: .WORD 0 ; CONTAINS (($REGAD)+44)
1557 001230 000000 $REG23: .WORD 0 ; CONTAINS (($REGAD)+46)
1558 001232 000000 $REG24: .WORD 0 ; CONTAINS (($REGAD)+50)
1559 001234 000000 $REG25: .WORD 0 ; CONTAINS (($REGAD)+52)
1560 001236 000000 $REG26: .WORD 0 ; CONTAINS (($REGAD)+54)
1561 001240 000000 $REG27: .WORD 0 ; CONTAINS (($REGAD)+56)
1562 001242 000000 $REG30: .WORD 0 ; CONTAINS (($REGAD)+60)
1563 001244 000000 $REG31: .WORD 0 ; CONTAINS (($REGAD)+62)
1564 001246 000000 $REG32: .WORD 0 ; CONTAINS (($REGAD)+64)
1565 001250 000000 $REG33: .WORD 0 ; CONTAINS (($REGAD)+66)
1566 001252 000000 $REG34: .WORD 0 ; CONTAINS (($REGAD)+70)
1567 001254 000000 $REG35: .WORD 0 ; CONTAINS (($REGAD)+72)
1568 001256 000000 $REG36: .WORD 0 ; CONTAINS (($REGAD)+74)
1569 001260 000000 $REG37: .WORD 0 ; CONTAINS (($REGAD)+76)
1570 001262 000000 $TMP0: .WORD 0 ; USER DEFINED
1571 001264 000000 $TMP1: .WORD 0 ; USER DEFINED
1572 001266 000000 $TMP2: .WORD 0 ; USER DEFINED
1573 001270 000000 $TMP3: .WORD 0 ; USER DEFINED
1574 001272 000000 $TMP4: .WORD 0 ; USER DEFINED
1575 001274 000000 $TMP5: .WORD 0 ; USER DEFINED
1576 001276 000000 $TMP6: .WORD 0 ; USER DEFINED
1577 001300 000000 $TMP7: .WORD 0 ; USER DEFINED
1578 001302 000000 $TMP10: .WORD 0 ; USER DEFINED
1579 001304 000000 $TMP11: .WORD 0 ; USER DEFINED
1580 001306 000000 $TMP12: .WORD 0 ; USER DEFINED
1581 001310 000000 $TMP13: .WORD 0 ; USER DEFINED
1582 001312 000000 $TMP14: .WORD 0 ; USER DEFINED
1583 001314 000000 $TMP15: .WORD 0 ; USER DEFINED
1584 001316 000000 $ESCAPE: 0 ; ESCAPE ON ERROR ADDRESS
1585 001320 177607 000377 $BELL: .ASCIZ <207><377><377> ; CODE FOR BELL
1586 001324 077 $QUES: .ASCII /?/ ; QUESTION MARK
1587 001325 015 $CRLF: .ASCII <15> ; CARRIAGE RETURN
1588 001326 000012 $LF: .ASCIZ <12> ; LINE FEED
1589 ;*****
1590 ;SBTTL APT MAILBOX-ETABLE
1591 ;*****
1592 ;*****
1593 ;*****
1594 001330 $MAIL: ; APT MAILBOX
1595 001330 000000 $MSGTY: .WORD AMSGTY ; MESSAGE TYPE CODE
1596 001332 000000 $FATAL: .WORD AFATAL ; FATAL ERROR NUMBER
1597 001334 000000 $TESTN: .WORD ATESTN ; TEST NUMBER
1598 001336 000000 $PASS: .WORD APASS ; PASS COUNT
1599 001340 000000 $DEVCT: .WORD ADEVCT ; DEVICE COUNT
1600 001342 000000 $UNIT: .WORD AUNIT ; I/O UNIT NUMBER
1601 001344 000000 $MSGAD: .WORD AMSGAD ; MESSAGE ADDRESS
1602 001346 000000 $MSGLG: .WORD AMSGLG ; MESSAGE LENGTH
1603 001350 $ETABLE: ; APT ENVIRONMENT TABLE
1604 001350 000 $ENV: .BYTE AENV ; ENVIRONMENT BYTE
1605 001351 000 $ENVM: .BYTE AENVM ; ENVIRONMENT MODE BITS
1606 001352 000000 $SWREG: .WORD ASWREG ; APT SWITCH REGISTER
1607 001354 000000 $USWR: .WORD AUSWR ; USER SWITCHES
1608 001356 000000 $CPUOP: .WORD ACPUOP ; CPU TYPE, OPTIONS
1609 ;*
1610 ;*

```

BITS 15-11=CPU TYPE
 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05

```

1611      ;*
1612      ;*
1613      ;*
1614      ;*
1615      001360      000
1616      001361      000
1617      ;*
1618      ;*
1619      ;*
1620      ;*
1621      001362      000000
1622      ;*
1623      001364      000
1624      001365      000
1625      001366      000000
1626      001370      000
1627      001371      000
1628      001372      000000
1629      001374      000
1630      001375      000
1631      001376      000000
1632      001400      120210
1633      001402      000000
1634      001404      177440
1635      001406      000377
1636      001410
1637

;*
;*
;*
;*
$MAMS1: .BYTE      AMAMS1
$MTYP1: .BYTE      AMTYP1
;*
;*
;*
;*
$MADR1: .WORD      AMADR1
;*
$MAMS2: .BYTE      AMAMS2
$MTYP2: .BYTE      AMTYP2
$MADR2: .WORD      AMADR2
$MAMS3: .BYTE      AMAMS3
$MTYP3: .BYTE      AMTYP3
$MADR3: .WORD      AMADR3
$MAMS4: .BYTE      AMAMS4
$MTYP4: .BYTE      AMTYP4
$MADR4: .WORD      AMADR4
$VECT1: .WORD      AVECT1
$VECT2: .WORD      AVECT2
$BASE:  .WORD      ABASE
$DEVM:  .WORD      ADEVM
$ETEND:
.MEXIT

11/70=06,PDQ=07,Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
;;HIGH ADDRESS,M.S. BYTE
;;MEM. TYPE,BLK#1
MEM.TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
;;HIGH ADDRESS,BLK#1
MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
;;HIGH ADDRESS,M.S. BYTE
;;MEM. TYPE,BLK#2
MEM.LAST ADDRESS,BLK#2
;;HIGH ADDRESS,M.S. BYTE
;;MEM. TYPE,BLK#3
MEM.LAST ADDRESS,BLK#3
;;HIGH ADDRESS,M.S. BYTE
;;MEM. TYPE,BLK#4
MEM.LAST ADDRESS,BLK#4
;;INTERRUPT VECTOR#1,BUS PRIORITY#1
;;INTERRUPT VECTOR#2,BUS PRIORITY#2
;;BASE ADDRESS OF EQUIPMENT UNDER TEST
;;DEVICE MAP

```

ERROR POINTER TABLE

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;	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	;	D TO C PARITY ERROR
EM6			
DH100			
DT100			
DF03			

;	ERROR 7	;	DRIVE DETECTED PARITY ERROR
EM7			
DH100			
DT100			

1638
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652 001410
 1653
 1654
 1655 001410 045417
 1656 001412 050512
 1657 001414 052276
 1658 001416 052406
 1659
 1660
 1661 001420 045443
 1662 001422 050512
 1663 001424 052276
 1664 001426 052406
 1665
 1666
 1667 001430 045467
 1668 001432 050512
 1669 001434 052276
 1670 001436 052406
 1671
 1672
 1673 001440 045512
 1674 001442 050512
 1675 001444 052276
 1676 001446 052406
 1677
 1678
 1679 001450 045533
 1680 001452 050512
 1681 001454 052276
 1682 001456 052436
 1683
 1684
 1685 001460 045552
 1686 001462 050512
 1687 001464 052276
 1688 001466 052472
 1689
 1690
 1691 001470 045576
 1692 001472 050512
 1693 001474 052276

1694	001476	052472	DF03	
1695				
1696			:ERROR 10	
1697	001500	045632	EM10	;AC LOW
1698	001502	050512	DH100	
1699	001504	052276	DT100	
1700	001506	052436	DF02	
1701				
1702			:ERROR 11	
1703	001510	045641	EM11	;SPEED LOSS
1704	001512	050512	DH100	
1705	001514	052276	DT100	
1706	001516	052436	DF02	
1707				
1708			:ERROR 12	
1709	001520	045654	EM12	;ILLEGAL FUNCTION
1710	001522	050512	DH100	
1711	001524	052276	DT100	
1712	001526	052436	DF02	
1713				
1714			:ERROR 13	
1715	001530	045675	EM13	;PROGRAMMING ERROR
1716	001532	050512	DH100	
1717	001534	052276	DT100	
1718	001536	052406	DF01	
1719				
1720			:ERROR 14	;NON-EXISTANT FUNCTION
1721	001540	045717	EM14	
1722	001542	050512	DH100	
1723	001544	052276	DT100	
1724	001546	052436	DF02	
1725				
1726			:ERROR 15	
1727	001550	045745	EM15	;DRIVE TYPE ERROR
1728	001552	050512	DH100	
1729	001554	052276	DT100	
1730	001556	052436	DF02	
1731				
1732			:ERROR 16	
1733	001560	045766	EM16	;FORMAT ERROR
1734	001562	050512	DH100	
1735	001564	052276	DT100	
1736	001566	052436	DF02	
1737				
1738			:ERROR 17	
1739	001570	046003	EM17	;WRITE LOCK ERROR
1740	001572	050512	DH100	
1741	001574	052276	DT100	
1742	001576	052436	DF02	
1743				
1744			:ERROR 20	
1745	001600	046024	EM20	;DRIVE UNSAFE
1746	001602	050512	DH100	
1747	001604	052276	DT100	
1748	001606	052436	DF02	
1749				

1750			:ERROR 21	
1751	001610	046041	EM21	;SEEK INCOMPLETE
1752	001612	050512	DH100	
1753	001614	052276	DT100	
1754	001616	052436	DF02	
1755				
1756			:ERROR 22	
1757	001620	046061	EM22	;CYLINDER OVERFLOW
1758	001622	050512	DH100	
1759	001624	052276	DT100	
1760	001626	052436	DF02	
1761				
1762			:ERROR 23	
1763	001630	046103	EM23	;ILLEGAL CYLINDER
1764	001632	050512	DH100	
1765	001634	052276	DT100	
1766	001636	052436	DF02	
1767				
1768			:ERROR 24	
1769	001640	046134	EM24	;DRIVE OFF TRACK
1770	001642	050512	DH100	
1771	001644	052276	DT100	
1772	001646	052436	DF02	
1773				
1774			:ERROR 25	
1775	001650	046154	EM25	;DRIVE TIMING ERROR
1776	001652	050512	DH100	
1777	001654	052276	DT100	
1778	001656	052436	DF02	
1779				
1780			:ERROR 26	
1781	001660	046177	EM26	;DATA LATE
1782	001662	050512	DH100	
1783	001664	052276	DT100	
1784	001666	052436	DF02	
1785				
1786			:ERROR 27	
1787	001670	046211	EM27	;CONTROLLER TIMEOUT
1788	001672	050512	DH100	
1789	001674	052276	DT100	
1790	001676	052436	DF02	
1791				
1792			:ERROR 30	
1793	001700	046234	EM30	;OPERATION INCOMPLETE
1794	001702	050512	DH100	
1795	001704	052276	DT100	
1796	001706	052546	DF05	
1797				
1798			:ERROR 31	
1799	001710	046261	EM31	;HEADER VRC ERROR
1800	001712	050512	DH100	
1801	001714	052276	DT100	
1802	001716	052546	DF05	
1803				
1804			:ERROR 32	
1805	001720	046302	EM32	;DATA CHECK ERROR

1806	001722	050512	DH100	
1807	001724	052276	DT100	
1808	001726	052602	DF07	
1809				
1810			.ERROR 33	
1811	001730	046323	EM33	;WRITE CHECK ERROR
1812	001732	050512	DH100	
1813	001734	052276	DT100	
1814	001736	052636	DF10	
1815				
1816			.ERROR 34	
1817	001740	046345	EM34	;DATA MISCOMPARE(S)
1818	001742	050512	DH100	
1819	001744	052276	DT100	
1820	001746	052532	DF04	
1821				
1822			.ERROR 35	
1823	001750	046365	EM35	;NO DRIVE RESPONSE-UFE & NXD
1824	001752	050512	DH100	
1825	001754	052276	DT100	
1826	001756	052406	DF01	
1827				
1828			.ERROR 36	
1829	001760	046421	EM36	;DRIVE ERROR WILL NOT CLEAR
1830	001762	000000	0	
1831	001764	000000	0	
1832	001766	000000	0	
1833				
1834			.ERROR 37	
1835	001770	046454	EM37	;DRIVE STATUS CHANGE WILL NOT CLEAR
1836	001772	000000	0	
1837	001774	000000	0	
1838	001776	000000	0	
1839				
1840			.ERROR 40	
1841	002000	046517	EM40	;ATTENTION BUT NO STATUS CHANGE OR FAULT
1842	002002	050512	DH100	
1843	002004	052276	DT100	
1844	002006	052436	DF02	
1845				
1846			.ERROR 41	
1847	002010	046563	EM41	;ATTENTION BUT DRIVE NOT AVAILABLE
1848	002012	050512	DH100	
1849	002014	052276	DT100	
1850	002016	052436	DF02	
1851				
1852			.ERROR 42	
1853	002020	046621	EM42	;ATTENTION WHEN NOT EXPECTED
1854	002022	050512	DH100	
1855	002024	052276	DT100	
1856	002026	052436	DF02	
1857				
1858			.ERROR 43	
1859	002030	046651	EM43	;ERROR WHILE GATHERING DRIVE STATUS
1860	002032	050512	DH100	
1861	002034	052276	DT100	

1862	002036	052702	DF12	
1863				
1864			:ERROR 44	
1865	002040	047126	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
1866	002042	050512	DH100	
1867	002044	052276	DT100	
1868	002046	052702	DF12	
1869				
1870			:ERROR 45	
1871	002050	047173	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
1872	002052	050512	DH100	
1873	002054	052276	DT100	
1874	002056	052702	DF12	
1875				
1876			:ERROR 46	
1877	002060	047236	EM65	;UNSOLICITED ATTENTION
1878	002062	050512	DH100	
1879	002064	052276	DT100	
1880	002066	052702	DF12	
1881				
1882			:ERROR 47	
1883	002070	047264	EM66	;UNEXPECTED DATA TYPE ERROR
1884	002072	050512	DH100	
1885	002074	052276	DT100	
1886	002076	052702	DF12	
1887				
1888			:ERROR 50	
1889	002100	047317	EM67	;ATTENTION DID NOT RESET WITH CLEAR
1890	002102	050512	DH100	
1891	002104	052276	DT100	
1892	002106	052702	DF12	
1893				
1894			:ERROR 51	
1895	002110	047356	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1896	002112	050512	DH100	
1897	002114	052276	DT100	
1898	002116	052702	DF12	
1899				
1900			:ERROR 52	
1901	002120	046706	EM52	;MULTIPLE DRIVE SELECT
1902	002122	050512	DH100	
1903	002124	052276	DT100	
1904	002126	052702	DF12	
1905				
1906			:ERROR 53	
1907	002130	046734	EM53	;ABREVIATED HCE ERROR
1908	002132	050512	DH100	
1909	002134	052276	DT100	
1910	002136	052732	DF13	
1911				
1912			:ERROR 54	
1913	002140	046234	EM30	;OPERATION INCOMPLETE ERROR
1914	002142	050512	DH100	
1915	002144	052276	DT100	
1916	002146	052762	DF14	
1917				

1918			:ERROR 55	
1919	002150	046261	EM31	;ABREVIATED HVRC ERROR
1920	002152	050512	DH100	
1921	002154	052276	DT100	
1922	002156	052732	DF13	
1923				
1924			:ERROR 56	
1925	002160	046761	EM56	;2 TIMEOUT ERROR
1926	002162	050512	DH100	
1927	002164	052276	DT100	
1928	002166	053012	DF15	
1929				
1930			:ERROR 57	;2ND LEVEL IN SUBSYSTEM TIMEOUT
1931	002170	046761	EM56	
1932	002172	050512	DH100	
1933	002174	052276	DT100	
1934	002176	053056	DF16	
1935				
1936			:ERROR 60	
1937	002200	047000	EM60	;ERROR IN RECAL FOR RECOVERY
1938	002202	000000	0	
1939	002204	000000	0	
1940	002206	000000	0	
1941				
1942			:ERROR 61	
1943	002210	047034	EM61	;ABORT MESSAGE
1944	002212	000000	0	
1945	002214	000000	0	
1946	002216	000000	0	
1947				
1948			:ERROR 62	
1949	002220	047102	EM62	;CYLINDER MISCOMPARE
1950	002222	050512	DH100	
1951	002224	052276	DT100	
1952	002226	053122	DF17	
1953				
1954			:ERROR 63	;DATA ERROR WORDS
1955	002230	000000	0	
1956	002232	000000	0	
1957	002234	052370	DT602	
1958	002236	053232	DF25	
1959				
1960			:ERROR 64	
1961	002240	047126	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
1962	002242	050512	DH100	
1963	002244	052276	DT100	
1964	002246	052436	DF02	
1965				
1966			:ERROR 65	
1967	002250	047173	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
1968	002252	050512	DH100	
1969	002254	052276	DT100	
1970	002256	052436	DF02	
1971				
1972			:ERROR 66	
1973	002260	047236	EM65	;UNSOLICITED ATTENTION

1974	002262	050512	DH100	
1975	002264	052276	DT100	
1976	002266	052436	DF02	
1977				
1978			:ERROR 67	
1979	002270	047264	EM66	;UNEXPECTED DATA TYPE ERROR
1980	002272	050512	DH100	
1981	002274	052276	DT100	
1982	002276	052436	DF02	
1983				
1984			:ERROR 70	
1985	002300	047317	EM67	;ATTENTION DID NOT RESET WITH CLEAR
1986	002302	050512	DH100	
1987	002304	052276	DT100	
1988	002306	052436	DF02	
1989				
1990			:ERROR 71	
1991	002310	047356	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR ATT
1992	002312	050512	DH100	
1993	002314	052276	DT100	
1994	002316	052436	DF02	
1995				
1996			:ERROR 72	
1997	002320	047425	EM71	;DATA LATE WHEN UNLOADING HEADER
1998	002322	050512	DH100	
1999	002324	052276	DT100	
2000	002326	052436	DF02	
2001				
2002			:ERROR 73	
2003	002330	047465	EM72	;CONTROLLER ERROR DURING DRIVER SERVICE
2004	002332	050512	DH100	
2005	002334	052276	DT100	
2006	002336	052436	DF02	
2007				
2008			:ERROR 74	
2009	002340	047534	EM73	;DRIVE DETECTED PARITY ERROR
2010	002342	050512	DH100	
2011	002344	052276	DT100	
2012	002346	052436	DF02	
2013				
2014			:ERROR 75	
2015	002350	047570	EM74	;UNDEFINED ERROR
2016	002352	050512	DH100	
2017	002354	052276	DT100	
2018	002356	052436	DF02	
2019				
2020			:ERROR 76	
2021	002360	047610	EM75	;MARKING SECTOR BAD MESSAGE
2022	002362	000000	0	
2023	002364	000000	0	
2024	002366	000000	0	
2025				
2026			:ERROR 77	
2027	002370	047640	EM76	;BAD DATA VERIFICATION WITH READ
2028	002372	051477	DH605	
2029	002374	052362	DT601	

G04

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 46
 CZR6Q8.P11 02-DEC-77 10:46 ERROR POINTER TABLE

SEQ 0045

2030	002376	053156	DF21	
2031				
2032			:ERROR 100	
2033	002400	047722	EM77	;RETRY SUCCESSFUL MESSAGE
2034	002402	000000	0	
2035	002404	052362	DT601	
2036	002406	053216	DF23	
2037				
2038			:ERROR 101	
2039	002410	047722	EM77	;ANOTHER RETRY SUCCESSFUL MESSAGE
2040	002412	000000	0	
2041	002414	052362	DT601	
2042	002416	053216	DF23	
2043				
2044			:ERROR 102	
2045	002420	047743	EM100	;RETRY UNSUCCESSFUL MESSAGE
2046	002422	000000	0	
2047	002424	052362	DT601	
2048	002426	053216	DF23	
2049				
2050			:ERROR 103	
2051	002430	047766	EM101	;NO VALID HEADERS IN TRACK JUST READ
2052	002432	051461	DH6042	
2053	002434	052362	DT601	
2054	002436	053226	DF24	
2055				
2056			:ERROR 104	
2057	002440	050044	EM102	;BSE ERROR ON SECTOR NOT LISTED AS BAD
2058	002442	050512	DH100	
2059	002444	052276	DT100	
2060	002446	052436	DF02	
2061				
2062			:ERROR 105	
2063	002450	050116	EM103	;TIMED-OUT ON READ HEADER
2064	002452	050512	DH100	
2065	002454	052276	DT100	
2066	002456	052472	DF03	
2067				
2068			:ERROR 106	
2069	002460	050144	EM104	;TIMED-OUT ON SEEK
2070	002462	050512	DH100	
2071	002464	052276	DT100	
2072	002466	052472	DF03	
2073				
2074			:ERROR 107	
2075	002470	050166	EM105	;DRIVE SIEZED BY OTHER PORT
2076	002472	050512	DH100	
2077	002474	052276	DT100	
2078	002476	052436	DF02	
2079				
2080			:ERROR 110	
2081	002500	050221	EM106	; "DATA MISCMPR WHILE BAI SET"
2082	002502	050512	DH100	
2083	002504	052276	DT100	
2084	002506	053246	DF27	
2085				

H04

SEQ 0046

ERROR POINTER TABLE

2086			.ERROR 111	
2087	002510	050254	EM107	;"NO NEM WHEN EXPECTED"
2088	002512	000000	0	
2089	002514	000000	0	
2090	002516	000000	0	
2091				
2092			.ERROR 112	
2093	002520	050321	EM110	;"INTRPT WHEN CNTRLR NOT READY"
2094	002522	050512	DH100	
2095	002524	052276	DT100	
2096	002526	052406	DF01	
2097				
2098			.ERROR 113	
2099	002530	050354	EM111	;"NO ATT'N ON SEEK"
2100	002532	050512	DH100	
2101	002534	052276	DT100	
2102	002536	052436	DF02	
2103				
2104			.ERROR 114	
2105	002540	050375	EM112	;"DRIVE'S CYLINDER INCORRECT"
2106	002542	051706	DH702	
2107	002544	052336	DT202	
2108	002546	053236	DF26	
2109				
2110			.ERROR 115	
2111	002550	000000	0	;"TYPE ADRS OF DATA MISCOMPARE(S)"
2112	002552	000000	0	
2113	002554	052362	DT601	
2114	002556	052662	DF11	
2115				
2116			.ERROR 116	
2117	002560	046345	EM34	;"DATA MISCOMPARE (11/70)"
2118	002562	051023	DH103	
2119	002564	000000	0	
2120	002566	053146	DF20	
2121				
2122			.ERROR 117	
2123	002570	000000	0	;"PART OF DATA MISCOMPARE"
2124	002572	000000	0	
2125	002574	052276	DT100	
2126	002576	053166	DF22	
2127				
2128			.ERROR 120	
2129	002600	050430	EM113	;"ABORT- CAN'T READ BSF"
2130	002602	050512	DH100	
2131	002604	052276	DT100	
2132	002606	052406	DF01	
2133				
2134			.ERROR 121	
2135	002610	050456	EM114	;"KT11 FAILURE"
2136	002612	050512	DH100	
2137	002614	052276	DT100	
2138	002616	053272	DF30	
2139				
2140			.ERROR 122	
2141	002620	050473	EM115	;"MEM PARITY ERROR"

CZR6080 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 48
 CZR608.P11 02-DEC-77 10:46 ERROR POINTER TABLE

SEQ 0047

2142	002622	050512	DH100	
2143	002624	052276	DT100	
2144	002626	053316	DF31	
2145				
2146			ERROR 123	
2147	002630	045467	EM3	;NED ON SIZING UNDER APT
2148	002632	000000	0	
2149	002634	000000	0	
2150	002636	000000	0	
2151				
2152				
2153				
2154				
2155				
2156				
2157				
2158				
2159				
2160				
2161				
2162				
2163				
2164				
2165				
2166				
2167				
2168				
2169				
2170				

J04

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 49
BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0048

.SBTTL BIT ASSIGNMENTS IN THE RK611 REGISTERS

RKCS1							
15	14	13	12	11	10	9	8
CERR	DI	DCPAR	CFMT	CTO	CDT	BA17	BA16
R/W	RO	RO	R/W	RO	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RDY	IE	SPARE	F4	F3	F2	F1	GO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RKCS2							
15	14	13	12	11	10	9	8
DLT	WCE	UPE	NED	NEM	PGE	MDS	UFE
RO	RO	RO	RO	RO	RO	RO	RO
7	6	5	4	3	2	1	0
OR	IR	CLR	BAI	DESL	DS2	DS1	DS0
RO	RO	WO	R/W	R/W	R/W	R/W	R/W
RKDC (DESIRED CYLINDER)							
15	14	13	12	11	10	9	8
NU	NU	NU	NU	NU	NU	DC9	DC8
READ ONLY							
7	6	5	4	3	2	1	0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
READ ONLY							
RKDA (DESIRED TRACK AND SECTOR)							
15	14	13	12	11	10	9	8
NU	UN	UN	UN	UN	TA2	TA1	TA0
READ ONLY							
7	6	5	4	3	2	1	0
UN	UN	UN	SA4	SA3	SA2	SA1	SA0
READ ONLY							

2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224

CZR6Q80 RK6 DF CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 50
CZR6Q8.P11 02-DEC-77 10:46 BIT ASSIGNMENTS IN THE RK611 REGISTERS

CZR6QBD RK6 DF CPT PROG MACY11 30(1046)
CZR6QB.P11 02-DEC-77 10:46

```

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
                        !ALWYSO
  -----

```

[illegible]

RKER (ERROR REGISTER)							
15	14	13	12	11	10	9	8
DCK	UNS	OPI	DTE	WLE	IDAE	COE	HVRC
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							
RKMR1 (MAINTENANCE REG 1)							
15	14	13	12	11	10	9	8
RD GATE	WRT GATE	ECCW	PCD	PCA	MEWD	MERD	MCLK
READ ONLY						READ/WRITE	
7	6	5	4	3	2	1	0
MIND	MSP	DMD	PAT	MS3	MS2	MS1	MS0
READ/WRITE							

M04

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 52
CZR6Q8.P11 02-DEC-77 10:46 BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0051

```

;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 !

```

2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352

NO4

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 53
CZR6QB.P11 02-DEC-77 10:46 BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0052

2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390

LINE B MESSAGE 00							
15	14	13	12	11	10	9	8
PAR	UNS	DROT	DCLC	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
	HDS		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					

2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446

LINE A MESSAGE							
15	14	13	12	11	10	9	8
PAR ! NU ! NU ! CYLINDER DIFF/OFFSET VALUE !							
7	6	5	4	3	2	1	0
CYLINDER DIFF/OFFSET VALUE! NU ! DRIVE SELECT CODE !							
LINE B MESSAGE							
15	14	13	12	11	10	9	8
PAR ! ALIGN! NU ! CYLINDER ADDRESS !							
SIGN !							
7	6	5	4	3	2	1	0
CYLINDER ADDRESS ! UN ! UN ! 1 ! 0 !							
LINE A MESSAGE							
15	14	13	12	11	10	9	8
PAR ! DRIVE SERIAL NUMBER !							
7	6	5	4	3	2	1	0
DRIVE SERIAL NUMBER ! DRIVE SELECT CODE !							
LINE B MESSAGE							
15	14	13	12	11	10	9	8
PAR ! NU ! NU ! DECODED HEAD ! SECTOR!							
ADDRESS ! COUNT !							
7	6	5	4	3	2	1	0
SECTOR COUNT ! UN ! UN ! 1 ! 1 !							

.SBTTL RK06 CONTROLLER REGISTER DEFINITION

000000
000002
000004
000006
000010

RKCS1= 0 ; CONTROL AND STATUS REGISTER 1
RKWC= 2 ; WORD COUNT REGISTER
RKBA= 4 ; BUS ADDRESS REGISTER
RKDR= 6 ; DESIRED TRACK SECTOR REGISTER
RKCS2= 10 ; CONTROL AND STATUS REGISTER 2

```

2447      000012      RKDS= 12      ;DRIVE STATUS REGISTER
2448      000014      RKER= 14      ;ERROR REGISTER
2449      000016      RKASOF= 16     ;ATTENTION SUMMARY AND OFFSET REGISTER
2450      000020      RKDC= 20      ;DESIRED CYLINDER REGISTER
2451      000020      RKDCYL= 20    ;DESIRED CYLINDER REGISTER
2452      000024      RKDB= 24      ;DATA BUFFER
2453      000026      RKMR1= 26     ;MAINTENANCE REGISTER 1
2454      000034      RKMR2= 34     ;MAINTENANCE REGISTER 2
2455      000036      RKMR3= 36     ;MAINTENANCE REGISTER 3
2456      000030      RKPOS= 30     ;ECC POSITION INFORMATION
2457      000030      RKECPS= 30    ;ECC POSITION INFORMATION
2458      000032      RKPAT= 32     ;ECC PATTERN INFORMATION
2459      000032      RKECPT= 32    ;ECC PATTERN INFORMATION
2460
2461      .SBTTL DRIVE COMMANDS
2462
2463      000101      SELDRV= 101     ;SELECT DRIVE
2464      000103      PACK= 103      ;PACK ACKNOWLEDGE
2465      000105      CLEAR= 105     ;DRIVE CLEAR
2466      000107      UNLOAD= 107    ;UNLOAD
2467      000111      SRTSPL= 111    ;START SPINDLE
2468      000113      RECAL= 113     ;RECALIBRATE
2469      000115      OFFSET= 115    ;OFFSET
2470      000117      SEEK= 117     ;SEEK
2471      000121      RDATA= 121     ;READ DATA
2472      000123      WRDATA= 123    ;WRITE DATA
2473      000125      RDHEAD= 125    ;READ HEADER
2474      000127      WRHEAD= 127    ;WRITE HEADER AND DATA
2475      000131      WRTCHK= 131    ;WRITE CHECK
2476
2477      ;          THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
2478      ;          TO SIMULATE A SPECIFIC DESIRED OPERATION
2479
2480      000140      RELEAS= 140      ;RELEASE DRIVE
2481      000141      RDSTAT= 141    ;GET ALL STATUS FROM DRIVE
2482      000164      RDALHD= 164     ;READ ALL HEADERS
2483      000176      CONCLR= 176     ;CONTROLLER CLEAR (BIT 15 OF CS1)
2484      000177      SUBCLR= 177    ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
2485      000300      INTR= 300       ;GENERATE INTERRUPT TO CPU
2486
2487      ;          DRIVER ISSUED SERVICE COMMANDS
2488
2489      000001      DR.SEL= 001      ;DRIVE SELECT
2490      000005      DR.CLR= 005     ;DRIVE CLEAR
2491
2492      .SBTTL CONTROL AND STATUS REGISTER 1 BITS
2493
2494      000001      GO= BIT0         ;GO BIT
2495      000100      IE= BIT6         ;INTERRUPT ENABLE
2496      000200      RDY= BIT7       ;CONTROLLER READY
2497      000400      BA16= BIT8      ;BUS ADDRESS BIT 16
2498      001000      BA17= BIT9      ;BUS ADDRESS BIT 17
2499      002000      CDT= BIT10      ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
2500      004000      CTO= BIT11     ;CONTROLLER TIMED OUT WAITING FOR
2501                                     ;DRIVE RESPONSE
2502      010000      CFMT= BIT12     ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

```

```

2503      020000      SPAR=   BIT13      ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
2504      040000      DI=     BIT14      ;DRIVE INTERRUPT
2505      100000      CERR=   BIT15      ;CONTROLLER ERROR
2506      100000      CCLR=   BIT15      ;CONTROLLER CLEAR
2507
2508      ;           THESE BIT DEFINITIONS ARE USED FOR ADDRESS
2509      ;           THE HIGH BYTE OF RKCS1
2510
2511      000001      B.BA16=  BIT0       ;BUS ADDRESS BIT 16
2512      000002      B.BA17=  BIT1       ;BUS ADDRESS BIT 17
2513      000004      B.CDT=   BIT2       ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
2514      000020      B.CFMT=  BIT4       ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
2515
2516      .SBTTL      CONTROL AND STATUS REGISTER 2 BITS
2517
2518      000007      DRVMSK=  7           ;MASK FOR DRIVE SELECTION CODE
2519      000010      DESL=    BIT3       ;DESELECT OR RELEASE DRIVE IN BITS 0-2
2520      000010      RLS=    BIT3       ;DESELECT OR RELEASE DRIVE IN BITS 0-2
2521      000020      BAI=     BIT4       ;BUS ADDRESS INCREMENT INHIBIT
2522      000040      CLR=     BIT5       ;CLEAR CONTROLLER AND ALL DRIVES
2523      000040      SCLR=    BIT5       ;CLEAR CONTROLLER AND ALL DRIVES
2524      000100      IR=      BIT6       ;INPUT READY
2525      000200      OR=      BIT7       ;OUTPUT READY
2526      000400      UFE=     BIT8       ;UNIT FIELD ERROR
2527      001000      MDS=     BIT9       ;MULTIPLE DRIVE SELECT
2528      002000      PGE=     BIT10      ;PROGRAMMING ERROR
2529      004000      NEM=     BIT11      ;NON-EXISTENT MEMORY
2530      010000      NED=     BIT12      ;NON-EXISTENT DRIVE
2531      020000      UPE=     BIT13      ;UNIBUS PARITY ERROR
2532      040000      WCE=     BIT14      ;WRITE CHECK ERROR
2533      100000      DLT=     BIT15      ;DATA LATE ERROR
2534
2535      .SBTTL      ERROR REGISTER BIT DEFINITION
2536
2537      000001      ILC=      BIT0       ;ILLEGAL FUNCTION CODE
2538      000002      *ILF=    BIT0       ;ILLEGAL FUNCTION CODE
2539      000004      SKI=     BIT1       ;SEEK INCOMPLETE
2540      000004      ILF=     BIT2       ;ILLEGAL DRIVE FUNCTION
2541      000004      NXF=     BIT2       ;ILLEGAL DRIVE FUNCTION
2542      000010      DRPAR=   BIT3       ;DRIVE DETECTED DRIVE BUS PARITY ERROR
2543      000020      FMTE=    BIT4       ;FORMAT ERROR
2544      000040      DTYE=    BIT5       ;DRIVE TYPE ERROR
2545      000100      ECH=     BIT6       ;ECC HARD
2546      000200      BSE=     BIT7       ;BAD SECTOR ERROR
2547      000400      HCRC=    BIT8       ;HEADER CRC ERROR
2548      000400      HVRC=    BIT8       ;HEADER VRC ERROR
2549      001000      COE=     BIT9       ;CYLINDER ADDRESS OVERFLOW ERROR
2550      002000      IDAE=    BIT10      ;INVALID DISK ADDRESS ERROR
2551      004000      WLE=     BIT11      ;WRITE LOCK ERROR
2552      010000      DTE=     BIT12      ;DRIVE TIMING ERROR
2553      020000      OPT=     BIT13      ;OPERATION (SEARCH) INCOMPLETE
2554      040000      UNS=     BIT14      ;DRIVE UNSAFE
2555      100000      DCK=     BIT15      ;DATA CHECK
2556
2557      .SBTTL      STATUS REGISTER BIT DEFINITION
2558

```

2559	000001	DRA=	BIT0	; DRIVE AVAILABLE (CONTROLLER IS SET IF
2560				; THIS BIT IS RESET)
2561	000004	OFST=	BIT2	; DRIVE OFFSET
2562	000010	ACLO=	BIT3	; AC LOW
2563	000020	SPDLSS=	BIT4	; SPEED LOSS
2564	000020	DCLO=	BIT4	; DC LOW
2565	000040	DROT=	BIT5	; DRIVE OFF TRACK
2566	000100	VV=	BIT6	; VOLUME VALID
2567	000200	DRY=	BIT7	; DRIVE READY
2568	000200	DRDY=	BIT7	; DRIVE READY
2569	000400	DDT=	BIT8	; DRIVE TYPE (0=RK06,1=RK07)
2570	004000	WRL=	BIT11	; WRITE LOCK
2571	020000	PIP=	BIT13	; POSITIONING IN PROGRESS
2572	040000	DSC=	BIT14	; DRIVE STATUS CHANGE
2573	100000	SVAL=	BIT15	; STATUS VALID
2574				
2575		.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION		
2576				
2577	000017	MESMSK=	17	; MESSAGE MASK
2578				
2579	000020	PAT=	BIT4	; FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
2580	000040	DMD=	BIT5	; DIAGNOSTIC MODE
2581	000100	MSP=	BIT6	; MAINTENANCE SECTOR PULSE
2582	000200	MIND=	BIT7	; MAINTENANCE INDEX
2583	000400	MCLK=	BIT8	; MAINTENANCE CLOCK
2584	001000	MERD=	BIT9	; MAINTENANCE ENCODED READ DATA
2585	002000	MEWD=	BIT10	; MAINTENANCE ENCODED WRITE DATA
2586	004000	PCA=	BIT11	; PRECOMPENSATION ADVANCE
2587	010000	PCD=	BIT12	; PRECOMPENSATION DELAY
2588	020000	ECCW=	BIT13	; ECC WORD IS BEING READ OR WRITTEN
2589	040000	WRTGAT=	BIT14	; WRITE GATE
2590	100000	RDGATE=	BIT15	; READ GATE
2591				
2592		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A		
2593				
2594	000040	S.DRA=	BIT5	; DRIVE AVAILIABLE
2595	000100	S.VV=	BIT6	; VOLUME VALID
2596	000200	S.DRY=	BIT7	; DRIVE READY
2597	000400	S.TYPE=	BIT8	; DRIVE TYPE
2598	001000	S.FORM=	BIT9	; DRIVE FORMAT
2599	002000	S.OFF=	BIT10	; OFFSET
2600	004000	S.WRL=	BIT11	; WRITE LOCK
2601	010000	S.SPIN=	BIT12	; SPINDLE ON
2602	020000	S.PIP=	BIT13	; POSITIONING IN PROGRESS
2603	040000	S.DSC=	BIT14	; DRIVE STATUS CHANGE
2604				
2605		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B		
2606				
2607	000040	S.ICYL=	BIT5	; ILLEGAL CYLINDER ADDRESS
2608	000100	S.ACLO=	BIT6	; AC LOW
2609	000200	S.FLT=	BIT7	; DRIVE FAULT
2610	000400	S.ILF=	BIT8	; ILLEGAL FUNCTION
2611	001000	S.PAR=	BIT9	; DRIVE DETECTED DRIVE BUS PARITY ERROR
2612	002000	S.SKI=	BIT10	; SEEK INCOMPLETE
2613	004000	S.WLE=	BIT11	; WRITE LOCK ERROR
2614	010000	S.SPLS=	BIT12	; SPEED LOSS

F05

CZR6Q80 PK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 58
DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

SEQ 0057

2615	010000	S.DCLO= BIT12	;DC LOW
2616	020000	S.DROT= BIT13	;DRIVE OFF TRACK
2617	040000	S.UNS= BIT14	;DRIVE UNSAFE
2618			
2619		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
2620			
2621	000020	S.XDOK= BIT4	;TRANSDUCER OK
2622	000040	S.HDHM= BIT5	;HEADS HOME
2623	000100	S.BRHM= BIT6	;BRUSHES HOME
2624	000200	S.DOOR= BIT7	;DOOR INTERLOCKED
2625	000400	S.CART= BIT8	;CARTRAGE INTERLOCK
2626	001000	S.SPOK= BIT9	;SPEED OK
2627	002000	S.FWD= BIT10	;FORWARD
2628	004000	S.REV= BIT11	;REVERSE
2629	010000	S.LOAD= BIT12	;HEADS LOADING
2630	020000	S.RTZ= BIT13	;RETURN TO ZERO
2631	040000	S.UNLD= BIT14	;HEADS UNLOADING
2632			
2633		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
2634			
2635	000020	S.SECT= BIT4	;SECTOR ERROR
2636	000040	S.WCLK= BIT5	;WRITE CLOCK AND NO WRITE GATE
2637	000100	S.WGAT= BIT6	;WRITE GATE AND NO TRANSISTIONS
2638	000200	S.HDFL= BIT7	;HEAD FAULT
2639	000400	S.MHD= BIT8	;MULTIPLE HEAD SELECT
2640	001000	S.XERR= BIT9	;INDEX ERROR
2641	002000	S.DIB= BIT10	;DIBIT ERROR
2642	004000	S.PLO= BIT11	;PLO ERROR
2643	010000	S.NMOV= BIT12	;SEEK AND NO MOTION
2644	020000	S.LIMD= BIT13	;LIMIT DETECT ON SEEK
2645	040000	S.BRKE= BIT14	;SERVO-BRAKE
2646			
2647		.SBTTL	COMMON MASKS
2648			
2649	000007	M.DRV= 7	;DRIVE CODE
2650	100000	M.PAR= BIT15	;PARITY
2651	000003	M.ID= 3	;BYTE ID
2652	017760	M.COIF= 17760	;CYLINDER DIFFERENCE/OFFSET
2653	017760	M.CADD= 17760	;CYLINDER ADDRESS
2654	077770	M.SER= 77770	;DRIVE SERIAL NUMBER
2655	000760	M.SECT= 760	;SECTOR COUNT
2656	007000	M.HEAD= 7000	;HEAD DECODE

.SBTTL PARAMETER 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
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
44
46
48
50
52
54
56
58
60
62

.SBTTL PARAMETERS PASSED TO THE DRIVER

```

; THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
; TO THE RK06/RK07 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.CSIH= 7	; RKCSI 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	DRVPOD= BIT2	; DRIVE POSITIONED FOR DATA TRANSFER
000010	UEXATT= BIT3	; UNEXPECTED ATTENTION
000020	DRVHRD= BIT4	; DRIVE HAS HARD ERROR
000040	DRVDSK= BITS	; DRIVE STATUS CHANGE DID NOT CLEAR

CZR608.P11 02-DEC-77 10:46

PROGRAM DEVICE STATUS REGISTER DEFINITION

2713	000100	CMDTO=	BIT6	; NO TERMINATION TO COMMAND FOR AT
2714				; LEAST 1 SECOND
2715	000200	W.WCK=	BIT7	; WRITE FOR WRITE CHECK
2716	000400	NOCHK=	BIT8	; NO CHECK, DO NOT SET INTERRUPT ENABLE
2717	001000	PBSVAL=	BIT9	; PARAMETER STATUS WORDS VALID
2718				; (SET WHEN ERROR TERMINATION OR
2719				; READ STATUS COMMAND)
2720	002000	DRPDIV=	BIT10	; DROP DRIVE FROM TEST SEQUENCE
2721	004000	NODSC=	BIT11	; ATTENTION SET BUT DCS AND FAULT RESET
2722	010000	DRVSZD=	BIT12	; DRIVE SEIZED BY OTHER PORT
2723	020000	E.UNLD=	BIT13	; DRIVE UNLOADED DUE TO ERROR
2724	040000	Q.INIT=	BIT14	; PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
2725	100000	DTBAII=	BIT15	; INHIBIT BUS ADDRESS INCREMENT
2726				
2727		.SBTTL	PARAMETERS PASSED FROM DRIVER TO PROGRAM	
2728				
2729		:	THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS	
2730		:	FROM THE DRIVER TO THE CALLING PROGRAM	
2731				
2732	000016	P.CS1=	16	; COMMAND AND STATUS REGISTER 1
2733	000020	P.CS2=	20	; COMMAND AND STATUS REGISTER 2
2734	000022	P.WCR=	22	; WORD COUNT REGISTER
2735	000024	P.BAR=	24	; BUS ADDRESS REGISTER
2736	000026	P.DTS=	26	; DESIRED TRACK SECTOR REGISTER
2737	000030	P.DCYL=	30	; DESIRED CYLINDER REGISTER
2738	000032	P.ASOF=	32	; ATTENTION SUMMARY/OFFSET REGISTER
2739	000034	P.ER=	34	; ERROR REGISTER
2740	000036	P.DS=	36	; STATUS REGISTER
2741	000040	P.A00=	40	; MESSAGE A STATUS BYTE 00
2742	000042	P.B00=	42	; MESSAGE B STATUS BYTE 00
2743	000044	P.A01=	44	; MESSAGE A STATUS BYTE 01
2744	000046	P.B01=	46	; MESSAGE B STATUS BYTE 01
2745	000050	P.A10=	50	; MESSAGE A STATUS BYTE 10
2746	000052	P.B10=	52	; MESSAGE B STATUS BYTE 10
2747	000054	P.A11=	54	; MESSAGE A STATUS BYTE 11
2748	000056	P.B11=	56	; MESSAGE B STATUS BYTE 11
2749	000060	P.EPOS=	60	; ECC POSITION INFORMATION
2750	000062	P.EPAT=	62	; ECC PATTERN INFORMATION
2751				
2752		.SBTTL	PARAMETER BLOCK 0 FOR DRIVE	
2753				
2754	002640	000	PARMO: .BYTE 0	; DRIVE NUMBER
2755	002641	000	.BYTE 0	; COMMAND
2756	002642	000000	.WORD 0	; CYLINDER ADDRESS
2757	002644	000	.BYTE 0	; SECTOR ADDRESS
2758	002645	000	.BYTE 0	; TRACK ADDRESS
2759	002646	000	.BYTE 0	; OFFSET VALUE
2760	002647	000	.BYTE 0	; BUS ADDRESS (BITS 16 AND 17)
2761	002650	0000000	.WORD 0	; BUS ADDRESS (BITS 0 - 15)
2762	002652	0000000	.WORD 0	; WORD COUNT (2'S COMPLEMENT)
2763	002654	0000000	.WORD 0	; PROGRAM DRIVE STATUS INFORMATION
2764	002656	0000000	.WORD 0	; COMMAND AND STATUS REGISTER 1
2765	002660	0000000	.WORD 0	; COMMAND AND STATUS REGISTER 2
2766	002662	0000000	.WORD 0	; WORD COUNT REGISTER
2767	002664	0000000	.WORD 0	; BUS ADDRESS REGISTER
2768	002666	0000000	.WORD 0	; DESIRED TRACK AND SECTOR REGISTER

2769	002670	000000	.WORD	0	; DESIRED CYLINDER REGISTER
2770	002672	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
2771	002674	000000	.WORD	0	; ERROR REGISTER
2772	002676	000000	.WORD	0	; STATUS REGISTER
2773	002700	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
2774	002702	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
2775	002704	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
2776	002706	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
2777	002710	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
2778	002712	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
2779	002714	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
2780	002716	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
2781	002720	000000	.WORD	0	; ECC POSITION INFORMATION
2782	002722	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL PARAMETER BLOCK 1 FOR DRIVE

2786	002724	000	PARM1: .BYTE	0	; DRIVE NUMBER
2787	002725	000	.BYTE	0	; COMMAND
2788	002726	000000	.WORD	0	; CYLINDER ADDRESS
2789	002730	000	.BYTE	0	; SECTOR ADDRESS
2790	002731	000	.BYTE	0	; TRACK ADDRESS
2791	002732	000	.BYTE	0	; OFFSET VALUE
2792	002733	000	.BYTE	0	; BUS ADDRESS (BITS 16 AND 17)
2793	002734	000000	.WORD	0	; BUS ADDRESS (BITS 0 - 15)
2794	002736	000000	.WORD	0	; WORD COUNT (2'S COMPLEMENT)
2795	002740	000000	.WORD	0	; PROGRAM DRIVE STATUS INFORMATION
2796	002742	000000	.WORD	0	; COMMAND AND STATUS REGISTER 1
2797	002744	000000	.WORD	0	; COMMAND AND STATUS REGISTER 2
2798	002746	000000	.WORD	0	; WORD COUNT REGISTER
2799	002750	000000	.WORD	0	; BUS ADDRESS REGISTER
2800	002752	000000	.WORD	0	; DESIRED TRACK AND SECTOR REGISTER
2801	002754	000000	.WORD	0	; DESIRED CYLINDER REGISTER
2802	002756	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
2803	002760	000000	.WORD	0	; ERROR REGISTER
2804	002762	000000	.WORD	0	; STATUS REGISTER
2805	002764	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
2806	002766	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
2807	002770	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
2808	002772	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
2809	002774	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
2810	002776	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
2811	003000	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
2812	003002	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
2813	003004	000000	.WORD	0	; ECC POSITION INFORMATION
2814	003006	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL TEMPORARY CONTROLLER REGISTER STORAGE

2818	003010	000000	T.CS1: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 1
2819					
2820	003012	000000	T.CS2: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 2
2821					
2822	003014	000000	T.WCR: .WORD	0	; TEMPORARY STORAGE FOR WORD COUNT REGISTER
2823	003016	000000	T.BA: .WORD	0	; TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
2824	003020	000000	T.DA: .WORD	0	; TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

2825	003022	000000	T.DC: .WORD	0	; TEMPORARY STORAGE FOR DRIVE CYLINDER
2826	003024	000000	T.ASOF: .WORD	0	; TEMPORARY STORAGE FOR ATTENTION SUMMARY
2827					AND OFFSET
2828	003026	000000	T.ER: .WORD	0	; TEMPORARY STORAGE FOR ERROR REGISTER
2829	003030	000000	T.DS: .WORD	0	; TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
2830	003032	000000	T.MR1: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
2831	003034	000000	T.MR2: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
2832	003036	000000	T.MR3: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
2833	003040	000000	T.POS: .WORD	0	; TEMPORARY STORAGE FOR ECC POSITION
2834	003042	000000	T.PAT: .WORD	0	; TEMPORARY STORAGE FOR ECC PATTERN
2835	003044	000000	T.DB: .WORD	0	; TEMPORARY STORAGE FOR DATA BUFFER REGISTER
2836					
2837			.SBTTL	DRIVER PARAMETERS	
2838					
2839	003046	177440	RKBAS: .WORD	177440	; ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
2840	003050	000210	RKVEC: .WORD	210	; ADDRESS OF R611 VECTOR
2841	003052	000240	RKPRI: .WORD	PR5	; RK611 INTERRUPT PRIORITY
2842	003054	030272	A.NORM: ERRFRE		; ADDRESS OF NORMAL RETURN FROM DRIVER
2843	003056	031504	A.ABNL: ERRHDL		; ADDRESS OF ABNORMAL RETURN FROM DRIVER
2844	003060	031000	A.CONT: CONERR		; ADDRESS OF CONTROLLER ERROR RETURN
2845	003062	000000	E.CONT: .WORD	0	; CONTROLLER ERROR STATUS
2846					THIS LOCATION IS CLEARED WHEN EVERY COMMAND
2847					IS INITIATED. IF A CONTROLLER ERROR
2848					OCCURS THE FOLLOWING BIT ASSIGNMENT IS
2849					USED:
2850					
2851		000001	E.CCLR=	BIT0	; CLEAR CONTROLLER DID NOT CLEAR ERROR
2852		000002	E.NOAT=	BIT1	; NO ATTENTION IN ATTENTION SUMMARY REG
2853		000004	E.UATT=	BIT2	; UNSOLICITED ATTENTION (SEQUENTIAL ONLY)
2854		000010	E.UDAT=	BIT3	; UNEXPECTED DATA TYPE ERROR
2855		000020	E.CLAT=	BIT4	; ATTENTION DID NOT RESET WITH CLEAR
2856		000040	E.SCLR=	BIT5	; SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
2857					ATTENTION
2858		000100	E.IILD=	BIT6	; ILLEGAL DRIVER COMMAND
2859		000400	E.DLT=	BIT8	; DATA LATE WHEN UNLOADING HEADER
2860		001000	E.CERR=	BIT9	; CONTROLLER ERROR DURING DRIVER SERVICING
2861		002000	E.DPAR=	BIT10	; DRIVE DETECTED PARITY ERROR
2862		040000	E.CMT0=	BIT14	; CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
2863		100000	E.MDS=	BIT15	; MULTIPLE DRIVE SELECT
2864					
2865	003064	000000	O.WAIT: .WORD	0	; PARAMETER BLOCK OF THE DRIVE
2866					WAITING FOR COMMAND COMPLETION
2867	003066	000400	W.MTIM: .WORD	400	; LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
2868	003070	000400	W.MILI: .WORD	400	; 16 MILLISECOND TIME FOR PROGRAM
2869					
2870					
2871					
2872					
2873					
2874					
2875					
2876					
2877					
2878					
2879					
2880					

CPU	VALUE
---	----
11/05	10
11/10	
11/20	
11/34	
11/40	
11/45	400
11/50	
11/70	

K05

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 63
 CZR6Q8.P11 02-DEC-77 10:46 DRIVER PARAMETERS

SEQ 0062

2881	003072	000300			W.SEC: .WORD	300	; SECOND COUNT COUNT FOR ALL COMMANDS
2882							; EXCEPT START SPINDLE
2883	003074	003000			W.BSEC: .WORD	3000	; 8 SECOND FOR DRIVE CYCLE DOWN
2884	003076	030000			W.MIN: .WORD	30000	; MINUTE TIME FOR START SPINDLE
2885	003100	000000			HDR.AD: .WORD	0	; ADDRESS USED FOR READ ALL HEADERS
2886	003102	000000			HDR.CT: .WORD	0	; NUMBER OF HEADERS LEFT TO READ FOR READ
2887							; ALL HEADERS
2888	003104	000			I.ISRL: .BYTE	0	; INTERRUPT OR RELEASED COMMAND ISSUED
2889	003105	002	004	010	H.HEAD: .BYTE	2,4,10	; HEAD DECODES
2890	003110	000			W.TIME: .BYTE	0	; DRIVES BEING WATCH-DOG TIMED
2891							
2892					.SBTTL	INTERRUPT MASKS	
2893							
2894	003111	000			INTMSK: .BYTE	0	; INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
2895							
2896						INTERRUPT MASK TABLE	
2897							
2898	003112	001			I.DRV: .BYTE	1	; INTERRUPT MASK FOR DRIVE 0
2899	003113	002			.BYTE	2	; INTERRUPT MASK FOR DRIVE 1
2900	003114	004			.BYTE	4	; INTERRUPT MASK FOR DRIVE 2
2901	003115	010			.BYTE	10	; INTERRUPT MASK FOR DRIVE 3
2902	003116	020			.BYTE	20	; INTERRUPT MASK FOR DRIVE 4
2903	003117	040			.BYTE	40	; INTERRUPT MASK FOR DRIVE 5
2904	003120	100			.BYTE	100	; INTERRUPT MASK FOR DRIVE 6
2905	003121	200			.BYTE	200	; INTERRUPT MASK FOR DRIVE 7
2906							
2907					.SBTTL	PARAMETER BLOCK TABLE	
2908							
2909	003122	002640			PBLKT: PARM0		; ADDRESS OF PARAMETER BLOCK GIVEN WITH
2910							; DRIVE CALL. MUST BE LOADED INTO PBLKT
2911							
2912							
2913					.SBTTL	TIME FOR WATCH-DOG TIMER	
2914							
2915	003124	000000			W.DRV: .WORD	0	; TIME FOR INSTRUCTION IN PARAMETER BLOCK
2916					.SBTTL	PROGRAM SPECIFIC RESERVED LOCATIONS	
2917	003126	000			MDFLAG: .BYTE	0	; FLAG TO INDIC. DEFLT OR PARAM MODE
2918	003127	000			TSTING: .BYTE	0	; CURRENTLY RUNNING TESTS IF = 1
2919	003130	000			DERCNT: .BYTE	0	; DATA ERROR COUNT
2920	003131	000			OPCOMP: .BYTE	0	; OPERATION COMPLETE FLAG
2921	003132	000			DONE: .BYTE	0	; DONE SWITCH
2922	003133	000			TYPFMT: .BYTE	0	; DRIVE TYPE & FORMAT CONTROL
2923	003134	000			FORMAT: .BYTE	0	; DRIVE FORMAT IN BIT 4 OF BYTE
2924	003135	000			ERRCNT: .BYTE	0	; ERROR COUNT
2925	003136	004			ERRLMT: .BYTE	4	; ERROR LIMIT
2926	003137	000			ORVERS: .BYTE	0	; ERROR COUNT FOR CURRENT DRIVE
2927	003140	000			OPCONT: .BYTE	0	; OPERATION CONTROL SWITCHES
2928	003141	000			DRNAFG: .BYTE	0	; =1 INDICATES DRIVE SIEZED BY OTHER PORT
2929	003142	000			REISSU: .BYTE	0	; DUAL-ACC FLAG TO RE-ISSUE COMMAND
2930	003143	000			TSTTYP: .BYTE	0	; OFFSET TEST IDENTIFIER
2931	003144	000			DCEFLG: .BYTE	0	; DATA CHECK ERROR FLAG
2932	003145	000			WCEFLG: .BYTE	0	; WRITE CHECK ERROR FLAG
2933	003146	000			DLTFLG: .BYTE	0	; DATA LATE ERROR FLAG
2934	003147	000			DIF100: .BYTE	0	; "CYL DIF = 100/200" FLAG
2935	003150	000			NORTRY: .BYTE	0	; "NO-RETRY" FLAG
2936	003151	000			MEMABT: .BYTE	0	; SET BYTE = 1 FOR NO ABORT

Line	Address	Value	Field	Unit	Description
2937					ON MEMORY PARITY ERRORS
2938	003152	000	HLPOVL: .BYTE	0	HELP FILE OVERLAID INDICATOR
2939	003153	000	NOTYPE: .BYTE	0	INDICATES PROGRAM JUST LOADED IF 0
2940	003154	000	LOGDRV: .BYTE	0	LOGICAL DRIVE NUMBER (00-17)
2941	003155	000	PASSNO: .BYTE	0	PASS NUMBER (= 1 OR 2)
2942	003156	000	OFSDIR: .BYTE	0	OFFSET DIRECTION FLAG (0=NEG)
2943	003157	000	OFSCNT: .BYTE	0	OFFSET COUNT FOR ONE DIRECTION
2944		000001	WHDSW=BIT0		WRITE HEADER & DATA SWITCH
2945		000002	VFHDSW=BIT1		VERIFY HEADERS SWITCH
2946		000004	WCDASW=BIT2		WRITE CHECK DATA SWITCH
2947		000010	RCDASW=BIT3		READ CHECK DATA SWITCH
2948		000020	OREQSW=BIT4		OFFSET REQUIRED SWITCH
2949			. EVEN		
2950					
2951		000114	MEMVEC=114		MEMORY PARITY TRAP VECTOR
2952	003160	000000	SAVWRD: .WORD	0	SCRATCH WORD
2953	003162	000400	BSSOFT: .BLKW	↑D256	RECORD OF BAD SECTORS FROM SOFTWARE
2954	004162	000400	BSFACT: .BLKW	↑D256	RECORD OF BAD SECTORS FROM FACTORY
2955	005162	000000	PRVCMO: .WORD	0,0,0,0,0,0	PREVIOUS COMMAND STORAGE
2956	005170	000000			
2957	005176	000000	COMSTR: .WORD	0,0,0,0,0,0	CURRENT COMMAND STORAGE
2958	005204	000000			
2959	005212	000102	BUFFO: .BLKW	↑D66	OUTPUT BUFFER 1
2960	005416	000000	LOWOCT: .WORD	0	LOW 16 BITS OF CONVERTED BINARY NUMBER
2961	005420	000000	HIGOCT: .WORD	0	HIGH BITS OF CONVERTED BINARY NO.
2962	005422	000000	BUFPRT: .WORD	0	BUFFER POINTER
2963	005424	000000	RECODE: .WORD	0	RECOVERY CODE WORD
2964	005426	000000	ERRCOM: .WORD	0	ERROR COMMAND
2965	005430	000000	DRIVE: .WORD	0	NO. OF DRIVE IN USE
2966	005432	000000	STALLS: .WORD	0	CURRENT NO. OF UNIT STALLS TO APPLY
2967	005434	000000	CYLNDR: .WORD	0	CURRENT CYLINDER NUMBER
2968	005436	000000	OFINUS: .WORD	0	OFFSET IN USE
2969	005440	000000	TRACK: .WORD	0	TRACK IN USE
2970	005442	000000	INTCHR: .WORD	0	TTY INTERRUPT INPUT WORD
2971	005444	000000	SCRACH: .WORD	0	ALL-PURPOSE SCRATCH WORD
2972	005446	000000	PATRN: .WORD	0	DATA PATTERN LIST WORD
2973	005450	000000	PLOFST: .WORD	0	(+) OFFSET VALUE
2974	005452	000000	NGOFST: .WORD	0	(-) OFFSET VALUE
2975	005454	000005	SAVPRS: .BLKW	5	SAVE INITIAL PARAMS FOR XFER
2976	005466	000000	WDSXFR: .WORD	0	NO. OF WORDS ACTUALLY XFERRED
2977	005470	000000	FINCYL: .WORD	0	FINAL CYLINDER ADRS
2978	005472	000	FINTRK: .BYTE	0	FINAL TRACK ADRS
2979	005473	000	FINSEC: .BYTE	0	FINAL SECTOR ADRS
2980	005474	000000	LASTWC: .WORD	0	ACTUAL FINAL WC
2981	005476	000000	NEWON: .WORD	0	NEW LOOK AT ONLINE DRIVES
2982	005500	000002	MA: .BLKW	2	
2983	005504	000002	PMA: .BLKW	2	
2984	005510	000000	SUBSYS: .WORD	0	MEM. BUFFER ADDRESS
2985	005512	000021	DRVLST: .BLKW	17.	CURRENT SUBSYSTEM NAME STORAGE (ASCII)
2986					EACH WORD CONTAINS THE NAME OF A DRIVE
2987					TO BE TESTED. HIGH BYTE IS SUBSYSTEM
2988					NAME, LOW BYTE IS DRIVE NO. (ASCII).
2989	005554	000000	DRVPTN: .WORD	0	IF WORD = 0, DRIVE IS NOT TESTED.
2990	005556	000010	SCRLLST: .BLKW	8.	DRIVE LIST POINTER
2991					SCRATCH DRIVE LIST
2992	005566		PATOO:		

2993
2994
2995
2996
2997
2998 005566
2999 005566 000620 000000
3000 005572 000620 000002
3001 005576 000620 000004
3002 005602 000620 000006
3003 005606 000620 000010
3004 005612 000620 000012
3005 005616 000620 000014
3006 005622 000620 000016
3007 005626 000622 000000
3008 005632 000622 000002
3009 005636 000622 000004
3010 005642 000622 000006
3011 005646 000622 000010
3012 005652 000622 000012
3013 005656 000622 000014
3014 005662 000622 000016
3015
3016
3017
3018 005666 001444 000000
3019 005672 001444 000002
3020 005676 001444 000004
3021 005702 001444 000006
3022 005706 001444 000010
3023 005712 001444 000012
3024 005716 001444 000014
3025 005722 001444 000016
3026 005726 001446 000000
3027 005732 001446 000002
3028 005736 001446 000004
3029 005742 001446 000006
3030 005746 001446 000010
3031 005752 001446 000012
3032 005756 001446 000014
3033 005762 001446 000016
3034
3035
3036
3037
3038 005766
3039 005766 072307
3040 005770 135143
3041 005772 156461
3042 005774 167230
3043 005776 073514
3044 006000 035646
3045 006002 016723
3046 006004 107351
3047 006006 143564
3048 006010 061672

;TABLE A - BASIC READ/WRITE TEST SECTORS FOR RK06

TABLE A:	WORD	620,0	DRIVE	0
	.WORD	620,2		1
	.WORD	620,4		2
	.WORD	620,6		3
	.WORD	620,10		4
	.WORD	620,12		5
	.WORD	620,14		6
	.WORD	620,16		7
	.WORD	622,0		10
	.WORD	622,2		11
	.WORD	622,4		12
	.WORD	622,6		13
	.WORD	622,10		14
	.WORD	622,12		15
	.WORD	622,14		16
	.WORD	622,16		17

;TABLE A - CORRESPONDING CYL NUMBERS FOR THE RK07

TABL7A:	WORD	1444,0	DRV	0
	.WORD	1444,2		1
	.WORD	1444,4		2
	.WORD	1444,6		3
	.WORD	1444,10		4
	.WORD	1444,12		5
	.WORD	1444,14		6
	.WORD	1444,16		7
	.WORD	1446,0		10
	.WORD	1446,2		11
	.WORD	1446,4		12
	.WORD	1446,6		13
	.WORD	1446,10		14
	.WORD	1446,12		15
	.WORD	1446,14		16
	.WORD	1446,16		17

;TABLE B - WORST CASE DATA PATTERN

TABLE B:	072307
	135143
	156461
	167230
	073514
	035646
	016723
	107351
	143564
	061672

3049	006012	030735	030735
3050	006014	114356	114356
3051	006016	046167	046167
3052	006020	123073	123073
3053	006022	151453	151453
3054	006024	164616	164616

;TABLE OF STARTING CYLINDERS FOR ALL OVWRT AND COMPAT CYL BLKS FOR RK06

TABLEC:	.WORD	0,60	FOR CURRENT ZONE	1
	.WORD	100,160		2
	.WORD	200,260		3
	.WORD	300,360		4
	.WORD	400,460		5
	.WORD	500,560		6
	.WORD	600		7

;TABLE C CORRESPONDING CYL NUMBERS FOR RK07

TABL7C:	.WORD	0,160	CURRENT ZONE	1
	.WORD	200,360		2
	.WORD	400,560		3
	.WORD	600,760		4
	.WORD	1000,1160		5
	.WORD	1200,1360		6
	.WORD	1400		7

;CYLINDER BLOCK LAYOUT TABLE

TABLED:	.BLKB	16.	CYLINDER 0 IN BLK	1
	.BLKB	16.		2
	.BLKB	16.		3
	.BLKB	16.		4
	.BLKB	16.		5
	.BLKB	16.		6
	.BLKB	16.		7
	.BLKB	16.		10
	.BLKB	16.		11
	.BLKB	16.		12
	.BLKB	16.		13
	.BLKB	16.		14
	.BLKB	16.		15
	.BLKB	16.		16
	.BLKB	16.		17

;TABLE G - PSEUDO-RANDOM DATA PATTERN

TABLEG:	040135
	177070
	070414
	064531

3055			
3056			
3057			
3058			
3059	006026	000000	000060
3060	006032	000100	000160
3061	006036	000200	000260
3062	006042	000300	000360
3063	006046	000400	000460
3064	006052	000500	000560
3065	006056	000600	
3066			
3067			
3068			
3069	006060	000000	000160
3070	006064	000200	000360
3071	006070	000400	000560
3072	006074	000600	000760
3073	006100	001000	001160
3074	006104	001200	001360
3075	006110	001400	
3076			
3077			
3078			
3079			
3080	006112	000020	
3081	006132	000020	
3082	006152	000020	
3083	006172	000020	
3084	006212	000020	
3085	006232	000020	
3086	006252	000020	
3087	006272	000020	
3088	006312	000020	
3089	006332	000020	
3090	006352	000020	
3091	006372	000020	
3092	006412	000020	
3093	006432	000020	
3094	006452	000020	
3095	006472	000020	
3096			
3097			
3098			
3099			
3100	006512		
3101	006512	040135	
3102	006514	177070	
3103	006516	070414	
3104	006520	064531	

3105	006522	174473	174473
3106	006524	062422	062422
3107	006526	114352	114352
3108	006530	036620	036620
3109	006532	010031	010031
3110	006534	052336	052336
3111	006536	017310	017310
3112	006540	011347	011347
3113	006542	102367	102367
3114	006544	152567	152567
3115	006546	001246	001246
3116	006550	160073	160073

3117
3118
3119

3120

3121	006552	000	001	003
3122	006555	006	012	017
3123	006560	005	014	004
3124	006563	015	007	002
3125	006566	016	013	011
3126	006571	010		

3127

3128

3129

3130				
3131	006572	000	001	002
3132	006575	003	005	006
3133	006600	007	010	012
3134	006603	013	014	015
3135	006606	017	020	021
3136	006611	022		

3137

3138

3139

3140

3141

3142

3143

3144

3145

3146

3147

3148

3149

3150

3151

3152

3153

3154

3155

3156

3157

3158

3159

3160

;LIST OF STARTING DRIVE NUMBERS, TO GENERATE TABLED
 STDLIST: .BYTE 0,1,3,6,12,17,5,14,4,15,7,2,16,13,11,10

;LIST OF USEABLE SECTORS IN EACH CYLINDER BLOCK
 SECLST: .BYTE 0,1,2,3,5,6,7,10,12,13,14,15,17,20,21,22

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR NEG OFST
 NOSCR0: .BLKW 16.

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR NEG OFST
 NOSCR1: .BLKW 16.

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR NEG OFST
 NOSCR2: .BLKW 16.

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR POS OFST
 POSCR0: .BLKW 16.

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR POS OFST
 POSCR1: .BLKW 16.

;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR POS OFST
 POSCR2: .BLKW 16.

3161			
3162			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR NEG OFST
3163	007112	000020	NCSCRO: .BLKW 16.
3164			
3165			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR NEG OFST
3166	007152	000020	NCSCR1: .BLKW 16.
3167			
3168			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR NEG OFST
3169	007212	000020	NCSCR2: .BLKW 16.
3170			
3171			
3172			
3173			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR POS OFST
3174	007252	000020	PCSCRO: .BLKW 16.
3175			
3176			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR POS OFST
3177	007312	000020	PCSCR1: .BLKW 16.
3178			
3179			:TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR POS OFST
3180	007352	000020	PCSCR2: .BLKW 16.
3181			
3182			
3183			
3184			
3185			:SELF-TEST SCORE FOR TRACK 0 FOR NEG OFST
3186	007412	000000	NSSCRO: .WORD 0
3187			
3188			:SELF-TEST SCORE FOR TRACK 1 FOR NEG OFST
3189	007414	000000	NSSCR1: .WORD 0
3190			
3191			:SELF-TEST SCORE FOR TRACK 2 FOR NEG OFST
3192	007416	000000	NSSCR2: .WORD 0
3193			
3194			
3195			
3196			:SELF-TEST SCORE FOR TRACK 0 FOR POS OFST
3197	007420	000000	PSSCRO: .WORD 0
3198			
3199			:SELF-TEST SCORE FOR TRACK 1 FOR POS OFST
3200	007422	000000	PSSCR1: .WORD 0
3201			
3202			:SELF-TEST SCORE FOR TRACK 2 FOR POS OFST
3203	007424	000000	PSSCR2: .WORD 0
3204			
3205			
3206			
3207			:AVERAGE OF OVRWRT SCORES OVER ALL SURFACES
3208	007426	000000	OVRAVG: .WORD 0
3209			
3210			:AVERAGE OF READ SCORES OVER ALL SURFACES
3211	007430	000000	COMAVG: .WORD 0
3212			
3213			
3214			
3215			:SCRATCH TABLE OF SCORES FOR TRACK 0 FOR NEG OFST
3216	007432	000020	NSCORO: .BLKB 16.

```

3217
3218 ; SCRATCH TABLE OF SCORES FOR TRACK 1 FOR NEG OFST
3219 007452 000020 NSCOR1: .BLKB 16.
3220
3221 ; SCRATCH TABLE OF SCORES FOR TRACK 2 FOR NEG OFST
3222 007472 000020 NSCOR2: .BLKB 16.
3223
3224
3225
3226 ; SCRATCH TABLE OF SCORES FOR TRACK 0 FOR POS OFST
3227 007512 000020 PSCOR0: .BLKB 16.
3228
3229 ; SCRATCH TABLE OF SCORES FOR TRACK 1 FOR POS OFST
3230 007532 000020 PSCOR1: .BLKB 16.
3231
3232 ; SCRATCH TABLE OF SCORES FOR TRACK 2 FOR POS OFST
3233 007552 000020 PSCOR2: .BLKB 16.
3234
3235
3236
3237 000002 LSTTRK=2 ; LAST TRACK = 2
3238 000002 BSERR=BIT1 ; BSE ERROR
3239 000004 HVR CER=BIT2 ; HVRC ERROR
3240 000010 OPIERR=BIT3 ; OPI ERROR
3241 000020 DCKERR=BIT4 ; DATA CHECK ERROR
3242 000040 ECCNC=BIT5 ; ECC NON-CORRECTABLE
3243 000100 WCERR=BIT6 ; WRITE CHECK ERROR
3244 000200 ABORT=BIT7 ; ABORT
3245 000400 LEV2ER=BIT8 ; LEVEL TWO ERROR
3246 001000 BADSEC=BIT9 ; BAD SECTOR FLAG
3247 002000 TWOTOS=BIT10 ; TWO TIME OUTS
3248 004000 RCLREQ=BIT11 ; RECALIBRATE REQUIRED
3249 010000 DRNAV1=BIT12 ; DRIVE NOT RELOD BY OTHER PORT
3250 100000 ANYDER=BIT15 ; ANY ERROR DETECTED FLAG
3251
3252 007572 005015 041412 051132 DZR6Q: .ASCIIZ <15><12><12>/CZR6Q80 - RK06-RK07 DRIVE COMPATIBILITY PROGRAM/<15><12><12>
3253 007600 050466 030102 026440
3254 007606 051040 030113 026466
3255 007614 045522 033460 042040
3256 007622 044522 042526 041440
3257 007630 046517 040520 044524
3258 007636 044502 044514 054524
3259 007644 050040 047522 051107
3260 007652 046501 005015 000012
3261 007660 051012 030113 026466
3262 007666 033460 041040 051525
3263 007674 040440 051104 036440
3264 007702 000040
3265 007704 045522 033060 030055
3266 007712 020067 042526 020103
3267 007720 042101 020122 020075
3268 007726 000
3269 007727 122 030113 026466
3270 007734 033460 050040 044522
3271 007742 051117 052111 020131
3272 007750 000075

```

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 70
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0069

3273	007752	052012	J50131	020105	TYP SYS: .ASCIZ <12>/TYPE NAME OF NEXT SUBSYS TO TEST (A-P) : /
3274	007760	040516	042515	047440	
3275	007766	020106	042516	052130	
3276	007774	051440	041125	054523	
3277	010002	020123	047524	052040	
3278	010010	051505	020124	040450	
3279	010016	050055	020051	020072	
3280	010024	000			
3281	010025	012	051104	053111	ASKTYP: .ASCIZ <12>/DRIVE TYPE 6<CR> FOR RK06, 7<CR> FOR RK07: /
3282	010032	020105	054524	042520	
3283	010040	020040	036066	051103	
3284	010046	020076	047506	020122	
3285	010054	045522	033060	020054	
3286	010062	036067	051103	020076	
3287	010070	047506	020122	045522	
3288	010076	033460	020072	000	
3289	010103	123	041125	054523	SYS DRV: .ASCIZ /SUBSYS DRIVE(S) = /
3290	010110	020123	020040	051104	
3291	010116	053111	024105	024523	
3292	010124	036440	000040		
3293	010130	044527	046114	052040	WILTST: .ASCIZ /WILL TEST DRIVE(S) /
3294	010136	051505	020124	051104	
3295	010144	053111	024105	024523	
3296	010152	000040			
3297	010154	047440	020116	052523	ON SUBS: .ASCIZ / ON SUBSYS /<15><12><12>
3298	010162	051502	051531	020040	
3299	010170	005015	000012		
3300	010174	051511	052040	042510	ANOTHR: .ASCIZ /IS THERE ANOTHER SUBSYS (Y OR N <CR>) ? /
3301	010202	042522	040440	047516	
3302	010210	044124	051105	051440	
3303	010216	041125	054523	020123	
3304	010224	054450	047440	020122	
3305	010232	020116	041474	037122	
3306	010240	020051	020077	000	
3307	010245	012	047515	042522	DROVFL: .ASCIZ <12>/MORE THAN 16 DRIVES ENTERED !/<15><12><12>
3308	010252	052040	040510	020116	
3309	010260	033061	042040	044522	
3310	010266	042526	020123	047105	
3311	010274	042524	042522	020104	
3312	010302	006441	005012	000	
3313	010307	012	025052	051440	SRPASS: .ASCIZ <12>/** STARTING PASS **/<15><12>
3314	010314	040524	052122	047111	
3315	010322	020107	047520	051523	
3316	010330	020040	023040	006452	
3317	010336	000012			
3318	010340	046412	052517	052116	MNTPAK: .ASCIZ <12>/MOUNT PACK ON DRIVE AND LOAD./<15><12>
3319	010346	050040	041501	020113	
3320	010354	047117	042040	044522	
3321	010362	042526	020040	020040	
3322	010370	047101	020104	047514	
3323	010376	042101	006456	000012	
3324	010404	052417	046116	040517	DISPAK: .ASCIZ <12>/UNLOAD DRIVE AND REMOVE PACK./<15><12>
3325	010412	02010	051104	053111	
3326	010420	020105	020040	040440	
3327	010426	042116	051040	046505	
3328	010434	053117	020105	040520	

3329	010442	045503	006456	000012	
3330	010450	051412	040524	052122	STR204: .ASCIZ <12>/START AT ADR 204 FOR PASS 1 ON SUBSYS ./<15><12><12>
3331	010456	040440	020124	042101	
3332	010464	020122	030062	020064	
3333	010472	047506	020122	040520	
3334	010500	051523	030440	047440	
3335	010506	020116	052523	051502	
3336	010514	051531	020040	006456	
3337	010522	005012	000		
3338	010525	012	052123	051101	STR220: .ASCIZ <12>/START AT ADR 220 FOR PASS 2 ON SUBSYS ./<15><12><12>
3339	010532	020124	052101	040440	
3340	010540	051104	031040	030062	
3341	010546	043040	051117	050040	
3342	010554	051501	020123	020062	
3343	010562	047117	051440	041125	
3344	010570	054523	020123	027040	
3345	010576	005015	000012		
3346	010602	054524	042520	051040	TYPRWN: .ASCIZ /TYPE R<CR> WHEN /
3347	010610	041474	037122	053440	
3348	010616	042510	020116	000	
3349	010623	104	044522	042526	DRIRDY: .ASCIZ /DRIVE READY : /
3350	010630	051040	040505	054504	
3351	010636	035040	000040		
3352	010642	047504	042516	035040	DRIDON: .ASCIZ /DONE : /
3353	010650	000040			
3354	010652	047516	042040	044522	NODRLS: .ASCIZ /NO DRIVES LISTED FOR THIS SUBSYS !/<15><12>
3355	010660	042526	020123	044514	
3356	010666	052123	042105	043040	
3357	010674	051117	052040	044510	
3358	010702	020123	052523	051502	
3359	010710	051531	020440	005015	
3360	010716	000			
3361	010717	015	051412	047503	RESLTS: .ASCII <15><12>/SCORES FOR DRIVE (0-12 DEC.) :/<15><12><12>
3362	010724	042522	020123	047506	
3363	010732	020122	051104	053111	
3364	010740	020105	020040	020040	
3365	010746	030050	030455	020062	
3366	010754	042504	027103	020051	
3367	010762	006472	005012		
3368	010766	051124	041501	004513	.ASCII /TRACK DRIVE OVRWRT OVRWRT READ READ/<15><12>
3369	010774	051104	053111	004505	
3370	011002	053117	053522	052122	
3371	011010	047411	051126	051127	
3372	011016	004524	042522	042101	
3373	011024	051011	040505	006504	
3374	011032	012			
3375	011033	116	046525	051011	.ASCII /NUM READ SCORE SCORE SCORE SCORE/<15><12>
3376	011040	040505	004504	041523	
3377	011046	051117	004505	041523	
3378	011054	051117	004505	041523	
3379	011062	051117	004505	041523	
3380	011070	051117	006505	012	
3381	011075	011	047411	051506	.ASCIZ / OFST- OFST+ OFST- OFST+/<15><12><12>
3382	011102	026524	047411	051506	
3383	011110	025524	047411	051506	
3384	011116	026524	047411	051506	

3385	011124	025524	005015	000012	
3386	011132	047440	051126	051127	OAVERG: .ASCIZ / OVRWRT AVG = /
3387	011140	020124	053101	020107	
3388	011146	020075	000		
3389	011151	040	042522	042101	RAVERG: .ASCIZ / READ AVG = /
3390	011156	040440	043526	020040	
3391	011164	036440	000040		
3392	011170	005015	025012	020052	ENDTST: .ASCIZ <15><12><12>/** END OF TESTING **/<15><12>
3393	011176	042440	042116	047440	
3394	011204	020106	042524	052123	
3395	011212	047111	020107	025040	
3396	011220	006452	000012		
3397	011224	020040	000011		TRKBUF: .ASCIZ / /
3398	011230	020040	004440	000	DRVBUF: .ASCIZ / /
3399	011235	123	046105	004506	SELF: .ASCIZ /SELF /
3400	011242	000			
3401	011243	011	000		TAB: .ASCIZ / /
3402	011245	123	051127	036440	SWRMSG: .ASCIZ /SWR = /
3403	011252	000040			
3404	011254	020040	042516	020127	NEWMSG: .ASCIZ / NEW = /
3405	011262	020075	000		
3406	011265	015	042012	044522	DRVSEQ: .ASCIZ <15><12>/DRIVE(S) = /
3407	011272	042526	051450	020051	
3408	011300	020075	000		
3409	011303	104	044522	042526	BADDRV: .ASCIZ /DRIVE /
3410	011310	020040	000040		
3411	011314	047516	026516	054105	NXDRIV: .ASCIZ /NON-EXISTENT/<15><12>
3412	011322	051511	042524	052116	
3413	011330	005015	000		
3414	011333	116	052117	051040	NTREDY: .ASCIZ /NOT READY/<15><12>
3415	011340	040505	054504	005015	
3416	011346	000			
3417	011347	127	044522	042524	WRTLOK: .ASCIZ /WRITE-LOCKED/<15><12>
3418	011354	046055	041517	042513	
3419	011362	006504	000012		
3420	011366	047514	042101	042105	ALNPAK: .ASCIZ /LOADED WITH ALIGN PACK/<15><12>
3421	011374	053440	052111	020110	
3422	011402	046101	043511	020116	
3423	011410	040520	045503	005015	
3424	011416	000			
3425	011417	111	020106	054130	REPLPK: .ASCIZ /IF XXDP PACK ON DRV D, TYPE "Y <CR>", & REPLACE IT : /
3426	011424	050104	050040	041501	
3427	011432	020113	047117	042040	
3428	011440	053122	030040	020054	
3429	011446	054524	042520	021040	
3430	011454	020131	041474	037122	
3431	011462	026042	023040	020040	
3432	011470	042522	046120	041501	
3433	011476	020105	052111	035040	
3434	011504	000040			
3435	011506	005015	025012	020052	NODRTS: .ASCII <15><12><12>/** NO DRIVES TO TEST/<15><12>
3436	011514	047516	042040	044522	
3437	011522	042526	020123	047524	
3438	011530	052040	051505	006524	
3439	011536	012			
3440	011537	120	042522	051523	CNTRDY: .ASCIZ /PRESS "CONT" WHEN RDY/<15><12>

H06

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 73
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0072

3441	011544	021040	047503	052116	
3442	011552	020042	044127	047105	
3443	011560	051040	054504	005015	
3444	011566	000			
3445	011567	110	046101	020124	HLTRQD: .ASCIZ /HAL* REQUESTED/<15><12>
3446	011574	042522	052521	051505	
3447	011602	042524	006504	000012	
3448	011610	051104	053111	020105	DROXDP: .ASCIZ /DRIVE 0 IS LOAD MEDIUM/<15><12>
3449	011616	020060	051511	046040	
3450	011624	040517	020104	042515	
3451	011632	044504	046525	005015	
3452	011640	000			
3453	011641	015	025012	020052	DROPDR: .ASCIZ <15><12>/** DROPPING DRIVE - 20 ERRORS/<15><12>
3454	011646	042040	047522	050120	
3455	011654	047111	020107	051104	
3456	011662	053111	020105	020055	
3457	011670	030062	042440	051122	
3458	011676	051117	006523	000012	
3459	011704	005015	052012	051505	TSTDNR: .ASCII <15><12><12>/TESTING DRIVE /
3460	011712	044524	043516	042040	
3461	011720	044522	042526	040	
3462	011725	040	005015	000	DRVNO: .ASCIZ /<15><12>
3463	011731	104	044522	042526	DRIV: .ASCIZ /DRIVE /
3464	011736	000040			
3465	011740	040503	052122	020056	CART: .ASCIZ /CART. /
3466	011746	000			
3467	011747	123	051105	020056	SERNM: .ASCIZ /SER. NO. /
3468	011754	047516	020056	000040	
3469	011762	005015	040506	052103	FACTBS: .ASCIZ <15><12>/FACTORY /
3470	011770	051117	020131	000	
3471	011775	012	047523	052106	SOFTBS: .ASCII <12>/SOFTWARE /
3472	012002	040527	042522	040	
3473	012007	102	042101	051440	BDSECT: .ASCIZ /BAD SECTORS :/<15><12>
3474	012014	041505	047524	051522	
3475	012022	035040	005015	000	
3476	012027	040	047040	047117	NOFALS: .ASCIZ / NONE/
3477	012034	000105			
3478					
3479	012036	025052	020040	040503	BAD632: .ASCIZ /** CANNOT READ BAD SECTOR TRACK!/<15><12>
3480	012044	047116	052117	051040	
3481	012052	040505	020104	040502	
3482	012060	020104	042523	052103	
3483	012066	051117	052040	040522	
3484	012074	045503	006441	000012	
3485	012102	041536	005015	000	CNTRLC: .ASCIZ /C/<15><12>
3486	012107	136	006532	000012	CNTRLZ: .ASCIZ /Z/<15><12>
3487	012114	051136	005015	000	CNTRLR: .ASCIZ /R/<15><12>
3488	012121	136	006525	000012	CNTRLU: .ASCIZ /U/<15><12>
3489	012126	043536	005015	000	CNTRLG: .ASCIZ /G/<15><12>
3490	012133	015	005012	000	CR2LF: .ASCIZ <15><12><12>
3491	012137	134	000		BKSLSH: .ASCIZ /\
3492	012141	054	000		COMMA: .ASCIZ /\
3493	012143	040	040		SPACE6: .ASCII /\
3494	012145	040			SPACE4: .ASCII /\
3495	012146	040			SPACE3: .ASCII /\
3496	012147	040			SPACE2: .ASCII /\

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 74
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0073

```

3497 012150 000040 SPACE1: .ASCIZ / /
3498 .EVEN
3499 012152 020040 000075 PRMBUF: .ASCIZ / =/
3500 012156 047527 042122 000040 WORDSP: .ASCIZ /WORD /
3501 012164 036440 000040 EQUALS: .ASCIZ / = /
3502 012170 020052 000 PROMPT: .ASCIZ /* /
3503 012173 076 000040 PRMPSP: .ASCIZ /
3504 .EVEN
3505
3506
3507
3508
3509
3510 012176 105037 003126 DFSTRT: CLRB MDFLAG ;SET FLAG FOR 200 START
3511 012202 000403 BR PISET ;CONTINUE
3512
3513 012204 112737 000001 003126 P1STRT: MOV #1 MDFLAG ;SET FLAG FOR 204 START
3514 012212 112737 000061 003155 PISET: MOV #1 PASSNO ;SET PASS NO. = 1
3515 012220 000406 BR CMSTRT ;CONTINUE
3516
3517 012222 112737 000002 003126 P2STRT: MOV #2 MDFLAG ;SET FLAG FOR 220 START
3518 012230 112737 000062 003155 MOV #2 PASSNO ;SET PASS NO. = 2
3519
3520 012236 012737 000340 177776 CMSTRT: MOV #PR7, 2#PS ;BLOCK ALL INTERRUPTS
3521 .SBT'L INITIALIZE THE COMMON TAGS
3522 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3523 012244 012706 001100 MOV #CMTAG, R6 ;FIRST LOCATION TO BE CLEARED
3524 012250 005026 CLR (R6)+ ;CLEAR MEMORY LOCATION
3525 012252 022706 001140 CMP #SWR, R6 ;;DONE?
3526 012256 001374 BNE -6 ;;LOOP BACK IF NO
3527 012260 012706 001100 MOV #STACK, SP ;;SETUP THE STACK POINTER
3528 ;;INITIALIZE A FEW VECTORS
3529 012264 012737 044606 000020 MOV #SCOPE, 2#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3530 012272 012737 000340 000022 MOV #340, 2#IOTVEC+2 ;;LEVEL 7
3531 012300 012737 044102 000030 MOV #ERROR, 2#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3532 012306 012737 000340 000032 MOV #340, 2#EMTVEC+2 ;;LEVEL 7
3533 012314 012737 045320 000034 MOV #TRAP, 2#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3534 012322 012737 000340 000036 MOV #340, 2#TRAPVEC+2 ;;LEVEL 7
3535 012330 012737 044452 000024 MOV #PWRON, 2#PWRVEC ;;POWER FAILURE VECTOR
3536 012336 012737 000340 000026 MOV #340, 2#PWRVEC+2 ;;LEVEL 7
3537 012344 013737 017250 017242 MOV $ENDCT, $EOPCT ;;SETUP END-OF-PROGRAM COUNTER
3538 012352 005037 001316 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3539 012356 112737 000001 001115 MOV #1, $ERMAX ;;ALLOW ONE ERROR PER TEST
3540 012364 012737 012364 001106 MOV #, $LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3541 012372 012737 012372 001110 MOV #, $LPERR ;;SETUP THE ERROR LOOP ADDRESS
3542 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3543 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
3544 012400 013746 000004 MOV 2#ERRVEC, -(SP) ;;SAVE ERROR VECTOR
3545 012404 012737 012440 000004 MOV #64$, 2#ERRVEC ;;SET UP ERROR VECTOR
3546 012412 012737 177570 001140 MOV #DSWR, SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
3547 012420 012737 177570 001142 MOV #DISP, DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
3548 012426 022777 177777 166504 CMP #-1, 2#SWR ;;TRY TO REFERENCE HARDWARE SWR
3549 012434 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3550 ;;AND THE HARDWARE SWR IS NOT = -1
3551 012436 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
3552 012440 012716 012446 64$: MOV #65$, (SP) ;;SET UP FOR TRAP RETURN

```

3553	012444	000002				RTI		
3554	012446	012737	000176	001140	65\$:	MOV	#SWREG,SWR	::POINT TO SOFTWARE SWR
3555	012454	012737	000174	001142		MOV	#DISPREC,DISPLAY	
3556	012462	012637	000004		66\$:	MOV	(SP)+,2#ERRVEC	::RESTORE ERROR VECTOR
3557								
3558	012466	005037	001336			CLR	\$PASS	::CLEAR PASS COUNT
3559	012472	132737	000200	001351		BITB	#APTSIZE,\$ENVM	::TEST USER SIZE UNDER APT
3560	012500	001403				BEQ	67\$	::YES,USE NON-APT SWITCH
3561	012502	012737	001352	001140		MOV	#SWREG,SWR	::NO,USE APT SWITCH REGISTER
3562	012510				67\$:			
3563	012510	000005				RESET		::CLEAR THE UNIBUS
3564	012512	012737	000006	000004		MOV	#6,2#ERRVEC	::SET TIME-OUT VECTOR
3565	012520	005037	000006			CLR	2#ERRVEC+2	
3566	012524	105037	003143			CLRB	TSTTYP	::CLEAR OFFSET TEST IDENTIFIER
3567	012530	105037	003137			CLRB	DRVERS	::CLEAR ERROR COUNT FOR CURRENT DRIVE
3568	012534	105037	003135			CLRB	ERRCNT	::CLEAR ERROR RETRY COUNT
3569	012540	105037	001102			CLRB	\$STNM	::SET TEST NO. = 0
3570	012544	112737	000001	003152		MOVB	#1,HLPOVL	::SET HLP FILE OVLD INDICTR
3571	012552	104401	007572			TYPE	DZREQ	::TYPE PROGRAM I.D.
3572	012556	105737	003153			TSTB	NOTYPE	::SEE IF OPERATOR NOTE SHOULD BE TYPED
3573	012562	001004				BNE	40\$	::BR IF NOT
3574	012564	104401	054342			TYPE	NOTMSG	::TYPE OPERATOR NOTE
3575	012570	105237	003153			INCB	NOTYPE	::SET FLAG FOR NEXT TIME
3576	012574	012737	020634	000060	40\$:	MOV	#KBDHDL,2#TKVEC	::LOAD VECTOR FOR TTY KBD
3577	012602	012737	000200	000062		MOV	#PR4,2#TKVEC+2	::SET KBD PRIORITY = 4
3578	012610	013701	003050			MOV	RKVEC,R1	::ADDR. OF RK06 VECTOR STORAGE
3579	012614	012721	034722			MOV	#I,INTR,(R1)+	::SET IT TO RK06 HANDLER
3580	012620	013711	003052			MOV	RKPRI,(R1)	::SET RK06 PRIORITY
3581	012624	105037	003127			CLRB	TSTING	::CLEAR "RUNNING TESTS" FLAG
3582	012630	012737	000000	177776		MOV	#PRO,2#PS	::ALLOW ALL INTERRUPTS AGAIN
3583	012636	012701	005162			MOV	#PRVCMO,R1	::ZERO OUT PREVIOUS COMMAND
3584	012642	012700	000006			MOV	#6,RO	::SET COUNTER
3585	012646	005021			42\$:	CLR	(R1)+	::ZERO A WORD
3586	012650	005300				DEC	RO	
3587	012652	001375				BNE	42\$	::BR IF NOT DONE YET
3588	012654	005037	005432			CLR	STALLS	::DON'T ALLOW STALLS YET
3589	012660	004737	021136			JSR	PC,GTSWRG	::OPEN SOFTWARE SWR FOR MODIFICATION
3590	012664	012737	176543	042350	44\$:	MOV	#176543,\$HINUM	::INIT. PSEUDO-RANDOM NOS.
3591	012672	012737	123456	042352		MOV	#123456,\$LONUM	
3592	012700	004737	024176			JSR	PC,GETTYP	
3593	012704	105737	003126			TSTB	MDFLAG	::SEE IF RK06 OR RK07 & SETUP
3594	012710	001034				BNE	ALLSYS	::SEE IF DEFAULT MODE
3595	012712	013700	001400			MOV	\$VECT1,RO	::BR IF NOT DEFAULT MODE
3596	012716	013737	001404	003046		MOV	\$BASE,RKBAS	::GET APT RK06 VECTOR AND PRTY
3597	012724	132737	000200	001351		MOV	\$BIT7,\$ENVM	::GET APT RK06 BASE ADDRESS
3598	012732	001005				BITB	18\$	::SEE IF NO SIZING
3599	012734	012700	120210			BNE	18\$	::BR IF NO SIZING
3600	012740	012737	177440	003046		MOV	#AVECT1,RO	::GET DEFAULT VECTOR AND PRIORITY
3601	012746	110037	003050		18\$:	MOV	#ABASE,RKBAS	::GET DEFAULT BASE ADDRESS
3602	012752	105037	003051			MOVB	RO,RKVEC	::STORE VECTOR
3603	012756	000300				CLRB	RKVEC+1	::CLEAR HI BYTE
3604	012760	042700	177437			SWAB	RO	::GET PRTY INTO BITS 5-7
3605	012764	010037	003052			BIC	#177437,RO	::CLEAR OTHER BITS
3606	012770	112737	000101	005510		MOV	RO,RKPRI	::STORE RK06 PRIORITY
3607	012776	000137	013774			MOVB	#A,SUBSYS	::SET SUBSYSTEM NAME = A
3608						JMP	DFLDR	::GO CHECK ALL DRIVES

```

3609
3610
3611
3612
3613
3614
3615
3616
3617 013002 104401 001325
3618 013006 105037 001102
3619 013012 112703 000101
3620 013016 012701 005512
3621 013022 012700 000021
3622 013026 005021
3623 013030 005300
3624 013032 001375
3625 013034 012704 005512
3626
3627 013040 110337 010112
3628 013044 104401 010103
3629 013050 004737 023424
3630 013054 013002
3631 013056 013002
3632 013060 013044
3633 013062 005700
3634 013064 001767
3635 013066 022700 000017
3636 013072 002005
3637 013074 104401 005212
3638 013100 104401 001325
3639 013104 000757
3640 013106 005001
3641 013110 126127 005212 000060
3642 013116 103766
3643 013120 126127 005212 000067
3644 013126 101362
3645 013130 005201
3646 013132 020100
3647 013134 001406
3648 013136 126127 005212 000054
3649 013144 001353
3650 013146 005201
3651 013150 000757
3652
3653 013152 012701 005556
3654 013156 005021
3655 013160 005021
3656 013162 005021
3657 013164 005011
3658 013166 005000
3659 013170 116001 005212
3660 013174 042701 000060
3661 013200 116061 005212 005556
3662 013206 062700 000002
3663 013212 105760 005211
3664 013216 001364

;*****
;
;SBTTL TEST 0 PARAMETER INPUT ROUTINES
;
;*****
;READ AND STORE ALL SUBSYSTEMS AND DRIVES TO TEST
ALLSYS: TYPE $CRLF ;TYPE <CR>, <LF>
CLR $STNM ;SET TEST NO. = 0
MOV #A,R3 ;INIT SUBSYS INDICATOR
MOV #DRVLST,R1 ;GET DRIVE LIST ADDRESS
MOV #17,R0 ;CLEAR OUT THE DRIVE LIST
2$: CLR (R1)+
DEC R0
BNE 2$
MOV #DRVLST,R4 ;GET STARTING ADRS OF DRIVE LIST
;ASK FOR DRIVES TO TEST ON THIS SUBSYSTEM
ALLSY1: MOV R3,SYSDRV+7 ;SET SUBSYS NAME FOR PRINTOUT
6$: TYPE SYSDRV ;ASK FOR DRIVES ON THIS SUBSYSTEM
JSR PC,RDCHRS ;READ INPUT STRING
ALLSYS ;(+C) RETURN ADDRESS
ALLSYS ;(+Z) RETURN ADDRESS
6$ ;(+U) RETURN ADDRESS
TST R0 ;SEE IF NULL INPUT
BEQ 6$ ;BR TO ASK AGAIN
CMP #15,R0 ;SEE IF TOO MANY CHARS TYPED
BGE 12$ ;BR IF NOT TOO MANY
10$: TYPE ,BUFF0 ;ECHO BAD INPUT
TYPE ,SQUES ;TYPE <?> AND <CR>, <LF>
BR 6$ ;GO ASK AGAIN
12$: CLR R1 ;CLEAR CHAR POINTER
14$: CMPB BUFF0(R1),#0 ;SEE IF DRIVE NO. < 0
BLO 10$ ;BR TO ECHO BAD INPUT
CMPB BUFF0(R1),#7 ;SEE IF CHAR > 7
BHI 10$ ;BR TO ECHO BAD INPUT
INC R1 ;INCR CHAR POINTER
CMP R1,R0 ;SEE IF MORE CHARS TO CHECK
BEQ 16$ ;BR IF ALL CHARS CHECKED
CMPB BUFF0(R1),#', ;SEE IF THIS IS A COMMA
BNE 10$ ;BR IF NOT A COMMA
INC R1 ;INCR CHAR POINTER
BR 14$ ;BR TO CONTINUE CHECKING CHARS
;PUT DRIVES FOR THIS SUBSYSTEM INTO SCRATCH LIST
16$: MOV #SCRLST,R1 ;GET SCRATCH DRIVE LIST ADRS
CLR (R1)+ ;CLEAR OUT SCRATCH DRIVE LIST
CLR (R1)+
CLR (R1)+
CLR (R1)
CLR R0
18$: MOVB BUFF0(R0),R1 ;INIT CHAR POINTER
BIC #0,R1 ;GET A DRIVE NO.
MOVB BUFF0(R0),SCRLST(R1) ;STRIP ASCII BITS
ADD #2,R0 ;SET ASCII DRV NO.
TSTB BUFF0-1(R0) ;POINT TO NEXT DRIVE NO.
BNE 18$ ;SEE IF NULL CHAR YET
;BR IF NOT DONE YET

```

```

3665 ;LOAD DRIVES FROM SCRATCH LIST INTO TOTAL DRIVE LIST
3666 013220 005000 ALLSY3: CLR R0 ;INIT SCRATCH DRV LIST POINTER
3667 013222 116014 005556 ALLSY2: MOV B SCRLST(P0),(R4) ;MOVE THIS DRIVE INTO ACTUAL LIST
3668 013226 001411 BEQ 22$ ;BR IF DRIVE NOT MARKED
3669 013230 005204 INC R4 ;INCR DRIVE LIST POINTER
3670 013232 110324 MOV B R3,(R4)+ ;PUT SUBSYS LETTER INTO HI BYTE
3671 013234 020427 005554 CMP R4,*DRVLST+34. ;SEE IF DRIVE LIST OVERFLOW
3672 013240 103404 BLO 22$ ;BR IF NO OVERFLOW YET
3673 013242 104401 010245 TYPE ,DROVFL ;TYPE "MORE THAN 16 DRIVES ENTERED !"
3674 013246 000137 013002 JMP ALLSYS ;GO START OVER, ASKING FOR DRIVES
3675 013252 005200 22$: INC R0 ;INCR SCRATCH DRIVE LIST POINTER
3676 013254 020027 000010 CMP R0,*8. ;SEE IF DONE WITH THIS SUBSYS
3677 013260 103760 BLO ALLSY2 ;BR IF NOT YET
3678 ;ECHO DRIVES TO TEST ON THIS SUBSYSTEM
3679 013262 104401 010130 TYPE ,WILTST ;TYPE "WILL TEST DRIVE(S) "
3680 013266 005037 005444 CLR SCRACH ;INIT CHAR OUTPUT BUF
3681 013272 005000 CLR R0 ;INIT SCRATCH LIST POINTER
3682 013274 005001 CLR R1 ;INIT COUNT OF LISTED DRIVES FOR SUBSYS
3683 013276 116037 005556 005444 23$: MOV B SCRLST(R0),SCRACH ;SEE IF THIS DRIVE LISTED
3684 013304 001407 BEQ 26$ ;BR IF DRIVE NOT LISTED
3685 013306 005701 TST R1 ;SEE IF FIRST TIME HERE
3686 013310 001402 BEQ 24$ ;BR IF FIRST TIME
3687 013312 104401 012141 TYPE ,COMMA ;TYPE A COMMA
3688 013316 104401 005444 24$: TYPE ,SCRACH ;TYPE A DRIVE NO.
3689 013322 005201 INC R1 ;INCR. COUNT OF LISTED DRIVES FOR THIS SUBSYS
3690 013324 005200 26$: INC R0 ;INCR LIST POINTER
3691 013326 020027 000010 CMP R0,*8. ;SEE IF DONE CHECKING LIST YET
3692 013332 002761 BLT 23$ ;BR IF NOT DONE YET
3693 013334 110337 010167 MOV B R3,ONSUBS+11. ;SET SUBSYS NO. IN MSG
3694 013340 104401 010154 TYPE ,ONSUBS ;TYPE "ON SUBSYSTEM "
3695 013344 005203 INC R3 ;INCR SUBSYSTEM NAME
3696 013346 020327 000120 CMP R3,*'P ;SEE IF NAME > P
3697 013352 101035 BHI ASKSYS ;BR IF YES
3698 013354 105737 003126 TST B MDFLAG ;SEE IF 200 START
3699 013360 001002 BNE 28$ ;BR IF NOT
3700 013362 000137 013734 JMP DRVTST ;START PASS 1
3701 ;ASK IF THERE IS ANOTHER SUBSYSTEM TO SPECIFY
3702 013366 104401 010174 28$: TYPE ,ANOTHR ;ASK FOR ANOTHER SUBSYS
3703 013372 004737 023424 JSR PC,ROCHRS ;READ RESPONSE
3704 013376 013002 ALLSYS ;(IC) RETURN ADDRESS
3705 013400 013002 ALLSYS ;(IZ) RETURN ADDRESS
3706 013402 013366 28$ ;(IU) RETURN ADDRESS
3707 013404 005700 TST R0 ;SEE IF NULL INPUT
3708 013406 001417 BEQ ASKSYS ;BR IF NULL INPUT, TO ASSUME <N>
3709 013410 022737 000131 005212 CMP #'Y,BUFF0 ;SEE IF <Y> TYPED
3710 013416 001002 BNE 30$ ;BR IF NOT <Y>
3711 013420 000137 013040 JMP ALLSY1 ;GO ASK FOR DRIVES ON NEXT SUBSYS
3712 013424 022737 000116 005212 30$: CMP #'N,BUFF0 ;SEE IF <N> TYPED
3713 013432 001405 BEQ ASKSYS ;BR IF <N>
3714 013434 104401 005212 TYPE ,BUFF0 ;ECHO BAD INPUT
3715 013440 104401 001324 TYPE ,SQUES ;TYPE <?> AND <CR>,<LF>
3716 013444 000750 BR 28$ ;GO ASK AGAIN
3717
3718 ;204 OR 220 START - GET NAME OF SUBSYSTEM TO TEST NEXT
3719 013446 105037 001102 ASKSYS: CLR B $TSTNM ;SET TEST NO. = 0
3720 013452 104401 007752 TYPE ,TYPYS ;ASK FOR NEXT SUBSYS TO TEST

```

```

3721 013456 004737 023424 JSR PC, RDCHRS ; READ RESPONSE
3722 013462 013002 ALLSYS ; (1C) RETURN ADDRESS
3723 013464 013446 ASKSYS ; (1Z) RETURN ADDRESS
3724 013466 013446 ASKSYS ; (1D) RETURN ADDRESS
3725 013470 005700 TST RO ; SEE IF NULL INPUT
3726 013472 001765 BEQ ASKSYS ; BR IF NULL, TO ASK AGAIN
3727 013474 023727 005212 000101 CMP BUFF0, #'A ; SEE IF A CHAR < A TYPED
3728 013502 103005 BHIS B$ ; BR IF NOT
3729 013504 104401 005212 4$: TYPE ,BUFF0 ; ECHO BAD INPUT
3730 013510 104401 001324 TYPE $QUES ; TYPE '?' AND <CR>, <LF>
3731 013514 000754 BR ASKSYS ; GO ASK AGAIN
3732 013516 023727 005212 000120 6$: CMP BUFF0, #'P ; SEE IF A CHAR > P TYPED
3733 013524 101367 BHIS 4$ ; BR IF YES
3734 013526 113737 005212 005510 MOV8 BUFF0, SUBSYS ; STORE CURRENT SUBSYSTEM NAME
3735 013534 012700 005513 MOV #DRVLST+1, RO ; SET DRIVE LIST POINTER
3736 013540 122037 005510 8$: CMPB (RO)+, SUBSYS ; SEE IF A DRIVE LISTED FOR THIS SUBSYS
3737 013544 001410 BEQ 9$ ; BR IF YES
3738 013546 005200 INC RO ; INCR POINTER
3739 013550 020027 005553 CMP RO, #DRVLST+33. ; SEE IF WHOLE LIST CHECKED YET
3740 013554 103771 BLO B$ ; BR IF NOT YET
3741 013556 104401 010652 TYPE ,NODRLS ; TYPE "NO DRIVES LISTED FOR SUBSYS"
3742 013562 000137 013446 JMP ASKSYS ; GO ASK FOR A SUBSYS AGAIN
3743 013566 162700 000002 9$: SUB #2, RO ; POINT TO LOW BYTE OF LIST ENTRY
3744 013572 010037 005554 MOV RO, DRVPTR ; STORE POINTER
3745 ; OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
3746 013576 013746 003046 MOV RKBAS, -(SP) ; PUT OLD VALUE ON STACK
3747 013602 104401 007660 TYPE RKBADR ; TYPE "RK06 BUS ADR = "
3748 013606 004737 021026 JSR PC, GETPRM ; TYPE OLD, GET NEW RKBAS VALUE
3749 013612 012637 003046 MOV (SP)+, RKBAS ; STORE NEW VALUE
3750 ; OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
3751 013616 013746 003050 MOV RKVEC, -(SP) ; PUT OLD VALUE ON STACK
3752 013622 104401 007704 TYPE RKVADR ; TYPE "RK06 VEC ADR = "
3753 013626 004737 021026 JSR PC, GETPRM ; TYPE OLD, GET NEW RKVEC VALUE
3754 013632 012637 003050 MOV (SP)+, RKVEC ; STORE NEW VALUE
3755 ; OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
3756 013636 013746 003052 11$: MOV RKPRI, -(SP) ; GET OLD VALUE OF PRIORITY
3757 013642 006316 ASL (SP) ; GET IT INTO BITS 0-2
3758 013644 006316 ASL (SP)
3759 013646 006316 ASL (SP)
3760 013650 000316 SWAB (SP)
3761 013652 104401 007727 TYPE RKPRTY ; TYPE "RK06 PRIORITY = "
3762 013656 004737 021026 JSR PC, GETPRM ; TYPE OLD, GET NEW RKPRI VALUE
3763 013662 012600 MOV (SP)+, RO
3764 013664 020027 000004 CMP RO, #4 ; SEE IF AT LEAST LEVEL 4
3765 013670 002414 000007 BLT 12$ ; BR IF NEW VALUE TOO SMALL
3766 013672 020027 000007 CMP RO, #7 ; SEE IF LEVEL 7 OR LESS
3767 013676 003011 BGT 12$ ; BR IF NEW VALUE TOO LARGE
3768 013700 000300 SWAB RO ; GET PRIORITY INTO BITS 5-7
3769 013702 006200 ASR RO
3770 013704 006200 ASR RO
3771 013706 006200 ASR RO
3772 013710 010037 003052 MOV RO, RKPRI ; STORE NEW VALUE
3773 013714 104401 001325 TYPE $CRLF
3774 013720 000405 BR DRVTST
3775 013722 104401 005212 12$: TYPE ,BUFF0 ; ECHO BAD INPUT
3776 013726 104401 001324 TYPE $QUES

```

```

3777 013732 000741 BR 11$ ;GO ASK AGAIN
3778
3779 013734 113737 003155 010331 DRVTST: MOV8 PASSNO,SRPASS+18. ;SET PASS NO. FOR PRINTOUT
3780 013742 104401 010307 TYPE SRPASS ;TYPE "STARTING PASS X"
3781 013746 122737 000061 003155 CMP8 #1,PASSNO ;SEE IF PASS 1
3782 013754 001002 BNE 2$ ;BR IF NOT PASS 1
3783 013756 000137 014346 JMP TST1 ;START PASS 1
3784 013762 112737 000004 001102 2$: MOV8 #4,$TSTNM ;SET TEST NUMBER = 4
3785 013770 000137 015146 JMP TST5 ;START PASS 2
3786
3787
3788 ;ADR 200 START - COMPILE SCRATCH LIST OF DRIVES TO SIZE
3789 013774 005000 DFLTR: CLR R0 ;CLEAR DRIVE NUMBER
3790 013776 013703 001406 MOV $DEV8,R3 ;GET APT DEVICE MAP
3791 014002 132737 000200 001351 BIT8 #BIT7,$ENV8 ;SEE IF NO SIZING
3792 014010 001002 BNE 1$ ;BR IF NO SIZING
3793 014012 012703 000377 MOV #377,R3 ;SET UP FOR SIZING
3794 014016 012701 000001 1$: MOV #BIT0,R1 ;SET BIT POINTER
3795 014022 105060 005556 2$: CLR8 SCRLST(R0) ;INITIALIZE DRIVE ENTRY
3796 014026 030103 BIT R1,R3 ;SEE IF THIS DRIVE IS REQUESTED
3797 014030 001405 BEQ 4$ ;BR IF NOT
3798 014032 110060 005556 MOV8 R0,SCRLST(R0) ;MARK DRIVE IN SCRATCH LIST
3799 014036 152760 000060 005556 BIS8 #0,SCRLST(R0) ;CONVERT NO. TO ASCII
3800 014044 006301 4$: ASL R1 ;SHIFT BIT POINTER
3801 014046 005200 INC R0 ;INCREMENT DRIVE NUMBER
3802 014050 022700 000010 CMP #10,R0 ;SEE IF DONE MARKING ALL DRIVES
3803 014054 001362 BNE 2$ ;BR IF NOT DONE YET
3804 014056 132737 000200 001351 BIT8 #BIT7,$ENV8 ;SEE IF PROGRAM SHOULD SIZE
3805 014064 001416 BEQ 8$ ;BR IF YES
3806 014066 012704 005512 10$: MOV #DRVLST,R4 ;GET DRIVE LIST ADDRESS
3807 014072 113703 005510 MOV8 SUBSYS,R3 ;GET SUBSYS NAME IN R3
3808 014076 010401 MOV R4,R1 ;GET DRIVE LIST ADRS
3809 014100 010437 005554 MOV R4,DRVPT8 ;SET POINTER TO FIRST DRIVE TO TEST
3810 014104 012700 000021 MOV #17,R0 ;SET COUNTER
3811 014110 005021 11$: CLR (R1)+ ;CLEAR A DRIVE LIST WORD
3812 014112 005300 DEC R0 ;DECREMENT COUNTER
3813 014114 001375 BNE 11$ ;BR IF NOT DONE YET
3814 014116 000137 013220 JMP ALLSY3 ;GO MOVE SCRATCH LIST INTO LIST
3815 ;REMOVE NON-EXISTENT DRIVES FROM SCRATCH LIST
3816 014122 005000 8$: CLR R0 ;CLEAR DRIVE NUMBER
3817 014124 105760 005556 12$: TST8 SCRLST(R0) ;SEE IF THIS DRIVE IS MARKED IN SCRATCH LIST
3818 014130 001475 BEQ 14$ ;BR IF NOT MARKED
3819 014132 010037 005430 MOV R0,DRIVE ;SET DRIVE NUMBER FOR SCNDRV
3820 014136 004737 021170 JSR PC,INITSS ;CLEAR DRIVER AND INIT S.S.
3821 014142 042712 000100 3$: BIC #IE,(R2) ;DISABLE RK06 INTERRUPT
3822 014146 113737 005430 000010 MOV8 DRIVE,RKCS2
3823 014154 105737 003133 TST8 TYPFMT ;SEE IF RK07
3824 014160 001003 BNE 5$ ;BR IF YES
3825 014162 012712 000001 MOV #GO,(R2) ;ELSE SET GO BIT FOR RK06 IN RKCS1
3826 014166 000402 BR 7$
3827 014170 012712 002001 5$: MOV #<CDT!GO>,(R2) ;ELSE SET GO FOR RK07 IN RKCS1
3828 014174 005037 005444 7$: CLR SCRACH ;CLEAR STALL CTR
3829 014200 005237 005444 15$: INC SCRACH ;INCREMENT STALL COUNTER
3830 014204 001375 BNE 15$ ;STALL FOR SEVERAL MILLI-SEC
3831 014206 032762 001000 000010 BIT #MDS,RKCS2(R2) ;SEE IF MDS ERROR
3832 014214 001417 BEQ 18$ ;BR IF NOT

```

```

3833 014216 112765 000101 000001      MOVB  #SELDIV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
3834 014224 011265 000016      MOV   (R2),P.CSI(R5) ;GET RKCS1 FOR REPORT
3835 014230 004737 037644      JSR   PC,1,CST1 ;GET OTHER RK611 REGS FOR REPORT
3836 014234 004737 031224      JSR   PC,REPSUP ;SET UP ERROR REPORT
3837 014240 104052      ERROR 52 ;REPORT MDS ERROR
3838 014242 052737 000200 005424      BIS   #ABORT,RECODE ;SET ABORT FLAG
3839 014250 000137 033336      JMP   ALLTRM ;GO ABORT TESTING
3840 014254 004737 021170 18$:      JSR   PC,INITSS ;INIT THE S.S.
3841 014260 112765 000141 000001      MOVB  #R0STAT,P.CMND(R5) ;SET READ STATUS COMMAND
3842 014266 012737 020500 003056      MOV   #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADR
3843 014274 012737 000377 005476      MOV   #377,NEWON ;INIT ONLINE INDICATOR
3844 014302 004737 030146      JSR   PC,DVVCAL ;READ DRIVE STATUS
3845 014306 012737 031504 003056      MOV   #ERRHDL,A.ABNL ;RESTORE ABNORMAL RETURN ADDRESS
3846 014314 022737 000377 005476      CMP   #377,NEWON ;SEE IF THIS DRIVE EXISTS
3847 014322 001006      BNE   16$ ;BR IF NO
3848 014324 005200 14$:      INC   R0 ;RETURN HERE IF DRIVE IS USEABLE
3849 014326 022700 000010      CMP   #10,R0 ;SEE IF DONE CHECKING LIST
3850 014332 001274      BNE   12$ ;BR IF NOT DONE YET
3851 014334 000137 014066      JMP   10$ ;GO BACK TO LIST SELECTED DRIVES
3852 014340 105060 005556 16$:      CLRB  SCRLST(R0) ;REMOVE INVALID DRIVE FROM LIST
3853 014344 000767      BR    14$ ;CONTINUE CHECKING THE LIST
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871 014346 000004      *****
3872 014350 012737 000001 001334 1ST1: SCOPE
3873 014356 004737 022042      MOV   #1,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
3874 014362 004737 021170      JSR   PC,SETUP ;SET UP FOR LOOP ON ERROR
3875 014366 004737 020774      JSR   PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3876 014372 004737 022110      JSR   PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888

```

TEST 1 MOUNTING OF TEST CARTRIDGE FOR PASS 1
*
* THE PROGRAM TYPES "STARTING PASS 1". THEN, THE OPERATOR
* MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
* AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
* FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
* DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
* ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
* IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS.

TEST 2 BASIC READ/WRITE DATA TEST
*
* THE PROGRAM PERFORMS A WRITE AND WRITE CHECK OPERATION
* USING A "WORST CASE" DATA PATTERN, AT THE APPROPRIATE SECTOR
* FOR THIS DRIVE (FROM TABLE A) ON ALL SURFACES. THE PURPOSE
* OF THIS OPERATION IS TO VERIFY THE BASIC READ/WRITE

```

3889      ;* CAPABILITY OF THE DRIVE ON PASS 1. THE ENTIRE SECTOR
3890      ;* IS WRITTEN WITH THE REPETITION OF THE DATA PATTERN SHOWN
3891      ;* IN TABLE B.
3892      ;*
3893      ;*****
3894      TST2: SCOPE
3895      MOV     #2,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3896      JSR     PC,SETUP      ;;SET UP FOR LOOP ON ERROR
3897      JSR     PC,INITSS     ;;INITIALIZE DRIVER PARAMS AND SUB-SYS
3898      JSR     PC,PREPKB     ;;PREPARE FOR POSSIBLE KBD INPUT
3899      MOV     #1,TSTING     ;;ALLOW TTY ESCAPE
3900      ;LOAD RWBUF WITH DATA PATTERN
3901      JSR     PC,LODSEC
3902      ;SET UP PARAMETERS
3903      MOV     LOGDRV,RO      ;;GET LOGICAL DRIVE NUMBER
3904      ASL     RO             ;;CONVERT IT TO INDEX
3905      ASL     RO
3906      TSTB    TYPFMT        ;;SEE IF RK07
3907      BNE     1$            ;;BR IF YES
3908      MOV     TABLEA(RO),P.CYLN(R5) ;;GET CYL # FOR THIS DRIVE
3909      ADD     #2,RO          ;;INCREMENT INDEX
3910      MOV     TABLEA(RO),P.SECT(R5) ;;GET SECTOR # FOR THIS DRIVE
3911      BR      2$
3912
3913      1$: MOV     TABL7A(RO),P.CYLN(R5) ;;GET RK07 TABLE
3914      ADD     #2,RO
3915      MOV     TABL7A(RO),P.SECT(R5) ;;GET SECTOR # FOR THE DRV
3916
3917      2$: MOV     #RWBUF,P.BALD(R5)    ;;SET BUS ADDRESS
3918      MOV     #-400,P.WC(R5)          ;;SET WORD COUNT FOR 1 SECTOR
3919      ;WRITE THE DATA
3920      6$: MOV     #WRDATA,P.CMND(R5)   ;;SET WRITE COMMAND
3921      JSR     PC,DRVCAL              ;;WRITE THE SECTOR
3922      ;PERFORM WRITE CHECK OF SECTOR
3923      MOV     #WRTCHK,P.CMND(R5)      ;;SET WRITE CHECK COMMAND
3924      JSR     PC,DRVCAL              ;;PERFORM THE WRITE CHECK
3925      ;INCREMENT TRACK ADDRESS
3926      INCB    P.TRCK(R5)              ;;INCREMENT TRACK NO.
3927      CMPB    P.TRCK(R5),#2          ;;SEE IF ALL SURFACES WRITTEN YET
3928      BLOS    6$                    ;;BR IF NOT YET
3929
3930
3931
3932
3933      ;*****
3934      ;TEST 3 WRITE OVRWRT AND COMPAT CYL BLOCKS FOR CURRENT DRIVE
3935      ;* ALL SECTORS ARE WRITTEN FOR THE CURRENT DRIVE WITHIN THE
3936      ;* CYLINDER BLOCKS IN TABLEC ON ALL SURFACES USING A SINGLE
3937      ;* REPEATED WORD OF THE PSEUDO-RANDOM DATA PATTERN IN TABLEG. LOGICAL DRIVE 0
3938      ;* USES WORD 0. LOGICAL DRIVE 14 USES WORD 14, ETC. THESE CYLINDER
3939      ;* BLOCKS ARE THE OVERWRITE AND COMPATIBILITY TEST DATA TO
3940      ;* BE USED IN PASS 2.
3941      ;*****
3942      TST3: SCOPE
3943      MOV     #3,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3944      JSR     PC,SETUP      ;;SET UP FOR LOOP ON ERROR

```

Address	Hex 1	Hex 2	Hex 3	Hex 4	Label	Operation	Comment
3945	014604	004737	021170		JSR	PC,INITSS	;INITIALIZE DRIVER PARAMS AND SUB-SYS
3946	014610	004737	020774		JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
3947	014614	004737	022642		JSR	PC,INTBLD	;LOAD INITIAL VALUES INTO TABLED
3948	014620	005000			CLR	RO	;INITIALIZE TABLEC INDEX TO 0
3949	014622	105737	003133	4\$:	TSTB	TYPFMT	;SEE IF RK07
3950	014626	001013			BNE	1\$	;BR IF YES
3951	014630	016065	006026	000002	MOV	TABLEC(RO),P.CYLN(R5)	;GET START OVRWRT CYL FROM TABLEC
3952	014636	004737	023106		JSR	PC,WRTBLK	;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S
3953	014642	062700	000002		ADD	#2,RO	;INCREMENT TABLEC POINTER
3954	014646	016065	006026	000002	MOV	TABLEC(RO),P.CYLN(R5)	;GET START COMPAT CYL FROM TABLEC
3955	014654	000412			BR	2\$	
3956							
3957	014656	016065	006060	000002	1\$:	MOV	TABL7C(RO),P.CYLN(R5)
3958	014664	004737	023106		JSR	PC,WRTBLK	
3959	014670	062700	000002		ADD	#2,RO	
3960	014674	016065	006060	000002	MOV	TABL7C(RO),P.CYLN(R5)	
3961	014702	004737	023106	2\$:	JSR	PC,WRTBLK	;WRITE THIS COMPAT CYL BLK ON ALL SURF'S
3962	014706	062700	000002		ADD	#2,RO	;INCREMENT TABLEC POINTER
3963	014712	020027	000030		CMP	RO,#30	;SEE IF ON LAST CURRENT ZONE YET
3964	014716	002003			BGE	8\$	;BR IF YES
3965	014720	004737	023022		JSR	PC,ROTBLD	;ROTATE TABLED SEVERAL SECTORS
3966	014724	000736			BR	4\$	;CONTINUE WRITING BLOCKS
3967	014726	004737	022642	8\$:	JSR	PC,INTBLD	;RE-LOAD INITIAL VALUES INTO TABLED
3968	014732	105737	003133		TSTB	TYPFMT	;SEE IF RK07
3969	014736	001004			BNE	3\$	;BR IF YES
3970	014740	016065	006026	000002	MOV	TABLEC(RO),P.CYLN(R5)	;GET START OVRWRT CYL FROM TABLEC
3971	014746	000403			BR	5\$	
3972							
3973	014750	016065	006060	000002	3\$:	MOV	TABL7C(RO),P.CYLN(R5)
3974	014756	004737	023106	5\$:	JSR	PC,WRTBLK	;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S
3975							
3976							
3977							
3978							
3979							
3980							
3981							
3982							
3983							
3984							
3985							
3986							
3987							
3988							
3989							
3990							
3991							
3992							
3993							
3994							
3995							
3996							
3997	014762	000004			1\$T4:	SCOPE	
3998	014764	012737	000004	001334	MOV	#4,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
3999	014772	004737	022042		JSR	PC,SETUP	;SET UP FOR LOOP ON ERROR
4000	014						

```

4001 015002 004737 020774 JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
4002 015006 105037 003127 CLRB TSTING ;DON'T ALLOW TTY ESCAPE
4003 015012 004737 022472 JSR PC,DMCART ;DIRECT OPERATOR TO DISMOUNT PACK
4004 015016 062737 000002 ADD #2,DRVPTA ;INCREMENT DRIVE LIST POINTER
4005 015024 013700 005554 MOV DRVPTA,RO
4006 015030 005710 TST (PO) ;SEE IF ALL DONE WITH PASS 1 YET
4007 015032 001420 BEQ B$ ;BR IF DONE WITH PASS 1
4008 015034 126037 000001 005510 CMPB 1(RO),SUBSYS ;SEE IF DONE WITH THIS SUBSYS YET
4009 015042 001004 BNE 4$ ;BR IF DONE WITH THIS SUBSYS
4010 015044 105037 001102 CLRB $TSTNM ;CLEAR TEST NUMBER
4011 015050 000137 014346 JMP TST1 ;DO PASS 1 ON NEXT DRIVE OF SUBSYS
4012 015054 116037 000001 010517 4$: MOVB 1(RO),STR204+39. ;PUT SUBSYS NAME INTO MSG
4013 015062 104401 010450 5$: TYPE ,STR204 ;TYPE RESTART MSG FOR PASS 1 ON NEXT SUBSYS
4014 015066 000000 6$: HALT ;HALT THE PROCESSOR SO THAT
4015 015070 000137 000204 JMP 204 ;OPERATOR CAN RESTART AT PROPER ADRS
4016 015074 012737 005512 005554 8$: MOV #DRVLIST,DRVPTA ;RE-INIT DRIVE LIST POINTER
4017 015102 105737 003126 TSTB MDFLAG ;SEE IF 200 START
4018 015106 001007 BNE 12$ ;BR IF NOT
4019 015110 112737 000062 010331 MOVB #'2,SRPASS+18. ;SET PASS NO. = 2 FOR PRINTOUT
4020 015116 104401 010307 TYPE SRPASS ;TYPE "STARTING PASS 2"
4021 015122 000137 015146 JMP TST5 ;GO START PASS 2
4022 015126 112737 000101 010574 12$: MOVB #'A,STR220+39. ;PUT SUBSYS NAME INTO MSG
4023 015134 104401 010525 TYPE ,STR220 ;TYPE RESTART MSG FOR PASS 2 ON FIRST SUBSYS
4024 015140 000000 HALT ;HALT PROCESSOR SO THAT OPERATOR CAN RESTART
4025 015142 000137 000220 JMP 220 ; AT THE PROPER ADRS

```

```

*****
;TEST 5 MOUNTING OF TEST CARTRIDGE FOR PASS 2

```

```

;
; THE PROGRAM TYPES "STARTING PASS 2". THEN, THE OPERATOR
; MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
; AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
; FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
; DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
; ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
; IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS (IF STARTED
; AT ADDRESS 204 OR 220).

```

```

*****

```

```

4041 015146 000004 TST5: SCOPE
4042 015150 012737 000005 001334 MOV #5,$TSTNM ;SET TEST NUMBER IN APT MAIL BOX
4043 015156 004737 022042 JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
4044 015162 004737 021170 JSR PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4045 015166 004737 020774 JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
4046 015172 112737 000062 003155 MOVB #'2,PASSNO ;SET PASS NO. = 2
4047 015200 004737 022110 JSR PC,MT CART

```

```

*****
;TEST 6 OVERWRITE TEST

```

```

;
; THE PROGRAM PROCEEDS TO TEST THIS DRIVE'S OVERWRITE CAPABILITY.
; FIRST, THE APPROPRIATE CYLINDERS IN TABLE E FOR THIS DRIVE

```

```

4057      * ARE OVERWRITTEN, ON EACH SURFACE. THE DATA USED IS A
4058      * REPETITION OF A SINGLE WORD OF THE PATTERN IN TABLE G.
4059      * DRIVE 0 USES WORD 0, DRIVE 1 USES WORD 1, DRIVE 10 USES
4060      * WORD 10, ETC.
4061      * THEN, EACH CYLINDER OVERWRITTEN IS READ BACK BY THIS DRIVE
4062      * WITH THE FOLLOWING RANGE OF OFFSET VALUES : 100, 200, 300,
4063      * 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-IN., IN BOTH
4064      * THE (+) AND (-) DIRECTIONS.
4065      * THE PROGRAM SCANS FOR DATA CHECK ERRORS DURING THIS READ, AND IF
4066      * ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA HAD NOT
4067      * BEEN CORRECTLY OVERWRITTEN, AND A SCORE FOR THAT DRIVE IS
4068      * DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR,
4069      * WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE
4070      * OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT,
4071      * WHILE THE CURRENT DRIVE IS PERFORMING THE OVERWRITES.
4072      * FOR EACH TRACK (0,1,2), SCORES ARE AVERAGED FOR EACH OFFSET
4073      * DIRECTION, OVER ALL CYLINDERS TESTED.
4074      *
4075      * *****
4076      * ST6: SCOPE
4077      * MOV #6, $TESTN ; SET TEST NUMBER IN APT MAIL BOX
4078      * JSR PC, SETUP ; SET UP FOR LOOP ON ERROR
4079      * JSR PC, INITSS ; INITIALIZE DRIVER PARAMS AND SUB-SYS
4080      * JSR PC, PREPKB ; PREPARE FOR POSSIBLE KBD INPUT
4081      * MOV #1, TSTING ; ALLOW TTY ESCAPE
4082      * CLRB DIF100 ; CLEAR "CYL DIF = 100/200" FLAG
4083      * ; OVERWRITE THE APPROPRIATE CYLS FOR THIS DRIVE
4084      * MOV P, CMND(R5) ; SET SEEK COMMAND
4085      * CLRB P, CYLN(R5) ; SET CYL = 0
4086      * JSR PC, DRVCAL ; SEEK TO CYL 0
4087      * MOV #7, R3 ; SET COUNTER = 7
4088      * MOV #WADATA, P, CMND(R5) ; SET WRITE DATA COMMAND
4089      * MOV LOGDRV, R0 ; GET LOGICAL DRIVE NUMBER
4090      * MOV R0, P, CYLN(R5) ; SET STARTING OVERWRITE CYL FOR THIS DRIVE
4091      * ASL R0 ; GET DATA PATTERN INDEX
4092      * ADD #TABLEG, R0 ; GET PATTERN WORD ADRS
4093      * MOV (R0), RWBUF ; PUT DATA WORD INTO BUFFER
4094      * MOV #RWBUF, P, BALO(R5) ; SET BUS ADDRESS
4095      * BIS #DTBAIL, P, PRST(R5) ; SET BUS ADRS INCREMENT INHIBIT
4096      * MOV #-16896, P, WC(R5) ; SET 22-SECTOR WORD COUNT
4097      * TSTB FORMAT ; DETERMINE THE FORMAT
4098      * BEQ 6$ ; BR IF 22 SECTORS
4099      * MOV #-15360, P, WC(R5) ; SET 20-SECTOR WORD COUNT
4100      * JSR PC, SVPRMS ; SAVE PARAMETERS OF TRANSFER
4101      * JSR PC, TRANSFR ; WRITE THE DATA
4102      * ADD HOLD3, P, CYLN(R5) ; INCR THE CYL NO.
4103      * DEC R3 ; SEE IF DONE WITH ALL CYLS YET
4104      * BNE 6$ ; BR IF NOT YET
4105      * ; READ OVERWRITE CYLS WITH RANGE OF OFFSETS
4106      * MOV #1, TSTTYP ; SET INDICATOR FOR OVRWRT TEST
4107      * MOV #NOSCR0, R0 ; GET STARTING ADRS OF OVRWRT SCORES
4108      * CLR (R0)+ ; INIT SCORE TO 0
4109      * CMP R0, #NOSCR0+192. ; SEE IF ALL SCORES CLEARED YET
4110      * BLO 10$ ; BR IF NOT DONE YET
4111      * MOV #7, R3 ; SET CYL CNTR = 7
4112      * MOV LOGDRV, R0 ; GET LOGICAL DRIVE NO.

```

G07

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 85
CZR6Q8.P11 02-DEC-77 10:46 T6 OVERWRITE TEST

SEQ 0084

4113	015426	010065	000002		MOV	RO,P.CYLN(R5)	;GET START OVRWRT CYL FOR THIS DRIVE
4114	015432	004737	022642		11\$: JSR	PC,INTBLD	;INIT TABLED
4115	015436	004737	017300		12\$: JSR	PC,REDOFS	;DO READS WITH OFFSETS ON THIS CYL
4116	015442	004737	017644		JSR	PC,ADSCOR	;ADD SCRATCH SCORES TO SUM
4117	015446	063765	024412	000002	ADD	HOLD3,P.CYLN(R5)	;INCR CYL NO.
4118	015454	112737	000001	003147	MOVB	#1,DIF100	;SET "CYL DIF = 100/200" FLAG
4119	015462	005303			DEC	R3	;DECR CYL CNTR
4120	015464	004737	023022		JSR	PC,ROTBLO	;ROTATE TABLED FOR NEXT CURRENT ZONE
4121	015470	020327	000001		CMP	R3,#1	;SEE IF ON LAST ZONE YET
4122	015474	003360			BGT	12\$	;BR IF NOT YET
4123	015476	001755			BEQ	11\$	;BR IF ON LAST CYL
4124	015500	004737	020024		JSR	PC,AVSCOR	;COMPUTE AVERAGE OVERWRITE SCORES
4125	015504	105037	003143		CLRB	TS1TYP	;CLEAR OVRWRT TEST INDICATOR

TEST 7 DRIVE SELF-TEST

* THE PROGRAM EVALUATES THE DRIVE'S ABILITY TO WRITE AND READ
* ITS OWN DATA, AT VARIOUS ANGULAR POSITIONS ON THE PACK. FIRST,
* SECTORS 4,11,16, AND 23 OF THE APPROPRIATE CYLINDERS SHOWN
* IN TABLE FOR THIS DRIVE ARE WRITTEN WITH THE DATA PATTERN
* SHOWN IN TABLE, FOR ALL SURFACES. THEN, THE SECTORS ARE READ
* WITH THE FOLLOWING OFFSETS APPLIED : 100,200,300,400,500,600
* 700,800,900,1000,1100,1200 MICRO-IN. IN (+) AND (-) DIRECTIONS.
* THE PROGRAM SCANS FOR DATA CHECK ERRORS DURING EACH READ, AND IT
* COMPUTES A SCORE WHICH IS PROPORTIONAL TO THE FAILING OFFSET
* MAGNITUDE. THEN, THE SCORES FOR ALL SECTORS READ ARE AVERAGED,
* TO COME UP WITH A DRIVE SELF-TEST SCORE FOR EACH OFFSET
* DIRECTION, FOR EACH SURFACE.
* THIS SCORE IS SAVED FOR LATER USE, TO BECOME THE STANDARD
* FOR THE COMPATIBILITY DATA READS WHICH ARE TO FOLLOW.

ST7: SCOPE

4149	015510	000004			MOV	#7,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
4149	015512	012737	000007	001334	JSR	PC,SETUP	;SET UP FOR LOOP ON ERROR
4150	015520	004737	022042		JSR	PC,INITSS	;INITIALIZE DRIVER PARAMS AND SUB-SYS
4151	015524	004737	021170		JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
4152	015530	004737	020774		CLR	NSSCR0	;INIT SELF-TEST SCORES TO 0
4153	015534	005037	007412		CLR	NSSCR1	
4154	015540	005037	007414		CLR	NSSCR2	
4155	015544	005037	007416		CLR	PSSCR0	
4156	015550	005037	007420		CLR	PSSCR1	
4157	015554	005037	007422		CLR	PSSCR2	
4158	015560	005037	007424		CLRB	DIF100	;CLEAR "CYL DIF = 100/200" FLAG
4159	015564	105037	003147		MOV	#6,R3	;SET CYL COUNTER = 6
4160	015570	012703	000006		MOVB	LOGDRV,RO	;GET LOGICAL DRIVE NO.
4161	015574	113700	003154		ADD	#60,RO	;COMPUTE STARTING CYL NO.
4162	015600	062700	000060		MOV	#RWBUF,P.BALO(R5)	;SET BUS ADRS
4163	015604	012765	053342	000010	JSR	PC,LODSEC	;LOAD R/W BUF WITH DATA PATTERN
4164	015612	004737	024502		MOV	#-256,P.WC(R5)	;SET WORD COUNT FOR 1 SECTOR
4165	015616	012765	177400	000012	MOV	#4,P.SECT(R5)	;SET SECTOR = 4
4166	015624	112765	000004	000004	3\$: CLRB	P.TRACK(R5)	;SET TRACK = 0
4167	015632	105065	000005		4\$: CLRB	NSSCR0	;INIT SCRATCH SCORES TO 0
4168	015636	105037	007432				

H07

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 86
CZR6QB.P11 02-DEC-77 10:46 T7 DRIVE SELF-TEST

SEQ 0085

4169	015642	105037	007452		CLRB	NSCOR1	
4170	015646	105037	007472		CLRB	NSCOR2	
4171	015652	105037	007512		CLRB	PSCOR0	
4172	015656	105037	007532		CLRB	PSCOR1	
4173	015662	105037	007552		CLRB	PSCOR2	
4174	015666	105037	003143		CLRB	TSTTYP	;CLEAR OFFSET TEST INDICATOR
4175	015672	005065	000002		CLR	P.CYLN(R5)	;SET CYL = 0
4176	015676	112765	000117	000001	MOVB	#SEEK,P.CMND(R5)	;SET SEEK COMMAND
4177	015704	004737	030146		JSR	PC,DRVCAL	;PERFORM SEEK TO 0
4178	015710	010065	000002		MOV	RD,P.CYLN(R5)	;SET CURRENT CYLINDER NUMBER
4179	015714	112765	000123	000001	MOVB	#WADATA,P.CMND(R5)	;SET WRITE DATA COMMAND
4180	015722	004737	030146		JSR	PC,DRVCAL	;WRITE A SECTOR
4181	015726	112737	000003	003143	MOVB	#3,TSTTYP	;SET INDICATOR FOR SELF-TEST
4182	015734	004737	017300		JSR	PC,REDOFS	;READ SECTOR WITH RANGE OF OFFSETS
4183	015740	105265	000005		INCB	P.TRCK(R5)	;INCREMENT TRACK
4184	015744	126527	000005	000002	CMPB	P.TRCK(R5),#2	;SEE IF ALL TRACKS DONE ON THIS CYL
4185	015752	101745			BLOS	6\$	;BR IF NOT YET
4186	015754	004737	017644		JSR	PC,ADSCOR	;ADD SCRATCH SCORES TO SUMS
4187	015760	062765	000005	000004	ADD	#5,P.SECT(R5)	;INCREMENT SECTOR BY 5
4188	015766	126527	000004	000023	CMPB	P.SECT(R5),#23	;SEE IF ALL SECTORS DONE ON THIS TRACK
4189	015774	101716			BLOS	4\$	;BR IF NOT YET
4190	015776	063700	024412		ADD	HOLD3,R0	;INCR CYL NO. BY 100/200
4191	016002	005303			DEC	R3	;DECR CYL CNTR
4192	016004	001307			BNE	3\$	;BR IF NOT DONE WITH ALL CYLS YET
4193	016006	004737	020024		JSR	PC,AVSCOR	;COMPUTE AVG SELF-TEST SCORES
4194	016012	105037	003143		CLRB	TSTTYP	;CLEAR OVRWRT TEST INDICATOR

```

*****
*TEST 10      COMPATIBILITY DATA READ TEST
*
*   HAVING ESTABLISHED A SELF-TEST SCORE FOR THIS DRIVE, THE
*   PROGRAM PROCEEDS TO PERFORM THE COMPATIBILITY DATA READS
*   OF THE PATTERNS WRITTEN BY ALL THE DRIVES IN EACH CYLINDER
*   BLOCK (ON EACH SURFACE). EACH COMPATIBILITY CYLINDER BLOCK
*   SHOWN IN TABLE C IS READ WITH THE FOLLOWING OFFSET VALUES :
*   100,200,300,400,500,600,700,800,900,1000,1100,1200 MICRO-IN.,
*   IN (+) AND (-) DIRECTIONS. THE PROGRAM SCANS FOR READ ERRORS
*   DURING EACH READ, AND IF ONE OCCURS, THE PROGRAM DETERMINES
*   WHICH DRIVE'S DATA WAS BEING READ AT THAT INSTANT AND A SCORE
*   FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED
*   AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE
*   DONE WITH ALL OF THE OFFSETS APPLIED, AND A SEPARATE SCORE
*   FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS READING
*   THE COMPATIBILITY DATA. THEN, EACH SCORE IS APPROPRIATELY
*   ADJUSTED TO REFLECT THE SELF-TEST SCORE FOR THE CURRENT DRIVE
*   ON THAT SURFACE. THE SCORES ARE THEN AVERAGED OVER ALL CYLINDER
*   BLOCKS. EACH SCORE IS PROPORTIONAL TO THE CAPABILITY OF THE
*   CURRENT DRIVE TO SUCCESSFULLY READ THE DATA WRITTEN BY ONE OF
*   THE OTHER DRIVES, AND SCORES ARE COMPUTED SEPARATELY FOR EACH
*   SURFACE (TRACK). THUS, THE SCORES REVEAL WHICH DRIVES ARE
*   INVOLVED IN A SITUATION IN WHICH A PARTICULAR DRIVE HAS
*   DIFFICULTY IN READING THE DATA OF ONE OR SEVERAL OTHER DRIVES.
*****

```

```

4225 016016 000004 TST10: SCOPE
4226 016020 012737 000010 MOV #10,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
4227 016026 004737 022042 JSR PC,$SETUP ;SET UP FOR LOOP ON ERROR
4228 016032 004737 021170 JSR PC,$INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4229 016036 004737 020774 JSR PC,$PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
4230 016042 112737 000002 003143 MOV #2,$TSTTYP ;SET INDIC FOR READ TEST
4231 016050 105037 003147 CLRB DIF100 ;CLEAR "CYL DIF = 100/200" FLAG
4232 016054 012700 007112 MOV #NCSRO,R0 ;GET ADRS OF COMPAT READ SCORES
4233 016060 005020 6$: CLR (R0)+ ;INIT A SCORE TO 0
4234 016062 020027 007412 CMP R0,#NCSRO+192. ;SEE IF ALL SCORES CLEARED YET
4235 016066 103774 BLO 6$ ;BR IF NOT YET
4236 016070 012703 000006 MOV #6,R3 ;SET CYL BLK CNTR = 6
4237 016074 004737 022642 JSR PC,$INTBLD ;INIT TABLE D
4238 016100 013701 024414 MOV #HOLD4,R1 ;SET START CYL = 60/160
4239 016104 012765 053342 000010 MOV #RWBUF,P,BALO(R5) ;SET BUS ADRS
4240 016112 052765 100000 000014 BIS #DTBA11,P,PRST(R5) ;SET BUS ADRS INCR INHIBIT
4241 016120 012765 137000 000012 MOV #-16896.,P,WC(R5) ;SET 22-SECTOR WORD COUNT
4242 016126 105737 003134 TSTB FORMAT ;TEST THE FORMAT
4243 016132 001403 BEQ 10$ ;BR IF 22 SECTORS
4244 016134 012765 142000 000012 MOV #-15360.,P,WC(R5) ;SET 20-SECTOR WORD COUNT
4245 016142 012700 000020 10$: MOV #16.,R0 ;INIT CYL CNTR = 16.
4246 016146 010165 000002 MOV R1,P,CYLN(R5) ;SET CYL NO.
4247 016152 004737 017300 12$: JSR PC,$REDOFS ;DO READS WITH OFSTS ON THIS CYL
4248 016156 004737 017644 JSR PC,$ADSCOR ;ADD SCRATCH SCORES TO SUM
4249 016162 005265 000002 INC P,CYLN(R5) ;INCR CYL NO.
4250 016166 005300 DEC R0 ;SEE IF DONE WITH THIS CYL BLK
4251 016170 001370 BNE 12$ ;BR IF NOT YET
4252 016172 004737 023022 JSR PC,$ROTBLO ;ROTATE TABLED FOR NEXT CURRENT ZONE
4253 016176 063701 024412 ADD #HOLD3,R1 ;INCR CYL BY 100/200
4254 016202 112737 000001 003147 MOV #1,DIF100 ;SET "CYL DIF = 100" FLAG
4255 016210 005303 DEC R3 ;SEE IF DONE WITH ALL CYL BLKS
4256 016212 001353 BNE 10$ ;BR IF NOT YET
4257 016214 004737 020024 JSR PC,$AVSCOR ;COMPUTE AVERAGES OF READ SCORES
4258 016220 105037 003143 CLRB $TSTTYP ;CLEAR OFST TEST INDICATOR
4259 ;ADJUST COMPAT READ SCORES (- OFSTS) BY SELF-TEST SCORES
4260 016224 012701 007112 MOV #NCSRO,R1 ;SET COMPAT SCORE POINTER
4261 016230 012703 007412 MOV #NSSRO,R3 ;SET SELF-TEST SCORE POINTER
4262 016234 012700 005512 14$: MOV #DRVLST,R0 ;SET DRIVE LIST POINTER
4263 016240 012737 000014 001304 MOV #12.,$TMP11 ;COMPUTE DIFFERENCE OF 12 (DEC) AND
4264 ; SELF-TEST SCORE FOR THIS TRACK
4265 016246 162337 001304 SUB (R3)+,$TMP11
4266 016252 010104 MOV R1,R4 ;GET READ SCORE POINTER IN R4
4267 016254 005720 15$: TST (R0)+ ;SEE IF A DRIVE LISTED HERE
4268 016256 001417 BEQ 18$ ;BR IF NOT
4269 016260 013737 001304 001306 MOV $TMP11,$TMP12 ;PUT DIFFERENCE INTO $TMP12
4270 016266 061437 001306 ADD (R4),$TMP12 ;ADD DIFF TO READ SCORE
4271 016272 023727 001306 000014 CMP $TMP12,#12. ;SEE IF 12 EXCEEDED
4272 016300 101403 BLOS 16$ ;BR IF NOT
4273 016302 012737 000014 001306 MOV #12,$TMP12 ;SET SCORE = 12.
4274 016310 013724 001306 16$: MOV $TMP12,(R4)+ ;PUT BACK ADJUSTED SCORE
4275 016314 000757 BR 15$ ;BR TO CONTINUE
4276 016316 062701 000040 18$: ADD #32.,R1 ;POINT TO SCORES FOR NEXT TRACK
4277 016322 020127 007212 CMP R1,#NCSRO+64. ;SEE IF SCORE TABLE EXCEEDED
4278 016326 101742 BLOS 14$ ;BR IF NOT YET
4279 ;ADJUST COMPAT READ SCORES (+ OFSTS) BY SELF-TEST SCORES
4280 016330 012701 007252 MOV #PCSCRO,R1 ;SET COMPAT SCORE POINTER

```

```

4281 016334 012703 007420      MOV      #PSSCR0,R3      ;SET SELF-TEST SCORE POINTER
4282 016340 012700 005512      MOV      #DRVLST,R0     ;SET DRIVE LIST POINTER
4283 016344 012737 000014 001304  MOV      #12, $TMP11    ;COMPUTE DIFFERENCE OF 12 (DEC) AND
4284                                     ;SELF-TEST SCORE FOR THIS TRACK
4285 016352 162337 001304      SUB      (R3)+, $TMP11
4286 016356 010104      MOV      R1, R4      ;GET READ SCORE POINTER IN R4
4287 016360 005720      TST      (R0),+      ;SEE IF A DRIVE LISTED HERE
4288 016362 001417      BEQ      28$      ;BR IF NOT
4289 016364 013737 001304 001306  MOV      $TMP11, $TMP12    ;PUT DIFFERENCE INTO $TMP12
4290 016372 061437 001306      ADD      (R4), $TMP12    ;ADD DIFF TO READ SCORE
4291 016376 023727 001306 000014  CMP      $TMP12, #12.    ;SEE IF 12 EXCEEDED
4292 016404 101403      BLOS     26$      ;BR IF NOT
4293 016406 012737 000014 001306  MOV      #12, $TMP12    ;SET SCORE = 12.
4294 016414 013724 001306      MOV      $TMP12, (R4)+    ;PUT BACK ADJUSTED SCORE
4295 016420 000757      BR       25$      ;BR TO CONTINUE
4296 016422 062701 000040      ADD      #32, R1      ;POINT TO SCORES FOR NEXT TRACK
4297 016426 020127 007352      CMP      R1, #PCSCRO+64. ;SEE IF SCORE TABLE EXCEEDED
4298 016432 101742      BLOS     24$      ;BR IF NOT YET
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321

```

```

*****
;TEST 11      TYPE TEST SCORES, DISMOUNT PACK, IN PASS 2
;
;THE OVERWRITE AND COMPATIBILITY DATA READ TEST SCORES FOR THIS
;DRIVE ARE CONVERTED AND TYPED. THEN, THE OPERATOR IS
;DIRECTED TO UNLOAD THE DRIVE CURRENTLY UNDER TEST, AND TO REMOVE
;THE TEST PACK. THEN, THE PROGRAM DOES 1 OF 4 THINGS :
;(1) IF THERE IS ANOTHER DRIVE TO TEST ON THIS SUBSYS, THE
;PROGRAM JUMPS TO TEST 5 FOR THE NEXT DRIVE.
;(2) IF THERE IS ONLY ONE SUBSYS (FROM 200 START), AND IF
;PASS 2 HAS BEEN COMPLETED ON ALL DRIVES, THE ENTIRE TESTING AND
;REPORTING HAVE BEEN COMPLETED, AND THE PROGRAM TYPES A
;COMPLETION MESSAGE, AND THEN IT RESTARTS AT ADRS 200.
;(3) IF THERE IS ANOTHER SUBSYS TO PERFORM PASS 2 ON, THE
;PROGRAM TELLS THE OPERATOR TO RESTART AT ADDRESS 220, AND THEN
;IT HALTS.
;(4) IF THERE ARE NO MORE DRIVES TO TEST ON ANY OTHER SUBSYS, THE
;PROGRAM TYPES A COMPLETION MESSAGE AND THEN IT HALTS.
*****

```

```

4322 016434 000004      ;ST11: SCOPE
4323 016436 012737 000011 001334  MOV      #11, $TESTN    ;SET TEST NUMBER IN APT MAIL BOX
4324 016444 004737 022042      JSR      PC, SETUP    ;SET UP FOR LOOP ON ERROR
4325 016450 004737 021170      JSR      PC, INITSS   ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4326 016454 004737 020774      JSR      PC, PREPKB   ;PREPARE FOR POSSIBLE KBD INPUT
4327 016460 112737 000001 003127  MOVB     #1, TSTING    ;ALLOW TTY ESCAPE
4328                                     ;CONVERT AND TYPE TEST SCORES
4329 016466 013700 005554      MOV      DRVPTR, R0    ;GET DRIVE LIST POINTER
4330 016472 112037 010743      MOVB     (R0)+, RESLTS+20. ;GET NO. OF THIS DRIVE
4331 016476 111037 010742      MOVB     (R0), RESLTS+19.
4332 016502 104401 010717      TYPE      RESLTS      ;TYPE TEST RESULT HEADINGS
4333 016506 012737 006612 001306  MOV      #NOSCR0, $TMP12    ;GET POINTER TO OVRWRT SCORES
4334 016514 012700 000060      MOV      #0, R0      ;INIT TRACK NO. TO 0
4335 016520 012701 005512      MOV      #DRVLST, R1    ;GET DRIVE LIST POINTER
4336 016524 110037 011225      MOVB     R0, TRKBUFF+1 ;SET TRACK NO. FOR PRINTOUT

```

K07

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 89
T11 TYPE TEST SCORES, DISMOUNT PACK, IN PASS 2

SEQ 0088

4337	016530	013704	001306		MOV	\$TMP12,R4	;GET SCORE POINTER	
4338	016534	004737	025564	6\$:	JSR	PC,CTLOUT	;CHECK FOR TTY ESCAPE RQST	
4339	016540	104401	011224		TYPE	TRKBUF	;TYPE TRACK (HEAD) NUMBER	
4340	016544	012103			MOV	(R1)+,R3	;GET A DRIVE NAME	
4341	016546	020377	167002		CMP	R3,@DRVPT	;SEE IF IT IS CURRENT DRIVE	
4342	016552	001003			BNE	8\$	;BR IF NOT	
4343	016554	104401	011235		TYPE	SELF	;TYPE "SELF"	
4344	016560	000407			BR	10\$	;CONTINUE	
4345	016562	110337	011232	8\$:	MOVB	R3,DRVBUF+2	;GET DRIVE NO.	
4346	016566	000303			SWAB	R3		
4347	016570	110337	011231		MOVB	R3,DRVBUF+1	;GET SUBSYS NAME	
4348	016574	104401	011230		TYPE	DRVBUF	;TYPE NO. OF DRIVE READ	
4349	016600	005037	005212	10\$:	CLR	BUFF0	;CLEAR TTY CHAR BUF	
4350	016604	005037	005214		CLR	BUFF0+2		
4351	016610	010437	001310		MOV	R4,\$TMP13	;GET SCORE POINTER	
4352	016614	012437	005212		MOV	(R4)+,BUFF0	;PUT AN OVRWRT SCORE INTO BUF	
4353	016620	012737	000004	001304	MOV	#4,\$TMP11	;SET COUNTER = 4	
4354	016626	023727	005212	000012	11\$:	CMP	BUFF0,#10.	;SEE IF SCORE < 10.
4355	016634	103002			BHIS	12\$	;BR IF NOT	
4356	016636	104401	012150		TYPE	SPACE1	;TYPE 1 SPACE	
4357	016642	023727	005212	000007	12\$:	CMP	BUFF0,#7	;SEE IF SCORE > 7
4358	016650	101003			BHI	14\$	;BR IF YES	
4359	016652	104401	012170		TYPE	PROMPT	;TYPE "*" "	
4360	016656	000402			BR	16\$	;CONTINUE	
4361	016660	104401	012147	14\$:	TYPE	SPACE2	;TYPE 2 SPACES	
4362	016664	012746	005212	16\$:	MOV	BUFF0,-(SP)	;GET BUF ADRS ON STACK	
4363	016670	004737	043174		JSR	PC,@\$0B2D	;CONVERT SCORE TO DEC.	
4364	016674	004737	043370		JSR	PC,@\$SUPRS	;TYPE IT	
4365	016700	104401	011243		TYPE	TAB	;TYPE A TAB	
4366	016704	062737	000140	001310	ADD	#96,\$TMP13		
4367	016712	017737	162372	005212	MOV	@\$TMP13,BUFF0	;PUT NEXT SCORE INTO BUFFER	
4368	016720	005337	001304		DEC	\$TMP11	;DECREMENT SCORE COUNTER	
4369	016724	001340			BNE	11\$	;BR IF READ SCORE SHOULD BE TYPED	
4370	016726	104401	001325		TYPE	\$CRLF	;TYPE <CR>, <LF>	
4371	016732	005711			TST	(R1)	;SEE IF ALL SCORES TYPED FOR THIS TRACK	
4372	016734	001277			BNE	6\$	;BR IF NOT	
4373	016736	104401	001325		TYPE	\$CRLF	;TYPE <CR>, <LF>	
4374	016742	005200			INC	R0	;INCR TRACK NO.	
4375	016744	062737	000040	001306	ADD	#32,\$TMP12	;INCREMENT SCORE POINTER FOR NEXT TRACK	
4376	016752	020027	000062		CMP	R0,#12	;SEE IF ALL TRACKS TYPED YET	
4377	016756	101660			BLOS	5\$	;BR IF NOT YET	
4378					; COMPUTE, CONVERT, AND TYPE TEST AVERAGES (OVER ALL SURFACES)			
4379	016760	004737	020262		JSR	PC,TRKAVG	;GET AVGS OVER ALL TRKS	
4380	016764	013700	005554		MOV	DRVPT,R0	;PUT DRIVE NAME INTO BUF	
4381	016770	112037	011312		MOVB	(R0)+,BADDRV+7		
4382	016774	111037	011311		MOVB	(R0),BADDRV+6		
4383	017000	005037	001304		CLR	\$TMP11	;CLEAR SCORE INDICATOR	
4384	017004	013737	007426	005212	MOV	OVRVAVG,BUFF0	;GET OVRWRT AVG SCORE	
4385	017012	012737	011132	017026	MOV	#OVRVAVG,22\$		
4386	017020	104401	011303	20\$:	TYPE	,BADDRV	;TYPE "DRIVE XX"	
4387	017024	104401			TYPE		;TYPE WHICH KIND OF AVERAGE	
4388	017026	000000		22\$:	.WORD	0		
4389	017030	012746	005212		MOV	#BUFF0,-(SP)	;PUT POINTER TO AVG ON STACK	
4390	017034	004737	043174		JSR	PC,@\$0B2D	;CONVERT TO DECIMAL	
4391	017040	004737	043370		JSR	PC,@\$SUPRS	;TYPE IT	
4392	017044	104401	001325		TYPE	, \$CRLF	;TYPE <CR>, <LF>	

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6GB.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 90
T11 TYPE TEST SCORES, DISMOUNT PACK, IN PASS 2

SEQ 0089

```

4393 017050 013737 007430 005212      MOV      COMAVG,BUFFD      ;GET READ AVG SCORE
4394 017056 012737 011151 017026      MOV      #RAVERG,22$
4395 017064 005137 001304              COM      $TMP11          ;COMPLEMENT SCORE INDIC
4396 017070 001353              BNE      20$            ;BR TO TYPE READ SCORE
4397 017072 104401 001325              TYPE     $SCRLF          ;TYPE <CR>,<LF>
4398                                ;DISMOUNT PACK ON THIS DRIVE
4399 017076 105037 003127      CLR      TSTING          ;DON'T ALLOW TTY ESCAPE
4400 017102 004737 022472      JSR      PC,DMCART        ;DIRECT OPERATOR TO DISMOUNT PACK
4401 017106 062737 000002 005554      ADD      #2,DRVPT        ;INCREMENT DRIVE LIST POINTER
4402 017114 013700 005554      MOV      DRVPT,R0
4403 017120 005710              TST      (R0)
4404 017122 001421              BEQ      28$            ;SEE IF ALL DONE WITH PASS 2 YET
4405 017124 126037 000001 005510      CMP      1(R0),SUBSYS    ;BR IF DONE WITH PASS 2
4406 017132 001005              BNE      24$            ;SEE IF DONE WITH THIS SUBSYS YET
4407 017134 112737 000004 001102      MOV      #4,$TSTNM       ;BR IF DONE WITH THIS SUBSYS
4408 017142 000137 015146      JMP      TST5            ;SET TEST NO. = 4
4409 017146 116037 000001 010574 24$:  MOV      1(R0),STR220+39. ;DO PASS 2 ON NEXT DRIVE OF SUBSYS
4410 017154 104401 010525              TYPE     ,STR220          ;PUT SUBSYS NAME INTO MSG
4411 017160 000000              26$:  HALT
4412 017162 000137 000220              JMP      220            ;TYPE RESTART MSG FOR PASS 2
4413 017166 104401 011170              TYPE     ,ENDTST         ;HALT THE PROCESSOR SO THAT OPERATOR
4414 017172 012737 005512 005554 28$:  MOV      #DRVLT,DRVPT    ;CAN RESTART AT PROPER ADRS
4415 017200 105737 003126              TST      MDFLAG          ;TYPE "*** END OF TESTING ***"
4416 017204 001002              BNE      30$            ;RE-INIT DRIVE LIST POINTER
4417 017206 000137 000200              JMP      200            ;SEE IF 200 START
4418 017212 000000              30$:  HALT          ;BR IF NOT
4419 017214 000137 000204              JMP      204            ;RESTART AT ADRS 200
4420                                ;RESTART AT ADRS 204

```

.SBTTL END OF PASS ROUTINE

```

;*****
;INCREMENT THE PASS NUMBER ($PASS)
;IF THERES A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO 200

```

```

4430 017220              $EOP:
4431 017220 000004              SCOPE
4432 017222 005037 001102      CLR      $TSTNM          ;;ZERO THE TEST NUMBER
4433 017226 005237 001336      INC      $PASS          ;;INCREMENT THE PASS NUMBER
4434 017232 042737 100000 001336      BIC      #100000,$PASS    ;;DON'T ALLOW A NEG. NUMBER
4435 017240 005327              DEC      (PC)+          ;;LOOP?
4436 017242 000001              $EOPCT: .WORD 1
4437 017244 003013              BGT      $DOAGN          ;;YES
4438 017246 012737              MOV      (PC)+,2(PC)+      ;;RESTORE COUNTER
4439 017250 000001              $ENDCT: .WORD 1
4440 017252 017242              $EOPCT
4441 017254 013700 000042      $GET42: MOV      2#42,R0          ;;GET MONITOR ADDRESS
4442 017260 001405              BEQ      $DOAGN          ;;BRANCH IF NO MONITOR
4443 017262 000005              RESET          ;;CLEAR THE WORLD
4444 017264 004710              $ENDAD: JSR      PC,(R0)          ;;GO TO MONITOR
4445 017266 000240              NOP
4446 017270 000240              NOP          ;;SAVE ROOM
4447 017272 000240              NOP          ;;FOR
4448 017274              $DOAGN:          ;;ACT11

```

```

4449 017274 000137 000200          JMP      @#200          ;; RETURN
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462 017300 104407
4463
4464 017302 122737 000003 003143
4465 017310 001406
4466 017312 012700 007432
4467 017316 105020
4468 017320 020027 007572
4469 017324 103774
4470
4471 017326 112765 000117 000001
4472 017334 105737 003147
4473 017340 001412
4474 017342 163765 024416 000002
4475 017350 004737 030146
4476 017354 063765 024416 000002
4477 017362 105037 003147
4478 017366 004737 030146
4479 017372 105037 003156
4480 017376 012703 000014
4481 017402 012700 000204
4482 017406 105037 003157
4483 017412 122737 000003 003143
4484 017420 001020
4485 017422 116501 000005
4486 017426 006301
4487 017430 006301
4488 017432 006301
4489 017434 006301
4490 017436 062701 007432
4491 017442 105737 003156
4492 017446 001402
4493 017450 062701 000060
4494 017454 012704 000001
4495 017460 000411
4496 017462 012701 007432
4497 017466 012704 000060
4498 017472 105737 003156
4499 017476 001402
4500 017500 012701 007512
4501 017504 122137 003157
4502 017510 001002
4503 017512 105261 177777
4504 017516 005304

;*****
;REDOFS - READ DATA WITH RANGE OF OFFSETS
;THIS SUBROUTINE PERFORMS READS WITH OFFSETS LISTED IN OFSTBL.
;THE OFFSETS ARE APPLIED MINIMUM TO MAXIMUM, IN BOTH DIRECTIONS,
;UNTIL THERE IS NO FAILURE. EACH TIME A READ ERROR OCCURS,
;THE APPROPRIATE SCORE FOR THE DRIVE INVOLVED IS INCREMENTED NO FURTHER
;(IN THE SCRATCH SCORE TABLE). EACH READ IS DONE TO A PACK
;AREA, WHOSE ADDRESS IS IN THE PARAMETER BLOCK ON ENTRY.
;*****
REDOFS: SAVREG          ;SAVE RD-RS
;INITIALIZE SCRATCH SCORES
CMPB      #3,TSTTYP      ;SEE IF DRIVE SELF TEST
BEQ       5$             ;BR IF YES
MOV       #NSCORD,R0      ;GET ADRS OF SCRATCH SCORE TABLE
4$: CLRB      (R0)+        ;INIT A SCORE TO 0
CMP       R0,#NSCORD+96.  ;SEE IF DONE YET
BLO       4$             ;BR IF NOT YET
;PERFORM READS WITH OFFSET
5$: MOVB     #SEEK,P.CMND(R5) ;SET SEEK COMMAND
TSTB     DIF100          ;SEE IF CYL DIF = 100/200 (OCT)
BEQ       9$             ;BR IF NOT
SUB       HOLD5,P.CYLN(R5) ;SEEK HALF THE CYLS
JSR       PC,DRVCAL
ADD       HOLD5,P.CYLN(R5)
CLRB     DIF100          ;CLEAR "CYL DIF = 100/200 FLAG"
9$: JSR       PC,DRVCAL    ;PERFORM SEEK
CLRB     OFSDIR          ;CLEAR OFFSET DIRECTION FLAG
MOV       #12,R3         ;SET OFFSET COUNTER FOR NEG OFSTS
MOV       #204,R0        ;INIT NEG OFST VALUE TO 100 UIN
12$: CLRB     OFSCNT       ;INIT OFFSET NUMBER
6$: CMPB     #3,TSTTYP    ;SEE IF SELF-TEST
BNE       15$            ;BR IF NOT
MOVB     P.TRCK(R5),R1    ;GET TRACK NUMBER
ASL       R1             ;MULT BY 16.
ASL       R1
ASL       R1
ASL       R1
ADD       #NSCORD,R1      ;GET POINTER TO SCORES
TSTB     OFSDIR          ;SEE WHICH OFST DIRECTION
BEQ       17$            ;BR IF NEGATIVE
ADD       #48,R1         ;POINT TO POSITIVE SCORES
17$: MOV      #1,R4        ;SET COUNTER TO 1
BR        14$            ;CONTINUE
15$: MOV      #NSCORD,R1  ;GET POINTER TO SCORES
MOV       #48,R4         ;SET COUNTER TO 48.
TSTB     OFSDIR          ;SEE WHICH OFST DIRECTION
BEQ       14$            ;BR IF NEGATIVE
MOV       #PSCORD,R1     ;GET POINTER FOR POS SCORES
14$: CMPB     (R1)+,OFSCNT ;SEE IF THIS IS A PERFECT SCORE YET
BNE       18$            ;BR IF NOT
INCB     -1(R1)          ;INCREMENT THIS SCORE
18$: DEC      R4          ;DECREMENT COUNTER

```

```

4505 017520 001371      BNE      14$      ;BR IF NOT DONE YET
4506 017522 110065 000006 16$: MOVB   RO,P.OFST(R5) ;GET AN OFFSET VALUE
4507 017526 112765 000115 000001 MOVB   #OFFSET,P.CMND(R5) ;SET OFFSET CMND
4508 017534 004737 030146 JSR     PC,DRVCAL      ;PERFORM OFFSET
4509 017540 105237 003157 INCB    OFSCNT         ;INCR OFFSET NUMBER
4510 017544 112765 000121 000001 MOVB   #RODATA,P.CMND(R5) ;SET READ COMMAND
4511 017552 004737 027500 JSR     PC,SVPRMS      ;SAVE PARAMETERS OF TRANSFER
4512 017556 004737 027570 JSR     PC,TRNSFR      ;DO XFER, HANDLE DCK ERRORS
4513 017562 112765 000177 000001 MOVB   #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR COMMAND
4514 017570 004737 030146 JSR     PC,DRVCAL      ;DO A SUBSYSTEM CLEAR
4515 017574 105065 000006 CLRB    P.OFST(R5)     ;CLEAR OFFSET VALUE
4516 017600 112765 000115 000001 MOVB   #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
4517 017606 004737 030146 JSR     PC,DRVCAL      ;PERFORM OFFSET TO 0
4518 017612 062700 000004 ADD     #4,RO          ;INCR OFST VALUE BY 100 UIN
4519 017616 005303 DEC     R3             ;DECR COUNTER
4520 017620 001274 BNE     6$            ;BR IF NOT DONE IN THIS DIRECTION YET
4521 017622 012703 000014 MOV     #12,R3         ;SET COUNTER FOR POS SFSTS
4522 017626 012700 000004 MOV     #4,RO          ;INIT POS OFST TO 100 UIN
4523 017632 105137 003156 COMB    OFSDIR      ;COMPLEMENT OFFSET DIRECTION FLAG
4524 017636 001263 BNE     12$          ;BR IF POS OFSTS NOT DONE YET
4525 017640 104410 RESREG   R5            ;RESTORE RO-R5
4526 017642 000207 RTS      PC          ;RETURN
4527
4528
4529
4530
4531
4532
4533
4534
4535 017644 104407      ;*****
4536 017646 122737 000003 003143 ADSCOR: SAVREG      ;SAVE RO-R5
4537 017654 001031 CMPB    #3,TSTTYP      ;SEE IF SELF-TEST
4538 017656 113700 007432 BNE     2$            ;BR IF NOT
4539 017662 060037 007412 MOVB    NSCOR0,RO      ;ADD SELF-TEST SCORES
4540 017666 113700 007452 ADD     RO,NSSCR0
4541 017672 060037 007414 MOVB    NSCOR1,RO
4542 017676 113700 007472 ADD     RO,NSSCR1
4543 017702 060037 007416 MOVB    NSCOR2,RO
4544 017706 113700 007512 ADD     RO,NSSCR2
4545 017712 060037 007420 MOVB    PSOR0,RO
4546 017716 113700 007532 ADD     RO,PSSCR0
4547 017722 060037 007422 MOVB    PSOR1,RO
4548 017726 113700 007552 ADD     RO,PSSCR1
4549 017732 060037 007424 MOVB    PSOR2,RO
4550 017736 000430 ADD     RO,PSSCR2
4551 017740 012700 006612 BR      8$            ;EXIT
4552 017744 012705 006752 2$: MOV     #NOSCR0,RO      ;GET POINTER TO (-) OVRWRT SCORES
4553 017750 122737 000001 003143 MOV     #POSCR0,R5      ;GET POINTER TO (+) OVRWRT SCORES
4554 017756 001404 CMPB    #1,TSTTYP      ;SEE IF READING OVRWRT DATA
4555 017760 012700 007112 BEQ     4$            ;BR IF YES
4556 017764 012705 007252 MOV     #NCSCR0,RO      ;GET POINTER TO (-) COMPAT DATA SCORES
4557 017770 012704 000060 MOV     #PCSCR0,R5      ;GET POINTER TO (+) COMPAT DATA SCORES
4558 017774 012701 007432 4$: MOV     #48,R4          ;INIT COUNTER TO 48.
4559 020000 012703 007512 MOV     #NSCOR0,R1      ;GET POINTER TO NEG SCRATCH SCORES
4560 020004 112102 MOV     #PSOR0,R3      ;GET POINTER TO POS SCRATCH SCORES
4561 020008 000000 6$: MOVB    (R1)+,R2      ;GET A SCRATCH NEG SCORE

```

4561	020006	060220			ADD	R2,(R0)+	;ADD IT TO (-) SCORE SUM
4562	020010	112302			MOVB	(R3)+,R2	;GET PNTR TO POS SCRATCH SCORES
4563	020012	060225			ADD	R2,(R5)+	;ADD IT TO (+) SCORE SUM
4564	020014	005304			DEC	R4	;DECREMENT COUNTER
4565	020016	001372			BNE	6\$	;BR IF NOT DONE YET
4566	020020	104410			8\$: RESREG		;RESTORE R0-R5
4567	020022	000207			RTS	PC	;RETURN

4568							
4569							
4570							
4571							
4572							
4573							
4574							
4575							
4576							
4577	020024	104407			AVSCOR: SAVREG		;SAVE R0-R5
4578	020026	122737	000003	003143	CMPB	#3,TSTTYP	;SEE IF SELF-TEST
4579	020034	001014			BNE	2\$	;BR IF NOT
4580	020036	012737	007412	001304	MOV	#NSSCRO,\$TMP11	;GET ADRS OF (-) SELF-TEST SCORES
4581	020044	012737	007420	001310	MOV	#PSSCRO,\$TMP13	;GET ADRS OF (+) SELF-TEST SCORES
4582	020052	012705	000030		MOV	#24,R5	;SET DIVISOR = 24.
4583	020056	012737	000003	001306	MOV	#3,\$TMP12	;SET COUNTER = 3
4584	020064	000427			BR	12\$	;CONTINUE
4585	020066	012737	006612	001304	2\$: MOV	#NOSCRO,\$TMP11	;GET POINTER TO (-) OVRWRT SCORES
4586	020074	012737	006752	001310	MOV	#POSCRO,\$TMP13	;GET POINTER TO (+) OVRWRT SCORES
4587	020102	012705	000007		MOV	#7,R5	;SET DIVISOR = 7.
4588	020106	122737	000001	003143	CMPB	#1,TSTTYP	;SEE IF READING OVRWRT DATA
4589	020114	001410			BEQ	4\$	;BR IF YES
4590	020116	012737	007112	001304	MOV	#NCSCRO,\$TMP11	;GET POINTER TO (-) COMPAT DATA SCORES
4591	020124	012737	007252	001310	MOV	#PCSCRO,\$TMP13	;GET POINTER TO (+) COMPAT DATA SCORES
4592	020132	012705	000140		MOV	#96,R5	;SET DIVISOR = 96.
4593	020136	012737	000060	001306	4\$: MOV	#48,\$TMP12	;SET COUNTER = 48.
4594	020144	105037	003156		12\$: CLRB	OFSDIR	;INIT FLAG FOR (-) DIRECTION
4595	020150	005000			6\$: CLR	R0	;CLEAR HI BITS OF DIVIDEND AND DIVISOR
4596	020152	005001					
4597	020154	005002					
4598	020156	005004					
4599	020160	017703	161120		MOV	#\$TMP11,R3	;SET DIVIDEND = (-) SUM
4600	020164	105737	003156		TSTB	OFSDIR	;SEE WHICH OFST DIRECTION
4601	020170	001402			BEQ	10\$	
4602	020172	017703	161112		MOV	#\$TMP13,R3	;SET DIVIDEND = (+) SUM
4603	020176	004737	042676		10\$: JSR	PC,M.DP10	;DIVIDE TO GET AVERAGES
4604	020202	010500			MOV	R5,R0	;GET DIVISOR
4605	020204	006200			ASR	R0	;DIVIDE IT BY 2
4606	020206	020100			CMP	R1,R0	;SEE IF REMNDR > HALF DIVISOR
4607	020210	103401			BLO	8\$	;BR IF NOT
4608	020212	005203			INC	R3	;ADD 1 TO QUOTIENT
4609	020214	105137	003156		8\$: COMB	OFSDIR	;COMPL DIR FLAG
4610	020220	001406			BEQ	14\$	;BR FOR (+)
4611	020222	010377	161056		MOV	R3,\$TMP11	;PUT (-) AVG INTO TABLE
4612	020226	062737	000002	001304	ADD	#2,\$TMP11	
4613	020234	000745			BR	6\$	
4614	020236	010377	161046		14\$: MOV	R3,\$TMP13	;PUT (+) AVG INTO TABLE
4615	020242	062737	000002	001310	ADD	#2,\$TMP13	
4616	020250	005337	001306		DEC	\$TMP12	;DECREMENT COUNTER

```

4617 020254 001333      BNE      12$      ;BR IF NOT DONE YET
4618 020256 104410      RESREG      ;RESTORE RO-R5
4619 020260 000207      RTS      PC      ;RETURN
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630 020262 104407      ;*****
4631                                     ;*TRKAVG - COMPUTE AVERAGES OF TEST SCORES OVER ALL SURFACES
4632                                     ;*THIS SUBROUTINE COMPUTES THE AVERAGE OF THE OVERWRITE TEST SCORES
4633                                     ;*OVER ALL TRACKS (SURFACES) AND PLACES THE AVERAGE IN OVRVAVG.
4634                                     ;*THEN, IT COMPUTES THE AVERAGE OF THE COMPATIBILITY DATA READ SCORES
4635                                     ;*OVER ALL TRACKS AND PLACES THE AVERAGE IN COMAVG.
4636                                     ;*****
4637 TRKAVG: SAVREG      ;SAVE RO-R5
4638 ;TAKE SUMS OF OVERWRITE AND COMPAT READ TEST SCORES OVER ALL TRACKS
4639      MOV      #DRVLIST,RO      ;INIT DRIVE LIST POINTER
4640      CLR      R1      ;INIT DRIVE COUNTER
4641      CLR      OVRVAVG      ;INIT SUMS TO 0
4642      CLR      COMAVG
4643      MOV      #NOSCR0,R2      ;SET POINTER TO OVRWRT SCORES
4644      MOV      #NCSCR0,R3      ;SET POINTER TO READ SCORES
4645      6$: TST      (RO)+      ;COMPUTE TOTAL NUMBER OF DRIVES
4646      BEQ      8$      ; IN R1
4647      INC      R1
4648      BR      6$
4649      8$: MOV      R1,RO      ;PUT NO. OF DRIVES IN RO
4650      MOV      R2,R4      ;SET POINTERS TO SCORES FOR THIS TRACK
4651      MOV      R3,R5
4652      9$: ADD      (R4)+,OVRVAVG      ;ADD SCORES TO SUMS
4653      ADD      94.(R4),OVRVAVG
4654      ADD      (R5)+,COMAVG
4655      ADD      94.(R5),COMAVG
4656      DEC      RO
4657      BNE      9$      ;SEE IF ALL SCORES ADDED FOR THIS TRACK
4658      ADD      #32.,R2      ;BR IF NOT YET
4659      ADD      #32.,R3      ;POINT TO SCORES FOR NEXT TRACK
4660      CMP      R2,#NOSCR0+64.      ;SEE IF ALL TRACKS DONE YET
4661      BLOS      8$      ;BR IF NOT YET
4662      ;DIVIDE TO GET AVERAGES
4663      CLR      R4      ;GET DIVISOR IN R4-R5
4664      CLR      R5
4665      ADD      R1,R5
4666      ADD      R1,R5
4667      ADD      R1,R5
4668      ASL      R5
4669      MOV      #OVRVAVG,$TMP11      ;SET POINTER TO OVRWRT SCORE SUM
4670      CLR      $TMP12      ;CLEAR SCORE INDICATOR
4671      10$: CLR      RO      ;GET DIVIDEND IN RO-R1-R2-R3
4672      CLR      R1
4673      CLR      R2
4674      MOV      2,$TMP11,R3
4675      JSR      PC,M.DP1D      ;DIVIDE TO GET AVERAGE
4676      MOV      R5,RO      ;GET DIVISOR
4677      ASR      RO      ;DIVIDE IT BY 2
4678      CMP      R1,RO      ;SEE IF REMNDR > HALF DIVISOR
4679      BLO      12$      ;BR IF NOT

```

```

4673 020452 005203          INC      R3          ;ADD 1 TO QUOTIENT
4674 020454 010377 160624    12$:    MOV      R3,$TMP11 ;STORE AVERAGE SCORE
4675 020460 012737 007430    001304  MOV      $COMAVG,$TMP11 ;SET POINTER FOR READ SCORE SUM
4676 020466 005137 001306    COM      $TMP12 ;COMPLEMENT SCORE INDICATOR
4677 020472 001354          BNE      10$          ;BR TO COMPUTE READ SCORE AVERAGE
4678 020474 104410          RESREG ;RESTORE R0-R5
4679 020476 000207          RTS      PC          ;RETURN
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689 020500 032765 010000 000020 NEDHDL: BIT      $NED,P.CS2(R5) ;SEE IF NED ON DRIVE SELECT
4690 020506 001002          BNE      1$          ;BR IF NED SET
4691 020510 000137 031504    JMP      ERRHDL ;GO HANDLE OTHER ERROR
4692 020514 010046          1$:    MOV      R0,-(SP) ;SAVE R0,R1
4693 020516 010146          MOV      R1,-(SP)
4694 020520 012700 000001    MOV      $1,R0 ;SET BIT POINTER
4695 020524 005001          CLR      R1 ;CLEAR COUNTER
4696 020526 120137 005430    2$:    CMPB    R1,DRIVE ;SEE IF R1 = CURRENT DRIVE NUMBER
4697 020532 001403          BEQ      3$          ;BR IF =
4698 020534 005201          INC      R1 ;INCREMENT COUNTER
4699 020536 006300          ASL      R0 ;SHIFT BIT POINTER
4700 020540 000772          BR       2$          ;TRY AGAIN
4701 020542 040037 005476    3$:    BIC      R0,NEWON ;CLEAR ONLINE BIT FOR THIS DRIVE
4702 020546 012601          MOV      (SP)+,R1 ;RESTORE R0,R1
4703 020550 012600          MOV      (SP)+,R0
4704 020552 000137 033754    JMP      RETNML ;TAKE NORMAL RETURN
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716 020556 005037 005424          DCKHDL: CLR      RECODE ;CLEAR ERROR FLAGS
4717 020562 032765 000400 000034  BIT      $HVRC,P.ER(R5) ;SEE IF HEADER VRC ERROR
4718 020570 001404          BEQ      4$          ;BR IF NOT
4719 020572 052737 000004 005424  BIS      $HVRCER,RECODE ;SET HVRC ERROR FLAG
4720 020600 000411          BR       8$          ;EXIT
4721 020602 032765 100000 000034  4$:    BIT      $DCK,P.ER(R5) ;SEE IF DCK ERROR OCCURRED
4722 020610 001002          BNE      6$          ;BR IF YES
4723 020612 000137 031504    JMP      ERRHDL ;GO HANDLE OTHER ERROR
4724 020616 052737 000020 005424  6$:    BIS      $DCKERR,RECODE ;SET DCK ERROR FLAG
4725 020624 105237 003144    8$:    INCB    DCEFLG ;INCREMENT DCK ERROR FLAG
4726 020630 000137 033754    JMP      RETNML ;TAKE NORMAL RETURN FROM DRIVER
4727
4728

```

```

4729
4730
4731
4732
4733
4734 020634 010146
4735 020636 017701 160304
4736 020642 042701 177600
4737 020646 120127 000172
4738 020652 003005
4739 020654 120127 000141
4740 020660 002402
4741 020662 042701 000040
4742 020666 010137 005442
4743 020672 122701 000003
4744 020676 001007
4745 020700 104401 012102
4746 020704 042777 000100 160232 1$:
4747 020712 012601
4748 020714 000002
4749 020716 122701 000032 2$:
4750 020722 001003
4751 020724 104401 012107
4752 020730 000765
4753 020732 122701 000022 3$:
4754 020736 001003
4755 020740 104401 012114
4756 020744 000757
4757 020746 122701 000007 4$:
4758 020752 001003
4759 020754 104401 012126
4760 020760 000751
4761 020762 104401 005442 5$:
4762 020766 104401 001325
4763 020772 000744
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777 020774 005077 160146
4778 021000 005037 005442
4779 021004 052777 000100 160132
4780 021012 000207
4781
4782
4783
4784

;*****
;SBTTL KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
;*****
KBDHDL: MOV R1, -(SP) ;SAVE R1 ON STACK
        MOV @STKB, R1 ;READ A CHAR FROM KBD BUFFER
        BIC #177600, R1 ;MASK OUT UNUSED BITS
        CMPB R1, #172 ;SEE IF LOWER CASE TYPED
        BGT 20$ ;BR IF NOT
        CMPB R1, #141 ;BR IF NOT
        BLT 20$ ;MAKE IT UPPER CASE
        BIC #BIT5, R1 ;SAVE INPUT CHARACTER
        MOV R1, INTCHR ;SEE IF (↑C) TYPED
        CMPB #003, R1 ;BR IF NOT (↑C)
        BNE 2$ ;ECHO (↑C)
        TYPE ,CNTRLC ;CLEAR KBD INTERRUPT ENABLE BIT
        BIC #BIT6, @STKB ;RESTORE R1
        MOV (SP)+, R1 ;RETURN
        RTI ;SEE IF (↑Z) TYPED
        CMPB #032, R1 ;BR IF NOT (↑Z)
        BNE 3$ ;ECHO (↑Z)
        TYPE ,CNTRLZ ;RETURN
        BR 1$ ;SEE IF (↑R) TYPED
        CMPB #022, R1 ;BR IF NOT (↑R)
        BNE 4$ ;ECHO (↑R)
        TYPE ,CNTRLR ;RETURN
        BR 1$ ;SEE IF (↑G) TYPED
        CMPB #007, R1 ;BR IF NOT (↑G)
        BNE 5$ ;ECHO (↑G)
        TYPE ,CNTRLG ;RETURN
        BR 1$ ;ECHO INPUT
        TYPE ,INTCHR ;DO <CR> AND <LF>
        TYPE ,$CRLF ;RETURN
        BR 1$

;*****
;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
;*****
PREPKB: CLR @STKB ;CLEAR KBD BUFFER AND DONE BIT
        CLR INTCHR ;CLEAR TTY INPUT WORD
        BIS #BIT6, @STKB ;ENABLE KBD INTERRUPT
        RTS PC ;RETURN

;*****

```

4785
 4786
 4787
 4788
 4789
 4790
 4791
 4792
 4793 021014 104401 005442
 4794 021020 104401 001324
 4795 021024 000207
 4796
 4797
 4798
 4799
 4800
 4801
 4802
 4803
 4804
 4805
 4806
 4807
 4808 021026 016646 000002
 4809 021032 104403
 4810 021034 006
 4811 021035 000
 4812 021036 104401 011254
 4813 021042 004737 023424
 4814 021046 021100
 4815 021050 021100
 4816 021052 021100
 4817 021054 005700
 4818 021056 001001
 4819 021060 000207
 4820 021062 020027 000006
 4821 021066 003407
 4822 021070 104401 005212
 4823 021074 104401 001324
 4824 021100 162716 000010
 4825 021104 000207
 4826 021106 012746 005212
 4827 021112 004737 042116
 4828 021116 021070
 4829 021120 012600
 4830 021122 005737 042250
 4831 021126 001360
 4832 021130 010066 000002
 4833 021134 000207
 4834
 4835
 4836
 4837
 4838
 4839
 4840

.SBTTL ECOBAD - ECHO BAD TTY INPUT
 ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
 ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
 ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
 ;*AND <LF> ARE DONE.
 ;* CALL - JSR PC,ECOBAD
 ;*****

ECOBAD: TYPE ,INTCHR ;ECHO BAD CHARACTER
 TYPE ,SQUES ;TYPE (?) AND <CR>, <LF>
 RTS PC ;RETURN

;*****
 ;* 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.
 ;*****

GETPRM: MOV 2(SP),-(SP) ;GET OLD VALUE
 TYP0S ;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 ,BUFF0 ;ECHO BAD INPUT
 TYPE ,SQUES
 7\$: SUB #10,(SP) ;FIX ERROR RETURN PC
 RTS PC ;ERROR RETURN
 8\$: MOV #BUFF0,-(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 \$HI0C↑ ;SEE IF HI BITS ARE 0
 BNE 6\$;BR IF NOT
 MOV R0,2(SP) ;PUT NEW VALUE ON STACK
 RTS PC ;RETURN

;*****
 .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

```

4841      ; * "SWR = XXXXXX NEW = " AND WAITS FOR A NEW OCTAL VALUE
4842      ; * OF UP TO SIX DIGITS TO BE TYPED.
4843      ; *****
4844      021136 022737 000176 001140 GTSWRG: CMP      #SWREG,SWR      ;SEE IF SOFTWARE SWR SELECTED
4845      021144 001010      BNE      6$      ;BR IF NOT
4846      021146 013746 000176      MOV      SWREG,-(SP)      ;PUT OLD VALUE ON STACK
4847      021152 104401 011245      TYPE     SWMSG      ;TYPE "SWR = "
4848      021156 004737 021026      JSR      PC,GETPRM      ;TYPE OLD, GET NEW SWREG VALUE
4849      021162 012637 000176      MOV      (SP)+,SWREG      ;STORE NEW VALUE
4850      021166 000207      6$:      RTS      PC      ;RETURN
4851
4852
4853
4854
4855
4856
4857
4858
4859      ; *****
4860      SBTTL  INITSS - INITIALIZE SUBSYSTEM
4861      ; *
4862      ; * THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
4863      ; * AND DOES A SUBSYSTEM CLEAR.
4864      ; * USES - R2,R5
4865      ; * CALL:
4866      ; *
4867      ; *      JSR      PC,INITSS
4868      ; *****
4869      021170 012737 031504 003056 INITSS: MOV      #ERRHDL,A.ABNL      ;SET UP ABNORMAL ERROR RETURN ADDRESS
4870      021176 012737 030272 003054      MOV      #ERRFRE,A.NORM      ;SET UP NORMAL RETURN ADDRESS
4871      021204 013702 003046      MOV      RKBAS,R2      ;GET ADDRESS OF RK611 REGISTERS
4872      021210 012705 002640      MOV      #PARMO,R5      ;GET ADDRESS OF PARAMETER BLOCK
4873      021214 105037 003150      CLRB      NORTRY      ;CLEAR "NO-RETRY" FLAG
4874      021220 004737 021270      JSR      PC,CLPRM      ;CLEAR DRIVER INPUT PARAMETERS
4875      021224 112765 000177 000001      MOVVB   #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
4876      021232 004737 030146      JSR      PC,DRVCL      ;DO SUBSYSTEM CLEAR
4877      021236 113765 005430 000000      MOVVB   DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
4878      021244 113737 005430 002724      MOVVB   DRIVE,PARM1      ;SET DRIVE NO. IN ALTERNATE P.B.
4879      021252 113765 003134 000007      MOVVB   FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
4880      021260 153765 003133 000007      BISB     TYPFMT,P.CS1H(R5) ;SET DRV TYPE. 0=RK06, 4=RK07
4881      021266 000207      RTS      PC      ;SUBROUTINE EXIT
4882
4883
4884
4885      ; *****
4886      SBTTL  CLPRM - CLEAR DRIVER INPUT PARAMETERS
4887      ; * THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
4888      ; * PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
4889      ; * CALL - JSR      PC,CLPRM
4890      ; *****
4891
4892      021270 010046      CLPRM: MOV      R0,-(SP)      ;SAVE R0
4893      021272 010546      MOV      R5,-(SP)      ;SAVE R5
4894      021274 010500      MOV      R5,R0      ;GET PARAMETER BLOCK ADDRESS
4895      021276 062705 000016      ADD      #P.CS1,R5      ;COMPUTE LIMIT ADDRESS
4896      021302 005020      1$:      CLR      (R0)+      ;CLEAR A WORD IN PARAMETER BLOCK

```

```

4897 021304 020005      CMP      R0,R5      ;SEE IF DONE YET
4898 021306 001375      BNE      1$      ;BR IF NOT DONE YET
4899 021310 012605      MOV      (SP)+,R5      ;RESTORE R5
4900 021312 012600      MOV      (SP)+,R0      ;RESTORE R0
4901 021314 000207      RTS      PC      ;RETURN
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920 021316 004737 021170      SCNDRV: JSR      PC,INITSS      ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
4921 021322 042712 000100      3$:      BIC      #IE,(R2)      ;DISABLE RK06 INTERRUPT
4922 021326 113762 005430      MOV      DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
4923 021334 105737 003133      TSTB     TYPFMT      ;SEE IF RK07
4924 021340 001003      BNE      5$      ;BR IF YES
4925 021342 012712 000001      MOV      #GO,(R2)      ;ELSE SET GO BIT FOR RK06 IN RKCS1
4926 021346 000402      BR      7$
4927
4928 021350 012712 002001      5$:      MOV      #<CDT!GO>,(R2) ;SET GO FOR RK07
4929 021354 005037 005444      7$:      CLR      SCRACH      ;CLEAR STALL COUNTER
4930 021360 005237 005444      15$:     INC      SCRACH      ;INCREMENT STALL COUNTER
4931 021364 001375      BNE      15$      ;STALL FOR SEVERAL MILLI-SEC
4932 021366 032762 001000      BIT      #MDS,RKCS2(R2) ;SEE IF MDS ERROR
4933 021374 001417      BEQ      18$      ;BR IF NOT
4934 021376 112765 000101      MOV      #SELDV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
4935 021404 011265 000016      MOV      (R2),P.CS1(R5) ;GET RKCS1 FOR REPORT
4936 021410 004737 037644      JSR      PC,I.CST1      ;GET OTHER RK611 REGS FOR REPORT
4937 021414 004737 031224      JSR      PC,REPSUP      ;SET UP ERROR REPORT
4938 021420 104052      ERROR      52      ;REPORT MDS ERROR
4939 021422 052737 000200      BIS      #ABORT,RECODE      ;SET ABORT FLAG
4940 021430 000137 033336      JMP      ALLTRM      ;GO ABORT TESTING
4941 021434 004737 021170      18$:     JSR      PC,INITSS      ;INIT THE S.S.
4942 021440 112765 000141      MOV      #ROSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
4943 021446 012737 020500      MOV      #NEDHDL,A.ABNL      ;SET NED ABNORMAL RETURN ADDRESS
4944 021454 012737 000377      MOV      #377,NEWON      ;INIT. ON-LINE INDICATOR
4945 021462 004737 030146      JSR      PC,DRVCL      ;READ ALL DRIVE STATUS
4946 021466 112737 000040      MOV      #40,BADDRV+7      ;PUT A SPACE IN MSG BUF
4947
4948 021474 012737 031504      ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
4949 021502 022737 000377      MOV      #ERRHDL,A.ABNL      ;RESTORE ABNL RETURN ADDRESS
4950 021510 001425      CMP      #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
4951 021512 113737 005430      BEQ      4$      ;BR IF NED NOT SET
4952 021520 152737 000060      MOV      DRIVE,BADDRV+6      ;GET DRIVE NO. INTO BUF
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999

```

```

4953 021526 104401 011303 TYPE ,BADDRV ;TYPE "DRIVE X"
4954 021532 104401 011314 TYPE ,NXDRIV ;TYPE "NON-EXISTENT"
4955 021536 132737 000200 001351 BITB #BIT7,$ENVM ;SEE IF UNDER APT
4956 021544 001401 BEQ 2$ ;BR IF NO
4957 021546 104123 ERROR 123 ;NED UNDER APT SIZING
4958 ;SERVICE ERRORS HERE
4959 021550 042762 000100 000000 2$: BIC #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT
4960 021556 017616 000000 MOV 2(SP),(SP) ;SET UP ERROR RETURN ADDRESS
4961 021562 000207 RTS PC ;ERROR RETURN
4962 ;SEE IF DRIVE IS READY
4963 021564 032765 000200 000040 4$: BIT #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
4964 021572 001013 BNE 6$ ;BR IF DRIVE IS READY
4965 021574 113737 005430 011311 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
4966 021602 152737 000060 011311 BISB #0,BADDRV+6 ;CONVERT TO ASCII
4967 021610 104401 011303 TYPE ,BADDRV ;TYPE "DRIVE X"
4968 021614 104401 011333 TYPE ,NTREDY ;TYPE "NOT READY"
4969 021620 000753 BR 2$ ;TAKE ERROR EXIT
4970 ;SEE IF DRIVE IS WRITE ENABLED
4971 021622 032765 004000 000040 6$: BIT #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
4972 021630 001413 BEQ 8$ ;BR IF WRITE LOCK NOT SET
4973 021632 113737 005430 011311 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
4974 021640 152737 000060 011311 BISB #0,BADDRV+6 ;CONVERT TO ASCII
4975 021646 104401 011303 TYPE ,BADDRV ;TYPE "DRIVE X"
4976 021652 104401 011347 TYPE ,WRTLOK ;TYPE "WRITE-LOCKED"
4977 021656 000734 BR 2$ ;TAKE ERROR EXIT
4978 ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
4979 021660 112765 000103 000001 8$: MOVB #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
4980 021666 004737 030146 JSR PC,DRVCL ;SET VOLUME VALID
4981 021672 112765 000113 000001 MOVB #RECAL,P.CMND(R5) ;SET RECAL CMND
4982 021700 004737 030146 JSR PC,DRVCL ;RECAL THE DRIVE
4983 021704 112765 000121 000001 MOVB #RDATA,P.CMND(R5) ;SET READ COMMAND
4984 021712 013765 024404 000002 MOV LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)/1456(8)
4985 021720 112765 000002 000005 MOVB #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
4986 021726 012765 053342 000010 MOV #RWBUFF,P.BALO(R5) ;BUS ADDRESS
4987 021734 012765 177774 000012 MOV #-4,P.WC(R5) ;READ 4 WORDS
4988 021742 142765 000020 000007 BICB #B.CFMT,P.CS1H(R5) ;SET 22 SECTOR FORMAT
4989 021750 153765 003133 000007 BISB TYPFMT,P.CS1H(R5)
4990 021756 004737 030146 JSR PC,DRVCL ;READ 4 WORDS OF BAD SECTOR FILE
4991 021762 032737 100000 005424 BIT #ANYDER,RECODE ;SEE IF DATA ERROR
4992 021770 001402 BEQ 10$ ;BR IF OK
4993 021772 104401 012036 TYPE ,BAD632 ;TYPE READ ERROR MESSAGE
4994 021776 022737 177777 053350 10$: CMP #177777,RWBUFF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
4995 022004 001013 BNE 12$ ;BR IF NOT ALL 1'S
4996 022006 113737 005430 011311 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
4997 022014 152737 000060 011311 BISB #0,BADDRV+6 ;CONVERT TO ASCII
4998 022022 104401 011303 TYPE ,BADDRV ;TYPE "DRIVE X"
4999 022026 104401 011366 TYPE ,ALNPAK ;TYPE "LOADED WITH ALIGN PACK"
5000 022032 000646 BR 2$ ;TAKE ERROR EXIT
5001 ;ERROR FREE RETURN
5002 022034 062716 000002 12$: ADD #2,(SP) ;FIX UP RETURN PC
5003 022040 000207 RTS PC ;RETURN

```

```

;*****
;SETUP - SET UP FOR LOOP ON ERROR
;THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER

```

```

5009 ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
5010 ;*****
5011 322042 011637 001076 SETUP: MOV (SP),2*STACK-2 ;MOVE RETURN PC ON STACK
5012 022046 012706 001076 MOV #STACK-2,SP ;RE-INIT THE STACK POINTER
5013 022052 005037 005424 CLR RECODE ;CLEAR ERROR RECOVERY FLAGS
5014 022056 105037 003135 CLRB EPRCNT ;CLEAR RETRY ERROR COUNT
5015 022062 012705 002640 MOV #PARMO,RS ;SET PARAM BLK ADRS
5016 022066 112765 000177 000001 MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
5017 022074 004737 030146 JSR PC,DRVCAL ;CLEAR THE SUBSYSTEM
5018 022100 012737 000000 177776 MOV #PRO,2*PS ;RE-ESTABLISH PRIORITY 0
5019 022106 000207 RTS PC ;RETURN
5020
5021
5022
5023
5024
5025 ;*****
5026 $BTTL MTCART - MOUNT TEST CARTRIDGE, CHECK DRIVE STATUS
5027 ;*THIS SUBROUTINE PERFORMS THE FOLLOWING FUNCTIONS :
5028 ;*IT TYPES "STARTING PASS X", DIRECTS THE OPERATOR TO MOUNT THE
5029 ;*TEST CARTRIDGE, CHECKS THE DRIVE STATUS, DETERMINES THE PACK FORMAT,
5030 ;*READS THE BAD SECTOR FILES, AND TYPES THE DRIVE AND CART. SERIAL
5031 ;*NUMBERS.
5032 ;*****
5033 MTCART:
5034 022110 013700 005554 MOV DRVPTR,R0 ;GET POINTER TO FIRST DRIVE FOR THIS SUBSYS
5035 022114 112037 010366 (R0)+,MNTPAK+22. ;PUT DRIVE NAME INTO MSG
5036 022120 111037 010365 MOVB (R0),MNTPAK+21.
5037 022124 162700 005512 SUB #DRVLIST,R0 ;SUBTRACT LIST ADRS FROM POINTER
5038 022130 006200 ASR R0 ;DIVIDE BY TWO TO GET DRIVE NO.
5039 022132 110037 003154 MOVB R0,LOGDRV ;STORE LOGICAL DRIVE NUMBER
5040 022136 004737 021170 JSR PC,INITSS ;INITIALIZE SUBSYSTEM
5041 022142 042762 000100 000000 BIC #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT FOR MAN. INTERVENTION
5042 022150 104401 010340 10$: TYPE ,MNTPAK ;TYPE MOUNT PACK MSG
5043 022154 104401 010602 5$: TYPE ,TYPRWN ;TYPE "TYPE R<CR> WHEN "
5044 022160 104401 010623 TYPE ,DRIRDY ;TYPE "DRIVE READY : "
5045 022164 004737 023424 JSR PC,RDCHRS ;WAIT FOR R<CR> RESPONSE
5046 022170 022214 12$: (1C) RETURN ADDRESS
5047 022172 022236 16$: (1Z) RETURN ADDRESS
5048 022174 022154 5$: (1U) RETURN ADDRESS
5049 022176 005700 TST R0 ;SEE IF NULL INPUT
5050 022200 001025 BNF 8$ ;BR IF NOT NULL
5051 022202 104401 005212 6$: TYPE ,BUFFD ;ECHO BAD INPUT
5052 022206 104401 001324 TYPE ,SQUES ;TYPE "<?> AND <CR>,<LF>"
5053 022212 000760 BR 5$ ;GO ASK AGAIN
5054 022214 012706 001100 12$: MOV #STACK,SP ;RE-INIT THE STACK
5055 022220 105737 003126 TSTB MDFLAG ;SEE IF 200 START
5056 022224 001002 BNE 14$ ;BR IF NOT
5057 022226 000137 012176 JMP DFSTRT ;RESTART PROGRAM - 200 START
5058 022232 000137 013002 14$: JMP ALLSYS ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
5059 022236 105737 003126 16$: TSTB MDFLAG ;SEE IF 200 START
5060 022242 001744 BEQ 5$ ;BR IF YES, TO ASK AGAIN
5061 022244 012706 001100 MOV #STACK,SP ;RE-INIT THE STACK
5062 022250 000137 013446 JMP ASKSYS ;GO ASK FOR SUBSYS TO TEST AGAIN
5063 022254 023727 005212 000122 8$: CMP BUFFD,#'R ;SEE IF R<CR> TYPED
5064 022262 001347 BNE 6$ ;BR IF NOT

```

K08

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 102
MTCART - MOUNT TEST CARTRIDGE, CHECK DRIVE STATUS

SEQ 0101

```

5065 022264 004737 021170      JSR    PC,INITSS      ;INITIALIZE S.S.
5066                                ;CHECK STATUS OF DRIVE TO BE TESTED NEXT
5067 022270 117737 163260 005430  MOVB   @DRVPTR,DRIVE  ;SET DRIVE NUMBER
5068 022276 142737 000060 005430  BICB   #'0,DRIVE      ;STRIP THE ASCII
5069 022304 004737 021316      JSR    PC,SCNDRV      ;CHECK STATUS OF THIS DRIVE -
5070 022310 022150                                ;IF NOT VALID, TYPE MSG AND RETURN
5071                                ;TO WAIT FOR R<CR> AGAIN
5072                                ;READ THE HEADER ON SECTOR 0, TRACK 0, CYL 0, AND GET THE DRIVE FORMAT
5073 022312 004737 021170      JSR    PC,INITSS      ;INIT. DRIVER PARAMS AND S.S.
5074 022316 112765 000125 000001  MOVB   @RDHEAD.P.CMND(R5) ;SET READ HEADER COMMAND
5075 022324 004737 030146      JSR    PC,DRVCAL      ;DO A READ HEADER
5076 022330 013762 024406 000000  MOV     HOLD1,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 0
5077 022336 032762 000200 000000 24$: BIT     @RDY,RKCS1(R2) ;WAIT FOR READY
5078 022344 001774                                BEQ     24$
5079 022346 013762 024410 000000  MOV     HOLD2,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
5080 022354 032762 000200 000000 26$: BIT     @RDY,RKCS1(R2) ;WAIT FOR READY
5081 022362 001774                                BEQ     26$
5082 022364 105037 003134      CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
5083 022370 005762 000024      TST     RKDB(R2)         ;POP SILO ONE TIME
5084 022374 032762 001000 000024  BIT     @BIT9,RKDB(R2) ;TEST FORMAT BIT IN HEADER WORD 2
5085 022402 001403                                BEQ     28$
5086 022404 152737 000020 003134  BISB    @B.CFMT,FORMAT ;SET 20 SECTOR FORMAT FOR THIS DRIVE
5087 022412 004737 021170      JSR    PC,INITSS      ;INIT THE S.S.
5088 022416 004737 026540      JSR    PC,REDBSF      ;READ BAD SECTOR FILE FOR THIS DRIVE
5089 022422 105737 003126      TSTB    MDFLAG          ;SEE IF 200 START
5090 022426 001004                                BNE     32$
5091 022430 122737 000061 003155  CMPB    #'1,PASSNO      ;SEE IF FIRST PASS
5092 022436 001012                                BNE     30$
5093 022440 004737 023736      JSR    PC,DRVSR      ;TYPE "DRIVE SER. NO. XXX"
5094 022444 004737 024056      JSR    PC,CRTSR      ;TYPE "CART. SER. NO. XXXXXXXXXXXX"
5095 022450 032777 000200 156462  BIT     @BIT7,@SWR      ;SEE IF BAD SECTORS SHOULD BE TYPED
5096 022456 001402                                BEQ     30$
5097 022460 004737 027030      JSR    PC,TYPBSF      ;TYPE BAD SECTOR FILES
5098 022464 005037 005442      CLR     INTCHR        ;INIT. TTY INPUT CHAR BUFFER
5099 022470 000207      RTS     PC                    ;RETURN
5100
5101
5102
5103 ;*****
5104 ;SBTTL DMCART - DISMOUNT TEST CARTRIDGE
5105 ;*THIS SUBROUTINE DIRECTS THE OPERATOR IN THE DISMOUNTING
5106 ;*OF THE TEST CARTRIDGE ON THE DRIVE CURRENTLY UNDER TEST.
5107 ;*****
5108 DMCART:
5109 022472 013700 005554      MOV     DRVPTR,R0      ;GET DRIVE LIST POINTER
5110 022476 112037 010423      MOVB    (R0)+,DISPAK+15. ;PUT DRIVE NAME INTO MSG
5111 022502 111037 010422      MOVB    (R0),DISPAK+14.
5112 022506 004737 021170      JSR    PC,INITSS      ;INITIALIZE SUBSYS
5113 022512 042762 000100 000000  BIC     @IE,RKCS1(R2) ;DISABLE RK06 INTRPT FOR MANUAL INTERVENTION
5114 022520 104401 010404      TYPE     ,DISPAK      ;TYPE "UNLOAD DRIVE XX AND REMOVE PACK"
5115 022524 104401 010602      TYPE     ,TYPRWN      ;TYPE "TYPE R<CR> WHEN "
5116 022530 104401 010642      TYPE     ,DRIDON      ;TYPE "DONE : "
5117 022534 004737 023424      JSR    PC,RDCHRS      ;WAIT FOR R<CR> RESPONSE
5118 022540 022564      12$
5119 022542 022606      16$
5120 022544 022524      5$
; (↑C) RETURN ADDRESS
; (↑Z) RETURN ADDRESS
; (↑U) RETURN ADDRESS

```

```

5121 022546 005700          TST      RO          ;SEE IF NULL INPUT
5122 022550 001025          BNE      B$          ;BR IF NOT NULL
5123 022552 104401 005212    6$:      TYPE     ,BUFFO    ;ECHO BAD INPUT
5124 022556 104401 001324    TYPE     ,SQUES    ;TYPE '<' AND '<CR>,<LF>'
5125 022562 000760          BR       S$          ;GO ASK AGAIN
5126 022564 012706 001100    12$:     MOV      #STACK,SP ;RE-INIT THE STACK
5127 022570 105737 003126    TSTB     MDFLAG    ;SEE IF 200 START
5128 022574 001002          BNE      14$        ;BR IF NOT
5129 022576 000137 012176    JMP      DFSTRT   ;RESTART PROGRAM - 200 START
5130 022602 000137 013002    14$:     JMP      ALLSYS  ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
5131 022606 105737 003126    16$:     TSTB     MDFLAG    ;SEE IF 200 START
5132 022612 001744          BEQ      S$          ;BR IF YES, TO ASK AGAIN
5133 022614 012706 001100    MOV      #STACK,SP ;RE-INIT THE STACK
5134 022620 000137 013446    JMP      ASKSYS  ;GO ASK FOR SUBSYS TO TEST AGAIN
5135 022624 023727 005212 000122 8$:     CMP      BUFFO,#'R ;SEE IF R<CR> TYPED
5136 022632 001347          BNE      B$          ;BR IF NOT
5137 022634 004737 021170    JSR      PC,INITSS ;INITIALIZE S.S.
5138 022640 000207          RTS      PC          ;RETURN
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150 022642 104407          ;*****
5151          ;SBTTL INTBLD - INITIALIZE TABLED
5152          ;*THIS SUBROUTINE LOADS TABLED WITH THE INITIAL PATTERN OF DRIVE NUMBERS
5153          ;*WHICH IS THE CYLINDER BLOCK LAYOUT FOR THE FIRST CURRENT ZONE. IT USES
5154          ;*THE STARTING DRIVE NUMBER LIST, STDLS, FOR THE FIRST COLUMN, AND
5155          ;*INCREMENTS THESE DRIVE NUMBERS ACROSS THE TABLE, FORMING ROWS. FOR
5156          ;*ANY DRIVE WHICH IS NOT PRESENT, THE ENTRY IS SET = 377.
5157          ;*****
5158          INTBLD: SAVREG          ;SAVE R0-R5
5159          ;LOAD THE FIRST COLUMN OF TABLED FROM STDLS
5160          MOV      #STDLS,R0      ;STARTING ADRS OF STDLS
5161          MOV      #TABLED,R1     ;STARTING ADRS OF TABLED
5162          MOV      #16,R3         ;SET COUNTER = 16
5163          4$:     MOVB    (R0)+,(R1) ;LOAD A BYTE INTO TABLED
5164          ADD      #16,R1         ;INCREMENT TABLED POINTER
5165          DEC      R3             ;DECR COUNTER
5166          BNE     4$             ;BR IF NOT DONE YET
5167          ;GET INDEX OF FIRST ZERO ENTRY IN DRVLST INTO R1
5168          MOV      #DRVLST,R0     ;STARTING ADRS OF DRVLST
5169          CLR      R1             ;INIT INDEX TO 0
5170          6$:     TST      (R0)+    ;SEE IF ENTRY = 0
5171          BEQ      B$             ;BR IF YES
5172          INC      R1             ;INCREMENT POINTER
5173          CMP      R1,#16         ;SEE IF DONE LOOKING YET
5174          BLT     6$             ;BR IF NOT DONE YET
5175          ;INCREMENT THE DRIVE NOS. TO FORM THE ROWS
5176          8$:     MOV      #TABLED,R0 ;INIT TABLED POINTER
5177          16$:    MOV      #15,R4    ;INIT ROW ELEMENT COUNTER
5178          MOVB    (R0)+,R3        ;INIT ROW ELEMENT VALUE
5179          18$:    INC      R3        ;INCR ROW ELEMENT VALUE
5180          CMP      R3,#20         ;SEE IF ROW ELEMENT IS VALID
5181          BLT     20$            ;BR IF VALUE IS OK
5182          CLR      R3             ;RESET VALUE TO 0
5183          20$:    CMP      R3,R1     ;SEE IF VALUE IS > ACTUAL DRIVES
5184          BLT     22$            ;BR IF VALUE OK
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196

```

```

5177 022744 112720 000377      MOVB    #377,(R0)+      ;SET INVALID ENTRY = 377
5178 022750 000401      BR      24$      ;PROCEED
5179 022752 110320      22$:  MOVB    R3,(R0)+      ;SET VALUE IN TABLE
5180 022754 005304      24$:  DEC     R4      ;DECR ROW ELEMENT COUNTER
5181 022756 001363      BNE     18$      ;BR IF NOT DONE WITH ROW YET
5182 022760 020027 006512      CMP     R0,#TABLED+256. ;SEE IF DONE WITH TABLE YET
5183 022764 103755      BLO     16$      ;BR IF NOT DONE YET
5184                                     ;SET INVALID ELEMENTS OF FIRST COLUMN = 377
5185 022766 012700 006112      MOV     #TABLED,R0      ;INIT COLUMN POINTER
5186 022772 012703 000020      MOV     #16.,R3      ;INIT COL. ELEMENT COUNTER
5187 022776 121001      26$:  CMPB    (R0),R1      ;SEE IF ELEMENT IS VALID
5188 023000 002402      BLT     30$      ;BR IF YES
5189 023002 112710 000377      MOVB    #377,(R0)      ;SET INVALID ELEMENT = 377
5190 023006 062700 000020      30$:  ADD     #16.,R0      ;INCREMENT COLUMN POINTER
5191 023012 005303      DEC     R3      ;SEE IF ALL DONE YET
5192 023014 001370      BNE     26$      ;BR IF NOT ALL DONE YET
5193 023016 104410      RESREG      ;RESTORE R0-R5
5194 023020 000207      RTS     PC      ;RETURN
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204 023022 104407      ;*****
5205 023024 012700 006512      ;SBTTL ROTBLD - ROTATE TABLED RIGHT 3 COLUMNS
5206 023030 010001      ;*THIS SUBROUTINE ROTATES ALL ELEMENTS OF TABLED RIGHT 3 COLUMNS,
5207 023032 012704 000015      ;*SO THAT THE DRIVE DATA WILL BE ROTATED SEVERAL SECTORS IN EACH
5208 023036 114037 001304      ;*CURRENT ZONE.
5209 023042 114037 001306      ;*****
5210 023046 114037 001310      ROTBLD: SAVREG      ;SAVE R0-R5
5211 023052 114041      MOV     #TABLED+256.,R0 ;POINT TO END OF TABLED
5212 023054 005304      MOV     R0,R1      ;GET A COPY IN R1
5213 023056 001375      4$:  MOV     #13.,R4      ;INIT ROW ELEMENT COUNTER
5214 023060 113741 001304      MOVB    -(R0),STMP11 ;SAVE LAST 3 ELEMENTS IN THIS ROW
5215 023064 113741 001306      MOVB    -(R0),STMP12
5216 023070 113741 001310      MOVB    -(R0),STMP13
5217 023074 020027 006112      6$:  MOVB    -(R0),-(R1) ;MOVE AN ELEMENT RIGHT 3 PLACES
5218 023100 101354      DEC     R4      ;DECR ROW ELEMENT COUNTER
5219 023102 104410      BNE     6$      ;BR IF NOT DONE YET
5220 023104 000207      MOVB    STMP11,-(R1) ;MOVE IN 3 SAVED ELEMENTS
5221                                     MOVB    STMP12,-(R1)
5222                                     MOVB    STMP13,-(R1)
5223                                     CMP     R0,#TABLED ;SEE IF DONE WITH TABLE YET
5224                                     BHI     4$      ;BR IF NOT DONE YET
5225                                     RESREG      ;RESTORE R0-R5
5226                                     RTS     PC      ;RETURN
5227
5228
5229
5230
5231
5232
;*****
;SBTTL WRTBLK - WRITE ONE CYLINDER BLOCK
;*THIS SUBROUTINE WRITES ALL THE SECTORS FOR THE CURRENT DRIVE
;*INTO THE CYLINDER BLOCK WHOSE STARTING CYL NUMBER IS IN
;*P.CYLN(R5) ON ENTRY. THE DATA IS WRITTEN ON ALL SURFACES
;*THE LOGICAL DRIVE NUMBER MUST BE IN LOGDRV ON ENTRY.
;*
;*IT CALLS WRTSEC TO ACTUALLY WRITE EACH 256-WORD SECTOR.
;*****

```

```

5233 023106 104407      WRTBLK: SAVREG      ;SAVE R0-R5
5234 023110 113700 003154      MOV      LOGDRV,R0      ;GET LOGICAL DRIVE NO.
5235 023114 006300      ASL      R0      ;CONVERT IT TO DATA PATTERN INDEX
5236 023116 016003 006512      MOV      TABLEG(R0),R3      ;GET DATA PATTERN WORD INTO R3
5237 023122 012701 006112      MOV      #TABLED,R1      ;GET TABLED POINTER
5238 023126 012704 000020      MOV      #16,R4      ;INIT CYLINDER COUNTER
5239 023132 010100      4$:      MOV      R1,R0
5240 023134 123720 003154      6$:      CMPB     LOGDRV,(R0)+      ;SEE IF DRIVE LISTED HERE
5241 023140 001375      BNE      6$      ;BR IF NOT FOUND YET IN THIS ROW
5242 023142 005300      DEC      R0      ;BACK UP BY 1 BYTE
5243 023144 160100      SUB      R1,R0      ;COMPUTE ROW POSITION
5244 023146 116065 006572 000004      MOVB     SECLST(R0),P.SEC      ;GET SECTOR NO. FROM ROW POSITION
5245 023154 105065 000005      CLRB     P.TRCK(R5)      ;SET TRACK = 0
5246 023160 004737 025770      8$:      JSR      PC,WRTSEC      ;WRITE 256 WORDS AT THIS SECTOR
5247 023164 105265 000005      INCB     P.TRCK(R5)      ;INCR TRACK NO.
5248 023170 126527 000005 000002      CMPB     P.TRCK(R5),#2      ;SEE IF ALL TRACKS DONE YET
5249 023176 003770      BLE      8$      ;BR IF NOT DONE YET
5250 023200 062701 000020      ADD      #16,R1      ;POINT TO START OF NEXT ROW
5251 023204 005265 000002      INC      P.CYL(R5)      ;INCR CYLINDER NO.
5252 023210 000004      DEC      R4      ;DECREMENT CYLINDER COUNTER
5253 023212 001341      BNE      4$      ;BR IF 16 CYLS NOT DONE YET
5254 023214 104410      RESREG      ;RESTORE R0-R5
5255 023216 000207      RTS      PC      ;RETURN
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272 023220 104407      ;*****
5273 023222 105737 003143      ;CKSCOR - DECREMENT DRIVE SCORES ROUTINE
5274 023226 001474      ;THIS SUBROUTINE CHECKS TO SEE IF THE PROGRAM IS CURRENTLY
5275 023230 032737 000024 005424      ;READING OVRWRT OR COMPAT DATA WITH OFFSETS AND HANDLING DATA CHECK
5276 023236 001470      ;ERRORS. IF SO, THE SCORE FOR THE DRIVE WHICH WROTE THE
5277      ;FAILING DATA IS DECREMENTED.
5278      ;ON ENTRY, THE BYTE LABELED "TSTTYP" HAS THE FOLLOWING
5279      ;POSSIBLE VALUES:
5280      ;   TSTTYP = 0 - NOT PRESENTLY READING WITH OFFSET
5281      ;   TSTTYP = 1 - PRESENTLY READING OVRWRT DATA WITH OFFSET
5282      ;   TSTTYP = 2 - PRESENTLY READING COMPAT DATA WITH OFFSET
5283      ;   TSTTYP = 3 - PRESENTLY READING SELF TEST DATA WITH OFFSET
5284      ;*****
5285      CKSCOR: SAVREG      ;SAVE R0-R5
5286      TSTB     TSTTYP      ;SEE IF CURRENTLY TESTING WITH OFFSET
5287      BEQ      20$      ;BR TO EXIT IF NOT
5288      BIT      #DCKERR!HVR CER,RECODE      ;SEE IF A DCK OR HVRC ERR OCCURRED
5289      BEQ      20$      ;BR IF NOT
5290      ;GET THE NO. OF THE DRIVE WHICH WROTE DATA WHICH CAUSED DCK OR HVRC ERR
5291      JSR      PC,GTPKAD      ;COMPUTE PACK ADDRESS OF ERROR
5292      CMPB     #3,TSTTYP      ;SEE IF DOING SELF-TEST
5293      BNE      4$      ;BR IF NOT
5294      CLRB     R4      ;SET DRIVE NO. = 0
5295      BR       10$      ;CONTINUE
5296      4$:      MOV      $REG5,R1      ;GET CYL NO. OF ERROR
5297      BIC      #177760,R1      ;CLEAR HI CYL BITS
5298      ASL      R1      ;CONVERT TO TABLED INDEX FOR ROW
5299      ASL      R1
5300      ASL      R1
5301      ASL      R1

```

```

5289 023300 062701 006112      ADD      #TABLED,R1      ; COMPUTE POINTER TO TABLED ROW
5290 023304 005000      CLR      RO      ; INIT SECLST INDEX TO 0
5291 023306 123760 001200 006572 6$: CMPB    $REG7,SECLST(RO) ; SEE IF SECTOR IS LISTED IN SECLST
5292 023314 001405      BEQ      8$      ; BR IF YES
5293 023316 005200      INC      RO      ; INCR SECLST INDEX
5294 023320 020027 000020      CMP      RO,#16.      ; SEE IF WHOLE LIST CHECKED YET
5295 023324 002770      BLT      6$      ; BR IF NOT YET
5296 023326 000434      BR      20$      ; SECTOR NOT LISTED - EXIT, BECAUSE
5297                                ; FAILING DRIVE IS NOT BEING TESTED
5298                                ; COMPUTE TABLED POINTER TO FAILING DRIVE
5298 023330 060001      8$: ADD      RO,R1      ; GET ACTUAL DRIVE NO.
5299 023332 111104      MOV      (R1),R4      ; SEE IF DRIVE IS NOT BEING TESTED
5300 023334 122704 000377      CMPB    #377,R4      ; BR IF DRIVE NOT TESTED
5301 023340 001427      BEQ      20$      ; GET ADRS OF NEG SCRATCH SCORES
5302 023342 012700 007432 10$: MOV      #NSCORD,RO      ; SEE WHICH OFST DIRECTION
5303 023346 105737 003156      TSTB    OFSDIR      ; BR IF NEG
5304 023352 001402      BEQ      12$      ; GET ADRS OF POS SCRATCH SCORES
5305 023354 012700 007512      MOV      #PSCORD,RO      ; SEE IF TRACK = 0
5306 023360 105737 001176 12$: TSTB    $REG6      ; BR IF YES
5307 023364 001410      BEQ      14$      ; POINT TO SCORES FOR TRACK 1
5308 023366 062700 000020      ADD      #16.,RO      ; SEE IF TRACK = 1
5309 023372 123727 001176 000001      CMPB    $REG6,#1      ; BR IF YES
5310 023400 001402      BEQ      14$      ; POINT TO SCORES FOR TRACK 2
5311 023402 062700 000020      ADD      #16.,RO      ; COMPUTE POINTER TO SCORE FOR DRIVE
5312 023406 060400 003157 14$: ADD      R4,RO      ; SEE IF THIS SCORE IS PERFECT SO FAR
5313 023410 121037      CMPB    (RO),OFSCNT      ; EXIT IF NOT
5314 023414 001001      BNE      20$      ; DECREMENT SCORE FOR THIS DRIVE
5315 023416 105310      DECB     (RO)      ; RESTORE RO-R5
5316 023420 104410 20$: RESREG      ; RETURN
5317 023422 000207      RTS      PC

```

```

;*****
;SMTL RDCHRS - READ A STRING OF KBD INPUT CHARS
;THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
;CHARACTERS AT THE KBD, TERMINATED BY (CR), AND PLACES
;THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
;RUB-OUT AND (IU) FEATURES ARE PROVIDED.
;IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
;RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
;TAKEN IF (IC) IS TYPED, THE SECOND IS TAKEN IF (IZ) IS
;TYPED, AND THE THIRD IS TAKEN IF (IU) OR INVALID INPUT IS TYPED.
;IF (IG) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
;FOR MODIFICATION, IF SELECTED, AND THEN THE (IG) RETURN
;IS TAKEN.
;THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
;IN RO.
;
; CALL -      JSR      PC,RDCHRS
;              <CONTROL-C RETURN ADDRESS>
;              <CONTROL-Z RETURN ADDRESS>
;              <CONTROL-U OR ERROR RETURN ADDRESS>
;              RETURN
;*****
RDCHRS:

```

5344 023424

5345	023424	010146			MOV	R1,-(SP)	;SAVE R1
5346	023426	010246			MOV	R2,-(SP)	;SAVE R2
5347	023430	005000			CLR	R0	;INITIALIZE CHARACTER COUNT
5348	023432	005001			CLR	R1	;INITIALIZE RUB-OUT INDICATOR
5349					;READ A CHARACTER		
5350	023434	104406			2\$:	RDCHR	;READ A CHARACTER
5351	023436	112602				MOVB	(SP),+,R2
5352							;GET CHARACTER INTO R2
5353	023440	122702	000003			;CHECK FOR (↑C)	
5354	023444	001006				CMPB	#003,R2
5355	023446	104401	012102			BNE	4\$
5356	023452	017666	000004			TYPE	CNTRLC
5357	023460	000523			3\$:	MOV	4(SP),4(SP)
5358						BR	24\$
5359	023462	122702	000032			;CHECK FOR (↑Z)	
5360	023466	001006			4\$:	CMPB	#032,R2
5361	023470	104401	012107			BNE	6\$
5362	023474	062766	000002	000004		TYPE	CNTRLZ
5363	023502	000763				ADD	4,4(SP)
5364						BR	3\$
5365	023504	122702	000025			;CHECK FOR (↑U)	
5366	023510	001006			6\$:	CMPB	#025,R2
5367	023512	104401	012121			BNE	8\$
5368	023516	062766	000004	000004		TYPE	CNTRLU
5369	023524	000752			7\$:	ADD	4,4(SP)
5370						BR	3\$
5371	023526	122702	000007			;CHECK FOR (↑G)	
5372	023532	001005			8\$:	CMPB	#007,R2
5373	023534	104401	012126			BNE	9\$
5374	023540	004737	021136			TYPE	CNTRLG
5375	023544	000764				JSR	PC,GTSWRG
5376						BR	7\$
5377	023546	122702	000177			;CHECK FOR RUB-OUT (DELETE)	
5378	023552	001020			9\$:	CMPB	#177,R2
5379	023554	005700				BNE	14\$
5380	023556	001726				TST	R0
5381	023560	005701				BEQ	2\$
5382	023562	001003				TST	R1
5383	023564	005201				BNE	11\$
5384	023566	104401	012137			INC	R1
5385	023572	005037	005444			TYPE	BKSLSH
5386	023576	005300			11\$:	CLR	SCRACH
5387	023600	116037	005212	005444		DEC	R0
5388	023606	104401	005444			MOVB	BUFF0(R0),SCRACH
5389	023612	000710				TYPE	SCRACH
5390	023614	005701				BR	2\$
5391	023616	001403			14\$:	TST	R1
5392	023620	104401	012137			BEQ	16\$
5393	023624	005001				TYPE	BKSLSH
5394						CLR	R1
5395	023626	122702	000015			;CHECK FOR CARRIAGE RETURN	
5396	023632	001426			16\$:	CMPB	#015,R2
5397						BEQ	19\$
5398	023634	005037	005444			;HANDLE POSSIBLE DIGIT	
5399	023640	110237	005444			CLR	SCRACH
5400	023644	104401	005444			MOVB	R2,SCRACH
						TYPE	,SCRACH

```

5401 023650 110260 005212          MOVB R2,BUFF0(R0) ;PUT CHARACTER INTO BUFFER
5402 023654 005200          INC R0 ;INCREMENT CHARACTER CCOUNTER
5403 023656 022700 000120          CMP #80.,R0 ;SEE IF TOO MANY CHARACTERS TYPED
5404 023662 001264          BNE 2$ ;BR IF NOT TOO MANY
5405 023664 104401 001325          TYPE $CRLF ;TYPE <CR> AND <LF>
5406 023670 112760 000000 005212  MOVB #0,BUFF0(R0) ;PUT TERMINATING NULL INTO BUFFER
5407 023676 104401 005212          TYPE ,BUFF0 ;ECHO INPUT STRING
5408 023702 104401 001324          TYPE $QUES ;TYPE <?> <CR> <LF>
5409 023706 000703          BR 7$ ;TAKE ERROR EXIT FROM RDCHRS
5410 023710 104401 001325          TYPE $CRLF ;TYPE <CR> <LF>
5411 023714 112760 000000 005212  MOVB #0,BUFF0(R0) ;PUT TERMINATING NULL INTO BUFFER
5412 023722 062766 000006 000004  ADD #6,4(SP) ;FIX UP RETURN ADDRESS
5413 023730 012602          MOV (SP)+,R2 ;RESTORE R2
5414 023732 012601          MOV (SP)+,R1 ;RESTORE R1
5415 023734 000207          RTS PC ;SUBROUTINE EXIT

```

```

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

```

```

DRVSR: SAVREG ;SAVE R0-R5
        JSR PC,INITSS ;CLEAR S.S. AND PARAMETERS
        MOVB #R0STAT,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 $SUPRS
        MOV #3,R3 ;INIT CHAR COUNT
        ROL R1 ;INITIALIZE BIT POSITIONS
4$: 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 4$ ;BR IF NOT 3 CHARS YET
    CLRB (R4) ;INSERT NULL TERMINATOR
    JSR PC,$$SUPRS ;TYPE DRIVE SER. NUMBER
    TYPE ,CRLF ;TYPE <CR> AND <LF>
    RESREG ;RESTORE R0-R5
    RTS PC ;RETURN

```

```

;*****
;SBTTL CRTSR - TYPE CARTRIDGE SERIAL NUMBER
;THIS SUBROUTINE TYPES "CART. SER. NO. XXXXXXXXXX" (IN OCTAL).

```

```

5457 ;*WITH LEADING ZEROS SUPPRESSED.
5458 ;*****
5459 CRTSER: JSR PC,INITSS ;CLEAR S.S. AND PARAMETERS
5460 BICB #B.CFMT,P.CSIH(R5) ;SET 22 SECTOR FORMAT
5461 TSTB FORMAT ;CHECK THE ACTUAL FORMAT
5462 BEQ 4$ ;BR IF 22 SECTORS
5463 INCB P.SECT(R5) ;IF 20 SECTORS, READ SECTOR 1
5464 MOV #RDATA,P.CMND(R5) ;SET READ COMMAND
5465 MOV LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)/1456(OCT)
5466 MOVB #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
5467 MOV #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
5468 MOV #2,P.WC(R5) ;SET WORD COUNT TO READ 2 WORDS
5469 JSR PC,DRVCAL ;READ SERIAL NO. IN BSF
5470 TYPE 'CART' ;TYPE "CART."
5471 TYPE 'SERNM' ;TYPE "SER. NO."
5472 MOV #RWBUF,-(SP) ;GET POINTER FOR $DB20
5473 JSR PC,$DB20 ;CONVERT BINARY TO OCTAL
5474 JSR PC,$SSUPRS ;TYPE CART. SERIAL NO. IN OCTAL
5475 TYPE 'SERIAL' ;TYPE (CR) AND (LF)
5476 RTS PC ;RETURN
5477
5478 ;THIS ROUTINE ASKS THE OPERATOR FOR THE DRIVE TYPE AND ALL FUTURE
5479 ;TESTS WILL BE PERFORMED ASSUMING ALL THE DRIVES ARE OF THAT TYPE
5480
5481 GETTYP: TYPE '$CRLF'
5482 TYPE 'ASKTYP' ;ASK DRIVE TYPE
5483 JSR PC,RDCHRS ;READ INPUT STRING
5484 GETTYP ;CONT-C RETURN
5485 GETTYP ;CONT-Z RET
5486 GETTYP ;CONT-U RET
5487 TST R0 ;SEE IF CHAR TYPED
5488 BNE 2$ ;BR IF NO
5489 TYPE 'BAD INPUT' ;ECHO BAD INPUT
5490 TYPE '?' ;TYPE (C)
5491 BR GETTYP ;ASK AGAIN
5492
5493 1$: CMP #6,BUFF0 ;SEE IF 6 TYPED
5494 BNE 3$ ;BR IF NO
5495 CLRB TYPFMT ;SETUP FOR RK06
5496 MOV #632,LSTCYL
5497 MOV #25,HOLD1
5498 MOV #10025,HOLD2
5499 MOV #100,HOLD3
5500 MOV #60,HOLD4
5501 MOV #40,HOLD5
5502 RTS PC
5503 3$: CMP #7,BUFF0 ;SEE IF 7 TYPED
5504 BNE 1$ ;BR IF NO
5505 BISB #B.CDT,TYPFMT ;SETUP FOR RK07
5506 MOV #1456,LSTCYL
5507 MOV #2025,HOLD1
5508 MOV #12025,HOLD2
5509 MOV #200,HOLD3
5510 MOV #160,HOLD4
5511 MOV #100,HOLD5
5512 RTS PC

```

```

5513
5514 024404 000000 LSTCYL: 0
5515 024406 000000 HOLD1: 0
5516 024410 000000 HOLD2: 0
5517 024412 000000 HOLD3: 0
5518 024414 000000 HOLD4: 0
5519 024416 000000 HOLD5: 0
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535 024420 010046 STALL: MOV RO, -(SP) ;SAVE RO
5536 024422 010146 MOV R1, -(SP) ;SAVE R1
5537 024424 032777 000400 154506 BIT #BIT08, RSWR ;APPLY RANDOM STALL ?
5538 024432 001407 BEQ 1$ ;BR IF NOT RANDOM
5539 024434 004737 042252 JSR PC, $RAND ;GENERATE PSEUDO-RANDOM NUMBER
5540 024440 013700 042352 MOV $LONUM, RO ;GET IT INTO RO
5541 024444 006200 ASR RO ;SCALE IT DOWN
5542 024446 006200 ASR RO
5543 024450 000403 BR 2$ ;GO STALL WITH RANDOM NO.
5544 024452 013700 005432 1$: MOV STALLS, RO ;GET REQUESTED NO. OF STALLS
5545 024456 001406 BEQ 6$ ;RETURN IF NO STALL REQUIRED
5546 024460 012701 000016 2$: MOV #14., R1 ;SET CONSTANT FOR 40 US
5547 024464 005301 4$: DEC R1 ;INNER LOOP COUNTER
5548 024466 001376 BNE 4$ ;INNER LOOP BR
5549 024470 005300 DEC RO ;OUTER LOOP COUNTER
5550 024472 001372 BNE 2$ ;OUTER LOOP BR
5551 024474 012601 6$: MOV (SP)+, R1 ;RESTORE R1
5552 024476 012600 MOV (SP)+, RO ;RESTORE RO
5553 024500 000207 RTS PC ;RETURN
5554
5555
5556
5557
5558
5559
5560
5561 024502 104407 LODSEC: SAVREG ;SAVE RO-R5
5562 024504 012701 053342 MOV #RWBUF, R1 ;GET ADRS OF R/W BUF INTO R1
5563 024510 012700 005766 2$: MOV #TABLEB, RO ;GET DATA PATTERN ADDRESS
5564 024514 012703 000020 MOV #16., R3 ;INIT PATTERN COUNTER
5565 024520 012021 4$: MOV (RO)+, (R1)+ ;MOVE A PATTERN WORD
5566 024522 020127 054342 CMP R1, #RWBUF+512. ;SEE IF DONE YET
5567 024526 103003 BHS 9$ ;BR IF DONE
5568 024530 005303 DEC R3 ;DECREMENT PATTERN COUNTER

```

5569 024532 001372
5570 024534 000765
5571 024536 104410
5572 024540 000207

BNE 4\$;BR IF NOT RESTARTING PATTERN YET
BR 2\$;BR TO RESTART PATTERN
9\$: RESREG ;RESTORE RO-R5
RTS PC ;RETURN

5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624

024542 104407
024544 013746 005474
024550 005416
024552 005046
024554 116616 000003
024560 005066 000002
024564 116566 000004 000002
024572 066616 000002
024576 005066 000002
024602 012700 000026
024606 105737 003134
024612 001402
024614 012700 000024
024620 020016
024622 101004
024624 160016
024626 005266 000002
024632 000772
024634 112637 005473
024640 005046
024642 116516 000005
024646 066616 000002
024652 005066 000002
024656 122716 000003
024662 101005
024664 162716 000003
024670 005266 000002
024674 000770
024676 112637 005472
024702 066516 000002
024706 011637 005470
024712 005726
024714 104410
024716 000207

: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.CYL(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.

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

```

5625
5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637 024720 104407
5638 024722 013702 005466
5639 024726 005737 005446
5640 024732 001007
5641 024734 012700 005566
5642 024740 012746 000400
5643 024744 012701 053342
5644 024750 000422
5645 024752 005000
5646 024754 032701 000001
5647 024760 001003
5648 024762 005200
5649 024764 006201
5650 024766 000772
5651 024770 006300
5652 024772 006300
5653 024774 006300
5654 024776 006300
5655 025000 006300
5656 025002 062700 005566
5657 025006 012746 000020
5658 025012 013701 005504
5659 025016 010003
5660 025020 011604
5661 025022 012321
5662 025024 005302
5663 025026 001403
5664 025030 005304
5665 025032 001373
5666 025034 000770
5667 025036 005726
5668 025040 104410
5669 025042 000207
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680

```

```

;*****
;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      #RWBUF,R1     ;SET BUFFER ADDRESS
                BR       30$            ;PROCEED
3$:             CLR      R0             ;INIT PATTERN NUMBER
4$:             BIT      #BIT0,R1      ;SEE IF THIS BIT IS SET
                BNE      6$             ;BR IF THIS BIT IS SET
                INC      R0             ;INCREMENT PATTERN NO.
                ASR      R1             ;SHIFT TO EXAMINE NEXT BIT
                BR       4$            ;BR TO CHECK NEXT BIT
6$:             ASL      R0             ;MULTIPLY PATTERN NO. BY 32(DEC)
                ASL      R0
                ASL      R0
                ASL      R0
                ASL      R0
                ADD      #PAT00,R0      ;GET ADDRESS OF DESIRED PATTERN
                MOV      #16,-(SP)     ;SET PATTERN WORD COUNT
                MOV      PMA,R1        ;SET BUFFER ADDRESS
30$:            MOV      R0,R3          ;GET A COPY OF PATTERN ADDRESS
                MOV      (SP),R4       ;INIT PATTERN WORD COUNT
34$:            MOV      (R3)+,(R1)+   ;LOAD A DATA WORD INTO BUFFER
                DEC      R2            ;DECREMENT WORD COUNTER
                BEQ      44$           ;BR IF ALL DONE
                DEC      R4            ;DECREMENT PATTERN WORD COUNT
                BNE      34$           ;BR IF NOT DONE WITH PATTERN YET
                BR       30$           ;BR TO REPEAT THE PATTERN
44$:            TST      (SP)+         ;POP THE STACK
                RESREG          ;RESTORE R0-R5
                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
;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
;*REPEATING DATA PATTERN TO COMPARE AGAINST.

```

```

5681 025044 104407
5682 025046 112737 000040 051665
5683 025046 112737 000007 053234
5684 025054 012737 000007
5685 025062 013702 005466
5686 025066 005037 005444
5687 025072 004737 031224
5688 025076 005737 005446
5689 025102 001007
5690 025104 012700 005566
5691 025110 012746 000400
5692 025114 012701 053342
5693 025120 000422
5694 025122 005000
5695 025124 032701 000001
5696 025130 001003
5697 025132 005200
5698 025134 006201
5699 025136 000772
5700 025140 006300
5701 025142 006300
5702 025144 006300
5703 025146 006300
5704 025150 006300
5705 025152 062700 005566
5706 025156 012746 000020
5707 025162 013701 005504
5708 025166 010003
5709 025170 011604
5710 025172 022321
5711 025174 001002
5712 025176 000137 025536
5713 025202 105037 051665
5714 025206 012737 000005 053234
5715
5716 025214 010237 001202
5717 025220 163737 005466 001202
5718 025226 013737 001202 005474
5719 025234 004737 024542
5720 025240 104407
5721 025242 013700 005470
5722 025246 113701 005472
5723 025252 113702 005473
5724 025256 004737 030042
5725 025262 104410
5726 025264 032737 001000 005424
5727 025272 001116
5728 025274 005737 005444
5729 025300 001010
5730 025302 104034
5731 025304 012737 177777 001174
5732 025312 005037 001176
5733 025316 005037 001200
5734 025322 005237 005444
5735 025326 032777 000001 153604
5736 025334 001004

;*****
CMPBUF: SAVREG ;SAVE R0-R5
MOV #40,DH701+38. ;RESTORE ERROR MSG PARAMS
MOV #7,DF25+2
MOV WDSXFR,R2 ;SET NO. OF WORDS
CLR SCRACH ;CLEAR COMPARE ERROR COUNT
JSR PC,REPSUP ;STORE PREV CMND FOR POSS. PRINT
TST PATRN ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
BNE 3$ ;BR IF NOT
MOV #PAT00,R0 ;GET DATA PATTERN STARTING ADDR
MOV #256,-(SP) ;SET PATTERN WORD COUNT
MOV #RWBUF,R1 ;SET BUFFER ADDRESS
BR 30$ ;PROCEED
3$: CLR R0 ;INIT PATTERN NO.
4$: BIT #BIT0,R1 ;SEE IF THIS BIT IS SET
BNE 6$ ;BR IF THIS BIT IS SET
INC R0 ;INCREMENT PATTERN NUMBER
ASR R1 ;SHIFT TO EXAMINE NEXT BIT
BR 4$ ;BR TO CHECK NEXT BIT
6$: ASL R0 ;MULTIPLY PATTERN NO. BY 32(DEC)
ASL R0
ASL R0
ASL R0
ASL R0
ADD #PAT00,R0 ;GET ADDRESS OF DESIRED PATTERN
MOV #16,-(SP) ;PATTERN WORD COUNT
MOV PMA,R1 ;SET BUFFER ADDRESS
30$: MOV R0,R3 ;GET COPY OF PATTERN ADDRESS
MOV (SP),R4 ;INIT PATTERN WORD COUNT
34$: CMP (R3)+,(R1)+ ;COMPARE DATA WORD TO PATTERN WORD
BNE 31$ ;BR IF COMPARE ERROR
JMP 40$ ;JUMP IF DATA COMPARES OK
31$: CLRB DH701+38. ;ADJUST DATA HEADER FOR MSG
MOV #5,DF25+2 ;ADJUST ERROR DATA WORD COUNT
;COMMON COMPARE ERROR HANDLER
35$: MOV R2,$REG10 ;GET WORD NO.
SUB WDSXFR,$REG10 ;GET 2'S COMP OF WORD NO.
MOV $REG10,LASTWC ;COMPUTE ACTUAL PACK ADRS
JSR PC,FINADR ;SAVE R0-R5
SAVREG ;GET CYL
MOV FINCYL,R0 ;GET TRACK
MOV FINTRK,R1 ;GET SECTOR
MOV FINSEC,R2 ;SEE IF THIS SECTOR LISTED BAD
JSR PC,BDSRCK ;RESTORE R0-R5
RESREG
BIT #BADSEC,RECODE ;BR IF LISTED- DON'T REPORT ERROR
BNE 50$ ;CHECK THE ERROR COUNT
TST SCRACH ;BR IF THIS IS NOT FIRST ERROR
BNE 36$ ;TYPE HEADING FOR ERROR MSG
ERROR 34 ;INIT CYL NO.
46$: MOV #-1,$REG5
48$: CLR $REG6
CLR $REG7
36$: INC SCRACH ;INCREMENT THE ERROR COUNT
BIT #BIT0,DSWR ;SEE IF ALL ERRORS SHOULD BE REPORTED
BNE 38$ ;BR TO REPORT ALL ERRORS

```

```

5737 025336 022737 000012 005444      CMP      #10.,SCRACH      ;SEE IF 10(DEC) ERRORS YET
5738 025344 002504                      BLT      54$           ;BR IF ERROR LIMIT EXCEEDED
5739 025346 023737 001174 005470 38$:  CMP      $REG5,FINCYL      ;SEE IF DIFFERENT CYL
5740 025354 001010                      BNE      42$           ;BR IF YES
5741 025356 123737 001176 005472      CMPB     $REG6,FINTRK      ;SEE IF DIFFERENT TRACK
5742 025364 001004                      BNE      42$           ;BR IF YES
5743 025366 123737 001200 005473      CMPB     $REG7,FINSEC      ;SEE IF DIFFERENT SECTOR
5744 025374 001412                      BEQ      44$           ;BR IF SAME PACK ADDRESS
5745 025376 013737 005470 001174 42$:  MOV      FINCYL,$REG5      ;SET NEW PACK ADRS FOR PRINTOUT
5746 025404 113737 005472 001176      MOVB     FINTRK,$REG6
5747 025412 113737 005473 001200      MOVB     FINSEC,$REG7
5748 025420 104115                      ERROR     11$           ;TYPE NEW PACK ADDRESS
5749 025422 005437 001202 44$:  NEG      $REG10          ;GET WORD NO.
5750 025426 016337 177776 001204      MOV      -2(R3),$REG11      ;GET GOOD DATA
5751 025434 016137 177776 001206      MOV      -2(R1),$REG12      ;GET BAD DATA
5752 025442 013737 001202 001212      MOV      $REG10,$REG14      ;COMPUTE PHYSICAL ADDRESS
5753 025450 005037 001210                      CLR      $REG13
5754 025454 006137 001212                      ROL      $REG14
5755 025460 006137 001210                      ROL      $REG13
5756 025464 063737 005504 001212      ADD      PMA,$REG14
5757 025472 005537 001210                      ADC      $REG13
5758 025476 063737 005506 001210      ADD      PMA+2,$REG13
5759 025504 010137 001214                      MOV      R1,$REG15          ;GET VIRT. ADRS FOR PRINTOUT
5760 025510 162737 000002 001214      SUB      #2,$REG15
5761 025516 104063                      ERROR     63$           ;TYPE GOOD AND BAD DATA, MEM. ADRS.
5762 025520 032777 000100 153412      BIT      #BIT6,$SWR          ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
5763 025526 001013                      BNE      54$           ;BR IF JUST 1 ERROR SHOULD BE REPORTED
5764 025530 005737 005446 50$:  TST      PATRN          ;SEE IF DEFAULT DATA TEST
5765 025534 001400                      BEQ      40$           ;BR IF YES
5766 025536 005302 40$:  DEC      R2              ;DECREMENT WORD COUNTER
5767 025540 001406                      BEQ      54$           ;BR IF ALL DONE
5768 025542 005304                      DEC      R4              ;DECREMENT PATTERN WORD COUNT
5769 025544 001002                      BNE      53$           ;BR IF NOT DONE WITH PATTERN YET
5770 025546 001137 025166                      JMP      30$           ;JUMP TO REPEAT THE PATTERN
5771 025552 000137 025172 53$:  JMP      34$
5772 025556 54$:
5773 025556 005726 56$:  TST      (SP)+          ;POP THE STACK
5774 025560 104410      RESREG      ;RESTORE R0-R5
5775 025562 000207      RTS      PC          ;RETURN
5776
5777
5778
5779
5780 ;*****
5781 ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
5782 ;* IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVIST. IF (↑Z)
5783 ;* WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
5784 ;* A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
5785 ;* IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
5786 ;* NO ACTION IS TAKEN.
5787 ;* CALL - JSR PC,CTLOUT
5788 ;* OR - CKEXIT
5789 ;*****
5789 025564 122737 000001 003127 CTLOUT: CMPB     #1,TSTING      ;SEE IF CURRENTLY RUNNING TESTS
5790 025572 001075                      BNE      10$           ;BR IF NOT RUNNING TESTS
5791 025574 005737 005442                      TST      INCHR          ;SEE IF ANY TTY INPUT
5792 025600 001472                      BEQ      10$           ;BR IF NO INPUT

```

```

5793 025602 105737 003126      TSTB      MDFLAG      ;SEE IF DEFAULT MODE RUN
5794 025606 001441      BEQ        12$      ;BR IF YES
5795 025610 122737 000003 005442  CMPB      #003,INTCHR ;SEE IF (↑C) TYPED
5796 025616 001026      BNE        4$      ;BR IF NOT (↑C)
5797 025620 012700 013002      MOV        #ALLSYS,RO ;SET RETURN ADDR = ALLSYS
5798 025624      2$:
5799 025624 000005      6$:      RESET      ;RESET ALL DEVICES
5800 025626 005037 005442      CLR        INTCHR    ;CLEAR TTY CHAR BUFFER WORD
5801 025632 105037 003127      CLRB      TSTING    ;CLEAR TTY ESCAPE ALLOW FLAG
5802 025636 012737 031504 003056  MOV        #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADDRESS
5803 025644 012706 001100      MOV        #STACK,SP ;RESET THE STACK
5804 025650 112765 000113 000001  MOVB      #RECAL,P.CMND(R5) ;SET RECAL COMMAND
5805 025656 004737 030146      JSR        PC,DRVCAL ;DO CLEANUP RECALIBRATE
5806 025662 105037 003143      CLRB      TSTTYP   ;CLEAR OFFSET TEST IDENTIFIER
5807 025666 005037 001102      CLR        STSTNM  ;CLEAR THE TEST NO.
5808 025672 000110      JMP        2RO      ;EXIT FROM TESTS
5809 025674 122737 000032 005442 4$:  CMPB      #032,INTCHR ;SEE IF (↑Z) TYPED
5810 025702 001021      BNE        7$      ;BR IF NOT (↑Z)
5811 025704 012700 013446      MOV        #ASKSYS,RO ;SET RETURN ADDR = ASKSYS
5812 025710 000745      BR         2$      ;TAKE EXIT
5813 025712 122737 000003 005442 12$:  CMPB      #003,INTCHR ;SEE IF (↑C) TYPED
5814 025720 001012      BNE        7$      ;BR IF NOT
5815 025722 104401 011567      TYPE      ,HLTRQD  ;TYPE "HALT REQUESTED"
5816 025726 104401 011537      TYPE      ,CNTRDY   ;TYPE "PRESS CONT WHEN RDY"
5817 025732 012700 033542      MOV        #HLTPRG,RO ;SET HALT ADDRESS
5818 025736 112737 000061 003155  MOVB      #'1,PASSNO  ;RESTORE PASS NO. TO 1
5819 025744 000727      BR         2$      ;TAKE EXIT
5820 025746 122737 000007 005442 7$:  CMPB      #007,INTCHR ;SEE IF (↑G) TYPED
5821 025754 001002      BNE        8$      ;BR IF NOT (↑G)
5822 025756 004737 021136      JSR        PC,GTSWRG  ;OPEN SOFTWARE SWR FOR MODIFICATION
5823 025762 004737 020774      JSR        PC,PREPKB ;ENABLE KBD INPUT AGAIN
5824 025766 000207      8$:      RTS        PC      ;RETURN
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834 025770 104407      ;*****
5835      ;WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO THE DISK.
5836      ;THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PACK
5837      ;ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
5838      ;USED.
5839      ;*****
5840      WRTSEC: SAVREG      ;SAVE RO-R5
5841      ;LOAD THE R/W BUFFER WITH DATA
5842      MOV        #RWBUF,RO ;BUFFER ADDRESS
5843      MOV        #400,R1   ;400(OCT) WORDS
5844      4$:      MOV        R3,(R0)+ ;LOAD A BUFFER WORD
5845      DEC        R1        ;DECR COUNTER
5846      BNE        4$      ;BR IF NOT DONE YET
5847      ;SET UP PARAMETERS
5848      MOV        #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
5849      MOV        #-400,P.WC(R5) ;SET WORD COUNT
5850      ;WRITE THE DATA
5851      MOVB      #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
5852      JSR        PC,DRVCAL ;WRITE THE DATA
5853      RESREG     ;RESTORE RO-R5
5854      RTS        PC      ;RETURN

```

5849
5850
5851
5852
5853
5854
5855
5856
5857
5858 026042 104407
5859 026044 005004
5860 026046 012703 000001
5861 026052 000403
5862 026054 104407
5863 026056 012704 177777
5864 026062 013702 005466
5865 026066 013701 005504
5866 026072 010321
5867 026074 005704
5868 026076 001001
5869 026100 005203
5870 026102
5871 026102 005302
5872 026104 001372
5873 026106 104410
5874 026110 000207
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886 026112 104407
5887 026114 005004
5888 026116 012703 000001
5889 026122 000403
5890 026124 104407
5891 026126 012704 177777
5892 026132 013702 005466
5893 026136 013701 005504
5894 026142 005037 005444
5895 026146 112737 000040 051665
5896 026154 012737 000007 053234
5897 026162 004737 031224
5898 026166 020321
5899 026170 001552
5900
5901 026172 010237 001202
5902 026176 163737 005466 001202
5903 026204 013737 001202 005474
5904 026212 004737 024542

```

*****
;LDMEM1 - LOAD MEMORY WITH INCREASING NUMBERS, STARTING WITH 1.
;LDMEM2 - LOAD MEMORY WITH REPEATED WORD, IN R3 ON ENTRY.
;BOTH SUBROUTINES REQUIRE MEMORY ADDRESS IN PMA, PMA+2, AND WORD COUNT
;MUST BE IN WDSXFR.
*****
LDMEM1: SAVREG                ;SAVE R0-R5
        CLR      R4           ;R4=0 INDICATES LDMEM1
        MOV      #1,R3        ;INIT DATA WORD TO 1
        BR       LDMO         ;PROCEED
LDMEM2: SAVREG                ;SAVE R0-R5
        MOV      #-1,R4        ;R4=177777 INDICATES LDMEM2
LDMO:   MOV      WDSXFR,R2     ;GET NO. OF WORDS
        MOV      PMA,R1        ;GET MEM ADRS
6$:     MOV      R3,(R1)+      ;LOAD A WORD INTO BUFFER
        TST      R4            ;SEE WHICH DATA DESIRED
        BNE      8$            ;BR IF NOT INCREMENTING DATA
        INC      R3            ;INCREMENT THE DATA
8$:
10$:    DEC      R2             ;DECREMENT COUNTER
        BNE      6$            ;BR IF NOT DONE LOADING YET
12$:    RESREG                ;RESTORE R0-R5
        RTS      PC            ;RETURN
*****
;CKMEM1 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH INCREASING
;NUMBERS, STARTING WITH 1.
;CKMEM2 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH REPEATED DATA WORD
;IN R3 ON ENTRY.
;BOTH SUBROUTINES REQUIRE MEM ADRS IN PMA, PMA+2, AND WORD COUNT MUST BE
;IN WDSXFR.
*****
CKMEM1: SAVREG                ;SAVE R0-R5
        CLR      R4           ;R4=0 INDICATES CKMEM1
        MOV      #1,R3        ;INIT DATA TO 1
        BR       CKMO         ;PROCEED
CKMEM2: SAVREG                ;SAVE R0-R5
        MOV      #-1,R4        ;R4=177777 INDICATES CKMEM2
CKMO:   MOV      WDSXFR,R2     ;GET NO. OF WORDS
        MOV      PMA,R1        ;GET MEM ADRS
        CLR      SCRACH
        MOV      #40,DH701+38. ;RESTORE ERROR MSG PARAMS
        MOV      #7,OF25+2
        JSR      PC,REPSUP
6$:     CMP      R3,(R1)+      ;STORE PREV CMND FOR POSS. PRINT
        BEQ      16$          ;COMPARE A DATA WORD
        ;BR IF DATA COMPARES OK
        ;COMPARE ERROR HANDLER
        MOV      R2,$REG10
        SUB      WDSXFR,$REG10 ;GET WORD NO.
        MOV      $REG10, LASTWC ;GET 2'S COMP OF WORD NO.
        JSR      PC,FINADR     ;COMPUTE ACTUAL PACK ADRS

```

Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

5961 026536 000207          RTS      PC          ;RETURN
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971 000540
5972 026540 004737 021170          JSR      PC,INITSS      ;INIT THE S.S.
5973 026544 142765 000020 000007  BICB      #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
5974 026552 112765 000121 000001  MOV      #RDATA,P.CMND(R5) ;SET READ DATA COMMAND
5975 026560 013765 024404 000002  MOV      LSTCYL,P.CYLN(R5)      ;SET CYL = 632/1456
5976 026566 112765 000002 000005  MOV      #2,P.TRACK(R5)      ;SET TRACK = 2
5977 026574 012703 000012          MOV      #10,R3      ;SET FACTORY BSF SECTOR LIMIT
5978 026600 105737 003134          TSTB      FORMAT
5979 026604 001403          BEQ      6$      ;BR IF 22 SECTORS
5980 026606 105265 000004          INCB      P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
5981 026612 005203          INC      R3
5982 026614 012765 177400 000012 6$:  MOV      #-400,P.WC(R5)      ;SET WORD COUNT FOR 1 SECTOR
5983 026622 012765 004162 000010  MOV      #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
5984 026630 012737 027004 003056 8$:  MOV      #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
5985 000636 105037 003145          CLR      WCEFLG      ;INIT THE ERROR FLAG
5986 000642 004737 030146          JSR      PC,DRVCAL      ;READ THE FACTORY BSF
5987 026646 105737 003145          TSTB      WCEFLG      ;SEE IF ANY ERRORS
5988 026652 001413          BEQ      12$      ;BR IF NOT
5989 026654 062765 000002 000004  ADD      #2,P.SECT(R5)      ;CHECK NEXT SECTOR
5990 026662 126503 000004          CMP      P.SECT(R5),R3      ;SEE IF LIMIT EXCEEDED YET
5991 026666 001360          BNE      8$      ;BR IF NOT YET
5992 026670 004737 031224          JSR      PC,REPSUP      ;GATHER STATUS FOR PRINTOUT
5993 026674 104120          ERROR      120      ;ABORTING- BAD BSF READ
5994 026676 000137 033542          JMP      HLTPRG      ;HALT- CAN'T PROCEED
5995 026702 012765 003162 000010 12$: MOV      #BSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
5996 026710 110365 000004          MOV      R3,P.SECT(R5)      ;SET STARTING SECTOR NO.
5997 026714 012703 000026          MOV      #22,R3      ;SET 22 SECTOR LIMIT
5998 026720 105737 003134          TSTB      FORMAT      ;SEE IF 22 SECTOR FORMAT
5999 026724 001402          BEQ      14$      ;BR IF YES
6000 026726 012703 000023          MOV      #19,R3      ;SET 20 SECTOR LIMIT
6001 026732 012737 027004 003056 14$: MOV      #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
6002 026740 105037 003145          CLR      WCEFLG      ;INIT THE ERROR FLAG
6003 026744 004737 030146          JSR      PC,DRVCAL      ;READ THE SOFTWARE BSF
6004 026750 105737 003145          TSTB      WCEFLG      ;SEE IF ANY ERRORS
6005 026754 001407          BEQ      16$      ;BR IF NOT
6006 026756 062765 000002 000004  ADD      #2,P.SECT(R5)      ;CHECK NEXT SECTOR
6007 026764 126503 000004          CMP      P.SECT(R5),R3      ;SEE IF LIMIT EXCEEDED YET
6008 026770 003760          BLE      14$      ;BR IF NOT YET
6009 026772 000736          BR      10$      ;REPORT ERROR AND ABORT
6010 026774 012737 031504 003056 16$: MOV      #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADRS
6011 027002 000207          RTS      PC          ;RETURN
6012
6013
6014
6015 027004 032765 130600 000034  ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
6016 027012 001002          ;*POSSIBLE ERRORS IN READING BAD SECTORS.
        BDSCHD: BIT      #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
        BNE      4$      ;BR IF A READ ERROR OCCURRED

```

6017 027014 000137 031504
6018 027020 105237 003145
6019 027024 000137 033754

4\$: JMP ERRHDL ;GO HANDLE OTHER ERROR
INCB WCEFLG ;SET ERROR FLAG
JMP RETNML ;TAKE NORMAL DRIVER RETURN

6020
6021
6022
6023
6024
6025
6026
6027

;TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
;WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
;LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
;2 BSF'S.

6028
6029 027030 104407
6030 027032 005004
6031 027034 012700 004172
6032 027040 104401 011762
6033 027044 104401 012007
6034 027050 104401 051432
6035 027054 104401 001325
6036 027060 005001
6037 027062 005710
6038 027064 100007
6039 027066 005701
6040 027070 001032
6041 027072 104401 012027
6042 027076 104401 001325
6043 027102 000425
6044 027104 012046
6045 027106 104402
6046 027110 104401 012147
6047 027114 011003
6048 027116 105003
6049 027120 000303
6050 027122 010346
6051 027124 104402
6052 027126 104401 012147
6053 027132 111003
6054 027134 010346
6055 027136 104402
6056 027140 104401 001325
6057 027144 005720
6058 027146 005201
6059 027150 020127 000176
6060 027154 002742
6061 027156 005704
6062 027160 001006
6063 027162 005204
6064 027164 012700 003172
6065 027170 104401 011775
6066 027174 000725
6067 027176 104410
6068 027200 000207
6069
6070
6071
6072

TYPBSF: SAVREG ;SAVE R0-R5
CLR R4 ;INIT BSF FILE INDICATOR
MOV #BSFACT+10,R0 ;ADRS OF DATA IN FACTORY BSF
TYPE ,FACTBS ;TYPE "FACTORY"
TYPE ,BDSECT ;TYPE "BAD SECTORS"
3\$: TYPE ,DH6041 ;TYPE "CYLNDR TRACK SECTOR"
TYPE ,\$CRLF
CLR R1 ;INIT LIMIT COUNTER
4\$: TST (R0) ;SEE IF A SECTOR LISTED HERE
BPL BS ;BR IF YES
TST R1 ;SEE IF FILE IS EMPTY
BNE 14\$;BR IF NOT EMPTY
TYPE ,NOFALS ;TYPE "NONE"
TYPE ,\$CRLF ;TYPE <CR>,<LF>
BR 14\$
8\$: MOV (R0)+,-(SP) ;GET A CYL NO.
TYP0C ;TYPE IT
TYPE ,SPACE2 ;TYPE SEPARATORS
MOV (R0),R3 ;GET TRACK AND SECTOR
CLAB R3
SWAB R3 ;GET THE TRACK NO.
MOV R3,-(SP)
TYP0C ;TYPE IT
TYPE ,SPACE2 ;TYPE SEPARATORS
MOVB (R0),R3 ;GET SECTOR NO.
MOV R3,-(SP)
TYP0C ;TYPE IT
TYPE ,\$CRLF
TST (R0)+ ;INCR THE POINTER
INC R1 ;INCR THE COUNTER
CMP R1,#126. ;SEE IF LIMIT REACHED IN THIS FILE
BLT 4\$;BR IF NOT YET
14\$: TST R4 ;SEE IF BOTH FILES TYPED YET
BNE 18\$;BR IF YES
INC R4 ;SET INDICATOR FOR SOFT. FILE
MOV #BSOFT+10,R0 ;ADRS OF DATA IN SOFTWARE BSF
TYPE ,SOFTBS ;TYPE "SOFTWARE BAD SECTORS :"
BR 3\$;GO TYPE SOFT. BSF
18\$: RESREG ;RESTORE R0-R5
RTS PC ;RETURN

```

6073
6074
6075
6076 027202 104407
6077 027204 016500 000002
6078 027210 116501 000005
6079 027214 116502 000004
6080 027220 005003
6081 027222 026500 000030
6082 027226 001006
6083 027230 126501 000027
6084 027234 001003
6085 027236 126502 000026
6086 027242 001404
6087 027244 005203
6088 027246 004737 027266
6089 027252 000763
6090 027254 000303
6091 027256 010337 005466
6092 027262 104410
6093 027264 000207
6094
6095
6096
6097
6098
6099
6100
6101
6102 027266 012704 000025
6103 027272 105737 003134
6104 027276 001402
6105 027300 012704 000023
6106 027304 005202
6107 027306 020204
6108 027310 003407
6109 027312 005002
6110 027314 005201
6111 027316 020127 000002
6112 027322 003402
6113 027324 005001
6114 027326 005200
6115 027330 000207
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126 027332 104407
6127 027334 004737 027202
6128 027340 016500 000030

```

```

; *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.
        MOV             P.TRACK(R5),R1 ;GET TRACK NO.
        MOV             P.SECT(R5),R2 ;SECTOR NO.
        CLR             R3             ;INIT SECTOR COUNT TO 0
6$:      CMP             P.DCYL(R5),R0 ;COMPARE ORIG. AND CURRENT CYLS
        BNE             B$            ;BR IF NOT EQUAL
        CMPB            P.DTS+1(R5),R1 ;COMPARE TRACKS
        BNE             B$            ;BR IF NOT EQUAL
        CMPB            P.DTS(R5),R2   ;COMPARE SECTORS
        BEQ             12$           ;BR IF ADRS ARE EQUAL
8$:      INC             R3            ;INCR SECTOR COUNT
        JSR             PC,INCRSC     ;INCR PACK ADRS BY 1 SECTOR
        BR              B$            ;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, P.BALO(R5),
; *P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
; *TERMINATED BY A BAD SECTOR ERROR (BSE), A DATA CHECK ERROR (DCK),
; *OR A HEADER VRC ERROR (HVRC).
; *****
MIDXFR: SAVREG          ;SAVE R0-R5
        JSR             PC,FINMEM     ;COMPUTE NO. OF WORDS XFERRED
        MOV             P.DCYL(R5),R0 ;GET CYL NO.

```

Address	Instruction	Comments
6129	027344 116501 000027	MOV B P.DTS+1(R5),R1 ; GET TRACK NO.
6130	027350 116502 000026	MOV B P.DTS(R5),R2 ; GET SECTOR NO.
6131	027354 013703 005466	MOV WDSXFR,R3 ; GET NO. OF WORDS TRANSFERRED
6132	027360 032737 000020	BIT #DCKERR,RECODE ; SEE IF DATA CHECK ERROR OCCURRED
6133	027366 001004	BNE 9\$; BR IF YES
6134	027370 062703 000400	ADD #400,R3 ; SKIP BAD SECTOR
6135	027374 004737 027266	JSR PC,INCRSC ; INCREMENT PACK ADRS TO NEXT SECTOR
6136	027400 010065 000002	9\$: MOV R0,P.CYLN(R5) ; UPDATE CYLINDER
6137	027404 110165 000005	MOV B R1,P.TRACK(R5) ; UPDATE TRACK
6138	027410 110265 000004	MOV B R2,P.SECT(R5) ; UPDATE SECTOR
6139	027414 060365 000012	ADD R3,P.WC(R5) ; UPDATE P.WC(R5)
6140	027420 032765 100000	000014: BIT #DIBAI1,P.PRST(R5) ; SEE IF BUS ADRS INCR INHIB SET
6141	027426 001022	BNE 16\$; BR IF YES
6142	027430 005004	CLR R4 ; GET BYTES XFERRED
6143	027432 006103	ROL R3
6144	027434 006104	ROL R4
6145	027436 060337 005504	ADD R3,PMA ; UPDATE PMA,PMA+2
6146	027442 005537 005506	ADC PMA+2
6147	027446 060437 005506	ADD R4,PMA+2
6148	027452 013765 005504	000010: MOV PMA,P.BALO(R5) ; UPDATE P.BALO(R5)
6149	027460 013700 005506	MOV PMA+2,R0
6150	027464 042700 177774	BIC #177774,R0
6151	027470 150065 000007	BISB R0,P.BAHI(R5) ; UPDATE P.BAHI(R5)
6152	027474 104410	16\$: RESREG ; RESTORE R0-R5
6153	027476 000207	RTS PC ; RETURN
6154		
6155		
6156		
6157		
6158		*****
6159		*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
6160		*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
6161	027500 104407	*****
6162	027502 012700 002642	SVPRMS: SAVREG ; SAVE R0-R5
6163	027506 012701 005454	MOV #PARMO+2,R0 ; ADRS OF PARAMS
6164	027512 016537 000012	005466: MOV #SAVPRS,R1 ; ADRS OF SAVE AREA
6165	027520 005437 005466	MOV P.WC(R5),WDSXFR ; GET WORD COUNT
6166	027524 012737 053342	005504: NEG WDSXFR ; MAKE IT POSITIVE
6167	027532 005037 005506	MOV #RWBUF,PMA ; INIT PMA TO RWBUF
6168	027536 000405	CLR PMA+2 ; INIT PMA+2 TO 0
6169	027540 104407	BR GTD ; CONTINUE
6170	027542 012700 005454	GTPRMS: SAVREG ; SAVE R0-R5
6171	027546 012701 002642	MOV #SAVPRS,R0 ; ADRS OF SAVE AREA
6172	027552 012702 000005	MOV #PARMO+2,R1 ; ADRS OF PARAMS
6173	027556 012021	6\$: MOV #5,R2 ; SET FOR 5 WORDS
6174	027560 005302	MOV (R0)+,(R1)+ ; MOVE A WORD
6175	027562 001375	DEC R2 ; SEE IF DONE YET
6176	027564 104410	BNE 6\$; BR IF NOT YET
6177	027566 000207	RESREG ; RESTORE R0-R5
6178		RTS PC ; RETURN
6179		
6180		
6181		*****
6182		*TRANSR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS
6183		*OR DATA CHECK ERRORS, IF ENCOUNTERED.
6184		*THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-

```

6185 ;*ROUTINE IS SUBSTITUTED FOR DRVCL WHEN BAD SECTORS MUST BE HANDLED.
6186 ;*AND IT IS ALSO USED TO HANDLE THE DATA CHECK ERRORS CAUSED
6187 ;*BY READING DATA WITH OFFSETS, IN THE OVERWRITE AND DATA COMPATIBILITY
6188 ;*TESTS.
6189 ;*****
6190 TRANSFR: MOV R4, -(SP) ;SAVE R4 ON STACK
6191 CLR R4 ;CLEAR ERROR INDICATOR
6192 CLRB DCEFLG ;CLEAR DCK ERROR FLAG
6193 4$: CLR RECODE ;CLEAR ERROR FLAGS
6194 MOV #DCKHDL, A.ABNL ;SET DCK ERROR HANDLER ADRS
6195 CLRB ERRCNT ;CLEAR ERROR RETRY COUNT
6196 JSR PC, DRVCL ;PERFORM THE S.S. FUNCTION
6197 BIT #BSERR!DCKERR!HVRCR, RECODE ;SEE IF BSE OR DCK OR HVRC ERROR OCCURRED
6198 BEQ 6$ ;BR IF NOT
6199 INC R4 ;INCR ERROR INDICATOR
6200 JSR PC, CKSCOR ;DECR. DRIVE SCORE, IF NECESSARY
6201 JSR PC, MIDXFR ;UPDATE PARAMS TO RESUME XFER
6202 TST P.WC(R5) ;SEE IF ENTIRE XFER IS COMPLETED
6203 BNE 4$ ;BR IF NOT
6204 6$: TST R4 ;SEE IF ANY ERRORS OCCURRED
6205 BEQ 8$ ;BR IF NOT
6206 JSR PC, GTPRMS ;RESTORE ORIGINAL PARAMS OF XFER
6207 JSR PC, SVPRMS ;RESTORE WDSXFR, PMA, PMA+2
6208 8$: MOV (SP)+, R4 ;RESTORE R4
6209 MOV #ERRHDL, A.ABNL ;RESTORE NORMAL ERROR HANDLER ADRS
6210 PC ;RETURN
6211
6212 ;*****
6213 ;SBTTL SEARCH BAD SECTOR TABLES ROUTINE
6214 ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
6215 ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
6216 ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
6217 ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
6218 ;*
6219 ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
6220 ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
6221 ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
6222 ;*****
6223 SRTBS: MOV R2, $TMP2
6224 MOV R3, $TMP3
6225 MOV (SP)+, $TMP4 ;STORE RETURN CONTENTS OF R4
6226 MOV (R4), R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
6227 MOV (R4)+, $TMP1 ;SETUP FOR LENGTH OF BAD SECTOR TABLE
6228 ADD #1000, $TMP1
6229 CLRB R3 ;CLEAR R3 FOR COUNT
6230 ADD #10, R2 ;SET R2 FOR POINTER TO CYLINDER ENTRY
6231 1$: TST (R2) ;TEST IF ALL ONES
6232 BMI 5$ ;YES-DONE
6233 CMP R0, (R2) ;TEST IF BAD SECTOR IN PRESENT CYL
6234 BEQ 3$ ;YES-GO CHECK TRACK
6235 ADD #4, R2 ;ELSE BUMP POINTER
6236 CMP R2, $TMP1 ;TEST IF OUT OF TABLE
6237 BGE 5$ ;YES-EXIT
6238 BR 1$ ;LOOP
6239
6240

```

```

6241 027762 005722 3$: TST (R2)+ ; TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
6242 027764 011237 001302 MOV (R2), $TMP10 ; GET TRK/SEC WORD
6243 027770 042737 174377 001302 BIC #174377, $TMP10 ; CLEAR ALL BUT TRACK
6244 027776 123701 001303 CMPB $TMP10+1, R1 ; CHECK IF BAD SECTOR IN THIS TRACK
6245 030002 001402 BEQ 4$ ; YES - GO PUT SECTOR NUMBER ON STACK
6246 030004 005722 TST (R2)+ ; ELSE BUMP POINTER TO NEXT CYL WORD
6247 030006 000753 BR 1$ ; GO TEST NEXT CYL
6248 030010 005203 4$: INC R3 ; BUMP BAD SECTOR COUNT
6249 030012 012246 MOV (R2)+, -(SP) ; PUT TRK/SEC WORD ON STACK
6250 030014 042716 177700 BIC #177700, (SP) ; CLEAR ALL BUT SECTOR NUMBER
6251 030020 000746 BR 1$ ; GO CHECK REST OF FILE
6252 030022 010346 5$: MOV R3, -(SP) ; EXIT - PUT NUMBER OF BAD
6253 030024 013702 MOV $TMP2, R2 ; SECTORS ON STACK-RESTORE
6254 030030 013703 MOV $TMP3, R3 ; REGISTERS
6255 030034 013746 MOV $TMP4, -(SP) ; PUT RETURN ON STACK
6256 030040 000204 RTS R4 ; RETURN
6257 *****
6258 ; SBTTL BAD SECTOR CHECK ROUTINE
6259 ; *THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
6260 ; *SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
6261 ; *TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
6262 ; *IS NOT LISTED THE FLAG IS RESET.
6263 *****
6264 030042 104407 BDSRCK: SAVREG
6265 030044 012737 004162 030056 MOV #BSFACT, 1$ ; SET TABLE TO SEARCH
6266 030052 004437 027700 JSR R4, SRHTBS ; GO SEARCH IT
6267 030056 000000 1$: .WORD ; TABLE ADDRESS GOES HERE
6268 030060 012603 MOV (SP)+, R3 ; GET NUMBER OF BAD SECTORS, IF ANY
6269 030062 001015 BNE 6$ ; IF ANY, GO TEST WHICH ONES
6270 030064 023727 030056 003162 7$: CMP 1$, #BSSOFT ; ELSE TEST IF OTHER TABLE ALREADY SEARCHED
6271 030072 001404 BEQ 3$ ; IF YES-EXIT
6272 030074 012737 003162 030056 MOV #BSSOFT, 1$ ; SET OTHER TABLE FOR SEARCH
6273 030102 000763 BR 2$ ; GO SEARCH IT
6274 030104 042737 001000 005424 3$: BIC #BADSEC, RECODE ; CLEAR BAD SECTOR BIT
6275 030112 104410 4$: RESREG
6276 030114 000207 RTS PC ; RETURN
6277 030116 022602 6$: CMP (SP)+, R2 ; THIS SECTOR IN TABLE?
6278 030120 001403 BEQ 8$ ; YES-GO SET BIT & EXIT
6279 030122 005303 DEC R3 ; DECREMENT BAD SECTOR COUNT
6280 030124 001374 BNE 6$ ; IF NOT ZERO-CHECK NEXT ENTRY
6281 030126 000756 BR 7$ ; ELSE GO SEARCH OTHER TABLE
6282 030130 052737 001000 005424 8$: BIS #BADSEC, RECODE ; SET BAD SECTOR BIT
6283 030136 005303 9$: DEC R3 ; CLEAR STACK OF OTHER BAD SECTOR
6284 030140 001764 BEQ 4$ ; NUMBER
6285 030142 005726 TST (SP)+
6286 030144 000774 BR 9$ ; EXIT
6287
6288
6289
6290
6291 *****
6292 ; SBTTL CALL DRIVER ROUTINE
6293 ; *ENTRY JSR PC, DRVCAL
6294 ; * WITH R5 POINTING TO PARAMETER BLOCK
6295 ; *RETURN RTS PC
6296 ; *

```

```

6297 ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
6298 ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
6299 ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
6300 ;*BLOCK WHEN THE ROUTINE IS CALLED.
6301 ;*
6302 ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
6303 ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
6304 ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
6305 ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
6306 ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
6307 ;*****
6308 DRVCAL: JSR PC,STALL ;PERFORM A STALL IF REQUIRED
6309 030146 JSR PC,CTLOUT ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
6310 030152 004737 024420 CLR B DONE ;CLEAR DONE FLAG
6311 030156 105037 003132 CLR B DRNAFG ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
6312 030162 105037 003141 JSR PC,STRCMD ;STORE PREV AND CURRENT COMMANDS
6313 030166 004737 030220 MOV R5,4$ ;GET PARAM BLOCK ADDRESS
6314 030172 010537 030202 JSR PC,C.INIT ;CALL DRIVER
6315 030176 004737 040222 4$: .WORD ;P.B. ADDRESS GOES HERE
6316 030202 000000 6$: JSR PC,W.WTCH ;CALL WATCH DOG
6317 030204 004737 034572 TST B DONE ;DONE SET?
6318 030210 105737 003132 BEQ 6$ ;NO-LOOP
6319 030214 001773 RTS PC ;YES-RETURN
6320 030216 000207
6321
6322
6323 ;*****
6324 ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
6325 ;*****
6326 STRCMD: SAVREG ;SAVE R0-R5
6327 030220 104407 MOV #PRVCMD,R1 ;ADDR OF PREV COMMAND STORAGE
6328 030222 012701 005162 MOV #COMSTR,R0 ;ADDR OF CURRENT COMMAND STORAGE
6329 030226 012700 005176 ;STORE PREVIOUS COMMAND
6330 MOV (R0)+,(R1)+
6331 030232 012021 MOV (R0)+,(R1)+
6332 030234 012021 MOV (R0)+,(R1)+
6333 030236 012021 MOV (R0)+,(R1)+
6334 030240 012021 MOV (R0)+,(R1)+
6335 030242 012021 MOV (R0)+,(R1)+
6336 030244 012021 MOV (R0)+,(R1)+
6337 ;STORE CURRENT COMMAND
6338 030246 012701 005176 4$: MOV #COMSTR,R1 ;R5 POINTS TO DRIVER PARAM BLK
6339 030252 012521 MOV (R5)+,(R1)+
6340 030254 012521 MOV (R5)+,(R1)+
6341 030256 012521 MOV (R5)+,(R1)+
6342 030260 012521 MOV (R5)+,(R1)+
6343 030262 012521 MOV (R5)+,(R1)+
6344 030264 012521 MOV (R5)+,(R1)+
6345 030266 104410 RESREG ;RESTORE R0-R5
6346 030270 000207 RTS PC ;RETURN
6347
6348
6349 ;*****
6350 ;*SBTTL DRIVE ERROR FREE RETURN ROUTINE
6351 ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
6352

```

```

6353 ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
6354 ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
6355 ;*ROUTINE.
6356 ;*****
6357 030272 152737 000377 003132 003132 003132 003132 003132 003132 003132
6358 030300 032737 100000 005424 005424 005424 005424 005424 005424
6359 030306 001403 003135 003135 003135 003135 003135 003135 003135
6360 030310 105037 003135 003135 003135 003135 003135 003135 003135
6361 030314 000413 003135 003135 003135 003135 003135 003135 003135
6362 030316 105737 003135 003135 003135 003135 003135 003135 003135
6363 030322 001410 003135 003135 003135 003135 003135 003135 003135
6364 030324 005037 001174 001174 001174 001174 001174 001174 001174
6365 030330 113737 003135 003135 003135 003135 003135 003135 003135
6366 030336 104101 003135 003135 003135 003135 003135 003135 003135
6367 030340 105037 003135 003135 003135 003135 003135 003135 003135
6368 030344 005037 005424 005424 005424 005424 005424 005424 005424
6369 030350 000207 005424 005424 005424 005424 005424 005424 005424
6370 ;*****
6371 ;SBTTL TYPE ERROR ROUTINE
6372 ;*ENTRY - JSR PC,TYPERR
6373 ;*RETURN - RTS PC
6374 ;*
6375 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
6376 ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
6377 ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
6378 ;*THE ERROR.
6379 ;*****
6380 030352 104407 003137 003137 003137 003137 003137 003137 003137
6381 030354 105237 003137 003137 003137 003137 003137 003137 003137
6382 030360 032777 020000 150552 150552 150552 150552 150552 150552
6383 030366 001402 030774 030774 030774 030774 030774 030774 030774
6384 030370 000137 030774 030774 030774 030774 030774 030774 030774
6385 030374 005000 030774 030774 030774 030774 030774 030774 030774
6386 030376 005005 030774 030774 030774 030774 030774 030774 030774
6387 030400 005105 001114 001114 001114 001114 001114 001114 001114
6388 030402 113700 001114 001114 001114 001114 001114 001114 001114
6389 030406 005300 001114 001114 001114 001114 001114 001114 001114
6390 030410 006300 001114 001114 001114 001114 001114 001114 001114
6391 030412 006300 001114 001114 001114 001114 001114 001114 001114
6392 030414 006300 001114 001114 001114 001114 001114 001114 001114
6393 030416 062700 001410 001410 001410 001410 001410 001410 001410
6394 030422 012037 030442 030442 030442 030442 030442 030442 030442
6395 030426 001406 030442 030442 030442 030442 030442 030442 030442
6396 030430 104401 012133 012133 012133 012133 012133 012133 012133
6397 030434 104401 045412 045412 045412 045412 045412 045412 045412
6398 030440 104401 045412 045412 045412 045412 045412 045412 045412
6399 030442 000000 045412 045412 045412 045412 045412 045412 045412
6400 030444 012037 030620 030620 030620 030620 030620 030620 030620
6401 030450 001467 030620 030620 030620 030620 030620 030620 030620
6402 030452 104401 001325 001325 001325 001325 001325 001325 001325
6403 030456 104401 045400 045400 045400 045400 045400 045400 045400
6404 030462 013746 001102 001102 001102 001102 001102 001102 001102
6405 030466 104403 001102 001102 001102 001102 001102 001102 001102
6406 030470 002 001102 001102 001102 001102 001102 001102 001102
6407 030471 000 001102 001102 001102 001102 001102 001102 001102
6408 030472 104401 012133 012133 012133 012133 012133 012133 012133

```

6409	030476	032777	010000	150434	BIT	#BIT12, @SWR	;REPORT DESCRIPTION ONLY ?
6410	030504	001133			BNE	20\$	;BR IF YES
6411	030506	104401	050534		TYPE	,DH105	;TYPE "PREVIOUS COMMAND :"
6412	030512	104401	001325		TYPE	, \$CRLF	
6413	030516	104401	050726		TYPE	,DH101+10	;TYPE PREV COMMAND HEADER
6414	030522	104401	001325		TYPE	, \$CRLF	
6415	030526	012701	000006		MOV	#6, R1	;SIX COMMAND VALUES
6416	030532	012702	001236		MOV	#\$REG26, R2	;STARTING ADDR OF PREV CMND VALUES
6417	030536	012246		30\$:	MOV	(R2)+, -(SP)	;PUT A WORD ON STACK
6418	030540	104402			TYPOC		;TYPE IT
6419	030542	104401	012147		TYPE	,SPACE2	;TYPE SEPARATORS
6420	030546	005301			DEC	R1	;SEE IF 7 VALUES TYPED YET
6421	030550	001372			BNE	30\$	;BR IF NOT
6422	030552	104401	001325		TYPE	, \$CRLF	
6423	030556	104401	012147		TYPE	,SPACE2	;INDENT
6424	030562	104401	051005		TYPE	,DH102	;TYPE HDR FOR BA DATA
6425	030566	104401	001325		TYPE	, \$CRLF	
6426	030572	104401	012147		TYPE	,SPACE2	;INDENT
6427	030576	012246			MOV	(R2)+, -(SP)	
6428	030600	104402			TYPOC		;TYPE PREV. HI BA BITS
6429	030602	104401	012147		TYPE	,SPACE2	
6430	030606	011246			MOV	(R2)+, -(SP)	
6431	030610	104402			TYPOC		;TYPE PREV. LO BA BITS
6432	030612	104401	012133		TYPE	,CR2LF	
6433	030616	104401			TYPE		;TYPE DATA HEADER
6434	030620	000000		4\$:	.WORD	0	;DH POINTER GOES HERE
6435	030622	104401	001325		TYPE	, \$CRLF	
6436	030626	005005			CLR	R5	;INIT INDENT INDICATOR
6437	030630	032777	010000	150302	5\$:	BIT	#BI.12, @SWR
6438	030636	001056			BNE	20\$	;REPORT DESCRIPTION ONLY ?
6439	030640	012001			MOV	(R0)+, R1	;BR IF YES
6440	030642	001454			BEQ	20\$	;GET DT POINTER
6441	030644	012000			MOV	(R0)+, R0	;BRANCH IF THERE ARE NONE
6442	030646	012002			MOV	(R0)+, R2	;GET DF POINTER
6443	030650	112003		10\$:	MOV	(R0)+, R3	;STORE NUMBER OF DH'S
6444	030652	105720			TSTB	(R0)+	;GET & STORE NUMBER OF DATA WORDS
6445	030654	005703			TST	R3	;BUMP PAST FORMAT WORD
6446	030656	001417			BEQ	14\$	;TEST IF ANY DATA FOR THIS HEADER
6447	030660	005705			TST	R5	;NO - SKIP DATA PRINT
6448	030662	001002			BNE	11\$	;SEE IF SHOULD INDENT
6449	030664	104401	012147		TYPE	,SPACE2	;BR IF NOT
6450	030670	013146		11\$:	MOV	(R1)+, -(SP)	;INDENT
6451	030672	104402			TYPOC		;PUT FIRST DATA WORD ON STACK
6452	030674	005303			DEC	R3	;TYPE IT
6453	030676	001403			BEQ	12\$	;MORE DATA WORDS
6454	030700	104401	012147		TYPE	,SPACE2	;NO-BRANCH
6455	030704	000771			BR	11\$	;TYPE SEPARATORS
6456	030706	005702		12\$:	TST	R2	;LOOP
6457	030710	001402			BEQ	14\$	;SEE IF <CR>, <LF> NEEDED
6458	030712	104401	001325		TYPE	, \$CRLF	;BR IF NOT
6459	030716	005302		14\$:	DEC	R2	;TYPE IT
6460	030720	003425			BLE	20\$	;MORE DH'S?
6461	030722	012037	030754	15\$:	MOV	(R0)+, .16\$	;NO-BRANCH
6462	030726	105710			TSTB	(R0)	;GET NEXT DH POINTER
6463	030730	001004			BNE	34\$	;SEE IF ANY DATA FOR HDR
6464	030732	104401	001325		TYPE	, \$RLF	;BR IF YES
							;SKIP EXTRA LINE

```

6465 030736 005005          CLR      R5          ;RE-INIT INDENT INDICATOR
6466 030740 000404          BR       36$         ;
6467 030742 005105          34$: COM      R5          ;COMPLEMENT INDENT INDICATOR
6468 030744 001002          BNE      36$         ;BR IF NO INDENT REQUIRED
6469 030746 104401 012147   TYPE     ,SPACE2     ;INDENT
6470 030752 104401          36$: TYPE     ;TYPE DH
6471 030754 000000          16$: .WORD    0        ;DH POINTER GOES HERE
6472 030756 104401 001325   TYPE     ,$CRLF      ;
6473 030762 105710          TSTB     (R0)         ;TYPE A DT?
6474 030764 001331          BNE      10$         ;YES-BRANCH
6475 030766 062700 000002   ADD      #2,R0       ;INCREMENT OF POINTER
6476 030772 000751          BR       14$         ;SEE IF END OF DF BLOCK
6477 030774 104410          20$: RESREG
6478 030776 000207          RTS      PC
6479
6480 ;*****
6481 ;SBTTL CONTROLLER ERROR REPORTER ROUTINE
6482 ;*ENTRY: JSR PC, CONERR
6483 ;*RETURN: RTS PC
6484 ;
6485 ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
6486 ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
6487 ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
6488 ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
6489 ;*AT THIS TIME.
6490 ;*****
6491 CONERR: SAVREG          ;SAVE R0-R5
6492 ZISB      #377,DONE     ;SET DONE FLAG
6493 INCB      ERRCNT        ;INCREMENT ERROR COUNT
6494 JSR      PC, TOPROC     ;LOAD RK REGS INTO $REGS
6495 BIT      #BIT0,E.CONT   ;ERROR 0?
6496 BEQ      1$            ;NO-BRANCH
6497 ERROR     64            ;CLEAR CONT DID NOT CLEAR ERROR
6498 BR       7$
6499 BIT      #BIT1,E.CONT   ;ERROR 1?
6500 BEQ      2$            ;NO-BRANCH
6501 ERROR     65            ;NO ATTENTION IN ATTENTION SUM REG
6502 BR       7$
6503 BIT      #BIT2,E.CONT   ;ERROR 2?
6504 BEQ      3$            ;NO-BRANCH
6505 TSTB     DRNAFG         ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
6506 BEQ      15$           ;BR IF NOT
6507 INCB      REISSU        ;SET FLAG TO RE-ISSUE COMMAND
6508 ERROR     66            ;UNSOLICITED ATTENTION
6509 BR       7$
6510 BIT      #BIT3,E.CONT   ;ERROR 3?
6511 BEQ      4$            ;NO-BRANCH
6512 ERROR     67            ;UNEXPECTED DATA TYPE ERROR
6513 BR       7$
6514 BIT      #BIT4,E.CONT   ;ERROR 4?
6515 BEQ      5$            ;NO-BRANCH
6516 ERROR     70            ;ATTENTION DID NOT RESET WITH CLEAR
6517 BR       7$
6518 BIT      #BIT5,E.CONT   ;ERROR 5?
6519 BEQ      6$            ;NO-BRANCH
6520 ERROR     71            ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
6521 BR       7$

```

```

6521 031142 032737 000400 003062 6$: BIT #BIT8,E.CONT
6522 031150 001401 BEQ 83
6523 031152 104072 ERROR 72 ;DATA LATE WHEN UNLOADING HEADER
6524 031154 032737 001000 003062 8$: BIT #BIT9,E.CONT
6525 031162 001401 BEQ 95
6526 031164 104073 ERROR 73 ;CONTROLLER ERROR DURING DRIVER SERVICE
6527 031166 032737 002000 003062 9$: BIT #BIT10,E.CONT
6528 031174 001401 BEQ 105
6529 031176 104074 ERROR 74 ;DRIVE DETECTED PARITY ERROR
6530 031200 032737 100000 003062 10$: BIT #BIT15,E.CONT
6531 031206 001401 BEQ 115
6532 031210 104052 ERROR 52 ;MULTIPLE DRIVE SELECT
6533 031212 104075 ERROR 75 ;UNDEFINED ERROR
6534 031214 005037 003062 7$: CLR E.CONT ;CLEAR CONTROLLER ERROR WORD
6535 031220 000137 033560 JMP BGNRTY ;GO DO RETRY
6536 ;*****
6537 ;SBTTL REPORT SUPPORT ROUTINE
6538 ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
6539 ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
6540 ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
6541 REPSUP:
6542 031224 104407 SAVREG
6543 031226 005037 005426 CLR ERRCOM ;CLEAR ERROR COMMAND STORE
6544 031232 116537 000001 005426 MOV P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
6545 031240 012700 001162 MOV #SREG0,R0 ;FOR REPORTING
6546 031244 016520 000002 MOV P.CYLN(R5),(R0)+
6547 031250 116520 000005 MOV P.TRCK(R5),(R0)+
6548 031254 105020 CLRB (R0)+
6549 031256 116520 000004 MOV P.SECT(R5),(R0)+
6550 031262 105020 CLRB (R0)+
6551 031264 016520 000012 MOV P.WC(R5),(R0)+
6552 031270 012700 001174 MOV #SREG5,R0
6553 031274 116503 000007 MOV P.BAHI(R5),R3
6554 031300 042703 177774 BIC #177774,R3
6555 031304 010337 001256 MOV R3,$REG36 ;HI BA BITS
6556 031310 016537 000010 MOV P.BALO(R5),$REG37 ;LO BA BITS
6557 031316 016520 000016 MOV P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
6558 031322 016520 000020 MOV P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
6559 031326 016520 000030 MOV P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
6560 031332 016520 000026 MOV P.OTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
6561 031336 016520 000022 MOV P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
6562 ;FOR ALL REPORTS (TO BE
6563 031342 016520 000024 MOV P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
6564 031346 016520 000032 MOV P.ASOF(R5),(R0)+ ;STORED ANY WAY.
6565 031352 016520 000036 MOV P.DS(R5),(R0)+
6566 031356 016520 000034 MOV P.ER(R5),(R0)+
6567 031362 016520 000040 MOV P.A00(R5),(R0)+
6568 031366 016520 000042 MOV P.B00(R5),(R0)+
6569 031372 016520 000044 MOV P.A01(R5),(R0)+
6570 031376 016520 000046 MOV P.B01(R5),(R0)+
6571 031402 016520 000050 MOV P.A10(R5),(R0)+
6572 031406 016520 000052 MOV P.B10(R5),(R0)+
6573 031412 016520 000054 MOV P.A11(R5),(R0)+
6574 031416 016520 000056 MOV P.B11(R5),(R0)+
6575 ;STORE PREVIOUS COMMAND FOR PRINTOUT
6576 031422 012701 001236 MOV #SREG26,R1 ;ADRS OF PRINT BUF AREA

```

```

6577 031426 012700 005162      MOV      #PRVCMO,R0      ;ADRS OF PREV CMND STORAGE
6578 031432 112021      MOV      (R0)+,(R1)+      ;DRIVE NO.
6579 031434 105021      CLRB      (R1)+
6580 031436 112021      MOV      (R0)+,(R1)+      ;COMMAND
6581 031440 105021      CLRB      (R1)+
6582 031442 012021      MOV      (P0)+,(R1)+      ;CYL ADDRESS
6583 031444 116021 000001      MOV      1(R0),(R1)+      ;TRACK
6584 031450 105021      CLRB      (R1)+
6585 031452 111021      MOV      (R0),(R1)+      ;SECTOR
6586 031454 105021      CLRB      (R1)+
6587 031456 016021 000006      MOV      6(R0),(R1)+      ;WORD COUNT
6588 031462 116003 030003      MOV      3(R0),R3      ;HI BA BITS
6589 031466 042703 177774      BIC      #177774,R3
6590 031472 010321      MOV      R3,(R1)+
6591 031474 016011 000004      MOV      4(R0),(R1)      ;LO BA BITS
6592 031500 104410      RESREG
6593 031502 000207      RTS      PC
6594
6595      ;*****
6596      ;SBTTL REPORT ERROR ROUTINE
6597      ;* ENTRY      JSR      PC,ERRHDL
6598      ;* RETURN     RTS      PC
6599      ;*
6600      ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
6601      ;*IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
6602      ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
6603      ;*****
6604 031504 104407      ERRHDL: SAVREG
6605 031506 152737 000377 003132      BISB      #377,DONE      ;SET DONE FLAG
6606 031514 105237 003135      INCB      ERRCNT      ;INCREMENT ERROR COUNT
6607 031520 005037 005424      CLR      RECODE      ;CLEAR RECOVERY CODE WORD
6608 031524 032737 000400 005424      BIT      #LEV2ER,RECODE      ;TEST IF 2ND LEVEL ERROR
6609 031532 001402      BEQ      52$      ;NO - SKIP PARAM BLOCK CHANGE
6610 031534 012705 002724      MOV      #PARM1,R5      ;ELSE SET R5 TO PARAMETER BLOCK 1
6611 031540 012737 033736 003056      MOV      #RETANL,A.ABNL      ;SET NOW ABNORMAL AND NORMAL RETURN FOR
6612 031546 012737 033754 003054      MOV      #RETNML,A.NORM      ;DRIVER OPERATIONS IN ERROR PROCESSING
6613 031554 004737 031224      JSR      PC,REPSUP      ;GO SET UP REGISTERS FOR REPORT
6614
6615      ;NOW BEGIN TESTING THE ERROR
6616      ;BITS. THE SEQUENCE IN
6617      ;WHICH THEY ARE TESTED IS
6618      ;CONSIDERED SIGNIFICANT IN
6619      ;THAT ERRORS OF A MORE
6620      ;CATASTROPHIC NATURE ARE FIRST
6621      ;TESTED.
6622
6623      ;IF AN ERROR IS FOUND SET,
6624      ;THAT ERROR IS REPORTED AND
6625      ;THE REPORTING IS TERMINATED.
6626      ;IF ADDITIONAL ERRORS ARE SET,
6627      ;THE RK611 REGISTER PRINTOUTS
6628      ;WILL SHOW THIS BUT THE
6629      ;REGISTER CONTENTS MUST BE
6630      ;MANUALLY DECODED TO LOCATE THE
6631      ;SECOND ERROR
6632 031560 016504 000034      MOV      P.ER(R5),R4      ;SET R4 TO ERROR REGISTER
6632 031564 032765 020000 000020      BIT      #UPE,P.CS2(R5)      ;TEST UPE. IF UES-SET

```

6633	031572	001406				BEQ	1\$		;REPORT ERROR
6634	031574	104001				ERROR	1		
6635	031576	052737	000200	005424		BIS	#ABORT,RECODE		
6636	031604	000137	033236			JMP	37\$		
6637	031610	032765	004000	000020	1\$:	BIT	#NEM,P.CS2(R5)		;TEST NON-EXISTANT MEMORY
6638	031616	001406				BEQ	2\$		
6639	031620	104002				ERROR	2		
6640	031622	052737	000200	005424		BIS	#ABORT,RECODE		
6641	031630	000137	033236			JMP	37\$		
6642	031634	032765	010000	000020	2\$:	BIT	#NED,P.CS2(R5)		;TEST NON-EXISTANT DRIVE
6643	031642	001412				BEQ	3\$		
6644	031644	032765	000400	000020		BIT	#UFE,P.CS2(R5)		;TEST IF NED & UFE BOTH SET
6645	031652	001403				BEQ	38\$		
6646	031654	104035				ERROR	35		;NED & UFE BOTH SET ERROR
6647	031656	000137	033236			JMP	37\$		
6648	031662	104003			38\$:	ERROR	3		;NED ONLY
6649	031664	000137	033236			JMP	37\$		
6650	031670	032765	000400	000020	3\$:	BIT	#UFE,P.CS2(R5)		;TEST UNIT FIELD ERROR
6651	031676	001412				BEQ	4\$		
6652	031700	032765	010000	000020		BIT	#NED,P.CS2(R5)		;TEST IF UFE & NED BOTH SET
6653	031706	001403				BEQ	39\$		;NO-SKIP
6654	031710	104035				ERROR	35		;REPORT NED & UFE BOTH SET
6655	031712	000137	033236			JMP	37\$		
6656	031716	104004			39\$:	ERROR	4		;REPORT UFE ONLY
6657	031720	000137	033236			JMP	37\$		
6658	031724	032765	000100	000014	4\$:	BIT	#CMDT0,P.PRST(R5)		
6659	031732	001423				BEQ	5\$		
6660	031734	004737	033764			JSR	PC, TOPROC		;GO PROCESS TIMEOUT
6661	031740	032737	002000	005424		BIT	#TWOLOS,RECODE		;2ND TIMEOUT IN TIMEOUT PROC?
6662	031746	001007				BNE	40\$		;YES - SKIP TO ERROR 56
6663	031750	032737	000400	005424		BIT	#LEV2ER,RECODE		;TEST IF LEVEL 2 ERROR
6664	031756	001006				BNE	41\$		;YES - SKIP TO ERROR 57
6665	031760	104005				ERROR	5		;ELSE MAKE FULL TIMEOUT REPORT
6666	031762	000137	033236			JMP	37\$		
6667	031766	104056			40\$:	ERROR	56		;TWO TIMEOUTS ERROR REPORT
6668	031770	000137	033236			JMP	37\$		
6669	031774	104057			41\$:	ERROR	57		;2ND LEVEL ERROR REPORT
6670	031776	000137	031524			JMP	ER2ENT		;GO BUILD AND MAKE 2ND REPORT
6671	032002	032765	010000	000014	5\$:	BIT	#DRVSZD,P.PRST(R5)		;SEE IF DRIVE SIEZED BY OTHER PORT
6672	032010	001403				BEQ	65\$		;OR IF DRIVE IS AVAILABLE
6673	032012	104107				ERROR	107		;DRIVE SIEZED BY OTHER PORT
6674	032014	000137	033236			JMP	37\$		
6675	032020	032765	020000	000016	65\$:	BIT	#SPAR,P.CS1(R5)		;TEST D TO C PARITY ERROR
6676	032026	001406				BEQ	6\$		
6677	032030	104006				ERROR	6		
6678	032032	052737	004000	005424		BIS	#RCLREQ,RECODE		
6679	032040	000137	033236			JMP	37\$		
6680	032044	032704	000010		6\$:	BIT	#DRPAR,R4		;TEST DRIVE DETECTED PARITY ERROR
6681	032050	001406				BEQ	7\$		
6682	032052	104007				ERROR	7		
6683	032054	052737	004000	005424		BIS	#RCLREQ,RECODE		
6684	032062	000137	033236			JMP	37\$		
6685	032066	032765	000010	000036	7\$:	BIT	#ACLO,P.DS(R5)		;TEST AC LOW
6686	032074	001403				BEQ	8\$		
6687	032076	104010				ERROR	10		
6688	032100	000137	033236			JMP	37\$		

6689	032104	032765	000020	000036	8\$:	BIT	#SPDLSS,P.DS(R5)	;TEST SPEED LOSS
6690	032112	001403				BEQ	24\$	
6691	032114	104011				ERROR	11	
6692	032116	000137	033236			JMP	37\$	
6693	032122	032765	004000	000016	24\$:	BIT	#CTO,P.CS1(R5)	;TEST FOR CONTROLLER TIMEOUT
6694	032130	001401				BEQ	25\$	
6695	032132	104027				ERROR	27	
6696	032134	032704	000001		25\$:	BIT	#ILC,R4	;TEST ILLEGAL FUNCTION CODE
6697	032140	001403				BEQ	10\$	
6698	032142	104012				ERROR	12	
6699	032144	000137	033236			JMP	37\$	
6700	032150	032765	002000	000020	10\$:	BIT	#PGE,P.CS2(R5)	;TEST PROGRAMMING ERROR
6701	032156	001403				BEQ	11\$	
6702	032160	104013				ERROR	13	
6703	032162	000137	033236			JMP	37\$	
6704	032166	032704	000004		11\$:	BIT	#ILF,R4	;TEST ILLEGAL DRIVE FUNCTION
6705	032172	001403				BEQ	12\$	
6706	032174	104014				ERROR	14	
6707	032176	000137	033236			JMP	37\$	
6708	032202	032704	000040		12\$:	BIT	#DTYPE,R4	;TEST DRIVE TYPE ERROR
6709	032206	001403				BEQ	13\$	
6710	032210	104015				ERROR	15	
6711	032212	000137	033236			JMP	37\$	
6712	032216	032704	000020		13\$:	BIT	#FMTE,R4	;TEST FORMAT ERROR
6713	032222	001403				BEQ	14\$	
6714	032224	104016				ERROR	16	
6715	032226	000137	033236			JMP	37\$	
6716	032232	032704	004000		14\$:	BIT	#WLE,R4	;TEST WRITE LOCK ERROR
6717	032236	001403				BEQ	15\$	
6718	032240	104017				ERROR	17	
6719	032242	000137	033236			JMP	37\$	
6720	032246	032704	040000		15\$:	BIT	#UNS,R4	;TEST DRIVE UNSAFE
6721	032252	001406				BEQ	16\$	
6722	032254	104020				ERROR	20	
6723	032256	052737	000200	005424		BIS	#ABORT,RECODE	
6724	032264	000137	033336			JMP	ALLTRM	
6725	032270	032704	000002		16\$:	BIT	#SKI,R4	;TEST SEEK INCOMPLETE
6726	032274	001406				BEQ	17\$	
6727	032276	104021				ERROR	21	
6728	032300	052737	004000	005424		BIS	#RCLREQ,RECODE	
6729	032306	000137	033236			JMP	37\$	
6730	032312	032704	001000		17\$:	BIT	#COE,R4	;TEST CYLINDER OVERFLOW
6731	032316	001406				BEQ	18\$	
6732	032320	104022				ERROR	22	
6733	032322	052737	004000	005424		BIS	#RCLREQ,RECODE	
6734	032330	000137	033236			JMP	37\$	
6735	032334	032704	002000		18\$:	BIT	#IDAE,R4	;TEST ILLEGAL CYLINDER
6736	032340	001406				BEQ	19\$	
6737	032342	104023				ERROR	23	
6738	032344	052737	004000	005424		BIS	#RCLREQ,RECODE	
6739	032352	000137	033236			JMP	37\$	
6740	032356	032765	000040	000036	19\$:	BIT	#DROT,P.DS(R5)	;TEST DRIVE OFF TRACK
6741	032364	001406				BEQ	20\$	
6742	032366	104024				ERROR	24	
6743	032370	052737	004000	005424		BIS	#RCLREQ,RECODE	
6744	032376	000137	033236			JMP	37\$	

6745	032402	032704	010000		20\$:	BIT	#DTE,R4	;TEST DRIVE TIMING ERROR
6746	032406	001406				BEQ	21\$	
6747	032410	104025				ERROR	25	
6748	032412	052737	000200	005424		BIS	#ABORT,RECODE	
6749	032420	000137	033236			JMP	37\$	
6750	032424	032765	100000	000020	21\$:	BIT	#DLT,P.CS2(R5)	;TEST DATA LATE
6751	032432	001403				BEQ	22\$	
6752	032434	104026				ERROR	26	
6753	032436	000137	033236			JMP	37\$	
6754	032442	032704	020000		22\$:	BIT	#OPI,R4	;TEST IF OPI ERROR
6755	032446	001470				BEQ	29\$	
6756	032450	052737	000010	005424		BIS	#OPIERR,RECODE	
6757	032456	105737	003130			TSTB	DERCNT	;TEST IF FIRST ERROR
6758	032462	001402				BEQ	50\$	
6759	032464	000137	033236			JMP	37\$	;NO - SKIP REPORT
6760	032470	032737	000400	005424	50\$:	BIT	#LEV2ER,RECODE	;TEST IF A SECOND LEVEL 2 ERROR
6761	032476	001403				BEQ	26\$	;HAS ALREADY OCCURRED
6762	032500	104054			27\$:	ERROR	54	
6763	032502	000137	033236			JMP	37\$	;GET OUT OF ERROR REPORT
6764	032506	004737	034444		26\$:	JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
6765	032512	004737	034134			JSR	PC,RDH00	;GET HEADER OF SECTOR 0
6766	032516	032737	000400	005424		BIT	#LEV2ER,RECODE	;TEST IF ERROR GETTING HDR
6767	032524	001036				BNE	28\$	;IF YES-GO MAKE ABBREVIATED REPORT
6768	032526	013702	003046			MOV	RKBAS,R2	;STORE HEADER 0 INTO REGISTERS
6769	032532	042762	000100	000000		BIC	#IE,RKCS1(R2)	;RESET INTERRUPT ENABLE
6770	032540	016237	000024	001202		MOV	RKDB(R2),SREG10	;FOR REPORTING
6771	032546	016237	000024	001204		MOV	RKDB(R2),SREG11	
6772	032554	016237	000024	001206		MOV	RKDB(R2),SREG12	
6773	032562	032762	100000	000000		BIT	#CERR,RKCS1(R2)	;TEST IF ERROR DURING STORAGE
6774	032570	001343				BNE	27\$	
6775	032572	105737	003143			TSTB	TSTTYP	;SEE IF READING WITH OFFSET
6776	032576	001403				BEQ	54\$	;BR IF NOT
6777	032600	105037	003135			CLRB	ERRCNT	;CLEAR ERROR COUNT
6778	032604	000401				BR	55\$	;CONTINUE
6779	032606	104030			54\$:	ERROR	30	;MAKE OPI REPORT
6780	032610	052737	004000	005424	55\$:	BIS	#RCLREQ,RECODE	;SET RECALIBRATE REQUEST BIT
6781	032616	000137	033236			JMP	37\$	
6782	032622	104054			28\$:	ERROR	54	
6783	032624	000137	031524			JMP	ER2ENT	;GO MAKE 2ND LEVEL REPORT
6784	032630	032704	000400		29\$:	BIT	#HVRG,R4	;TEST IF HVRG ERROR
6785	032634	001457				BEQ	23\$	
6786	032636	052737	000004	005424		BIS	#HVRGER,RECODE	
6787	032644	105737	003130			TSTB	DERCNT	;TEST IF FIRST ERROR
6788	032650	001402				BEQ	30\$	;YES - REPORT
6789	032652	000137	033236			JMP	37\$	;JUMP TO RETURN
6790	032656	032737	000400	005424	30\$:	BIT	#LEV2ER,RECODE	;TEST IF A 2ND LEVEL ERROR HAS ALREADY
6791	032664	001403				BEQ	31\$	;OCCURRED. NO-SKIP EXIT
6792	032666	104055			51\$:	ERROR	55	
6793	032670	000137	033236			JMP	37\$	;GET OUT OF ERROR REPORT
6794	032674	004737	034444		31\$:	JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
6795	032700	004737	034134			JSR	PC,RDH00	;GO GET HDR 0
6796	032704	032737	000400	005424		BIT	#LEV2ER,RECODE	;TEST IF ERROR IN GETTING HDR
6797	032712	001025				BNE	32\$	;IF YES-GO MAKE ABBREVIATED REPORT
6798	032714	013702	003046			MOV	RKBAS,R2	;GET RK611 BASE ADDRESS
6799	032720	042762	000100	000000		BIC	#IE,RKCS1(R2)	;CLEAR INTERRUPT ENABLE
6800	032726	016237	000024	001202		MOV	RKDB(R2),SREG10	;STORE OFF HEADER

Address	Instruction	Comment
6801	032734 016237 000024 001204	MOV RKDB(R2), \$REG11
6802	032742 016237 000024 001206	MOV RKDB(R2), \$REG12
6803	032750 032762 100000 000000	BIT #CERR, RKCS1(R2) ; TEST IN ANY ERROR IN UNLOAD
6804	032756 001343	BNE 51\$; IF YES - GO MAKE SHORT REPORT
6805	032760 104031	ERROR 31 ; MAKE FULL REPORT
6806	032762 000137 033236	JMP 37\$
6807	032766 104055	ERROR 55 ; ABBREVIATED HVRC ERROR REPORT
6808	032770 000137 031524	JMP ER2ENT ; GO REPORT 2ND LEVEL ERROR
6809	032774 032704 000200	BIT #BSE, R4 ; TEST FOR BAD SECTOR ERROR
6810	033000 001436	BEQ 33\$; NO - SKIP
6811	033002 052737 000002 005424	BIS #BSERR, RECODE ; SET ERROR FLAG
6812	033010 016500 000030	MOV P.DCYL(R5), R0 ; GET CYL NO.
6813	033014 116501 000027	MOVB P.DTS+1(R5), R1 ; GET TRACK NO.
6814	033020 042701 177774	BIC #177774, R1 ; CLEAR ALL BITS EXCEPT TRACK
6815	033024 016502 000026	MOV P.DTS(R5), R2 ; GET SECTOR IN ERROR
6816	033030 042702 177740	BIC #177740, R2 ; CLEAR ALL BITS EXCEPT SECTOR
6817	033034 004737 030042	JSR PC, BDSRCK ; GO SEE IF THIS SECTOR LISTED
6818	033040 032737 001000 005424	BIT #BADSEC, RECODE ; TEST RESULT
6819	033046 001073	BNE 37\$; YES - EXIT, NO ERROR OR REPORT
6820	033050 105737 003143	TSTB TSTTYP ; SEE IF READING WITH OFFSET
6821	033054 001403	BEQ 56\$; BR IF NOT
6822	033056 105037 003135	CLRB ERRCNT ; CLEAR ERROR COUNT
6823	033062 000465	BR 37\$; CONTINUE
6824	033064 104104	ERROR 104 ; ELSE REPORT BAD BSE
6825	033066 042737 000002 005424	BIC #BSERR, RECODE ; RESET BSE ERROR FLAG
6826	033074 000460	BR 37\$; GO EXIT
6827	033076 032704 100000	BIT #DCK, R4 ; TEST IF DATA CHECK
6828	033102 001435	BEQ 36\$
6829	033104 052737 000020 005424	BIS #DCKERR, RECODE ; SET DATA CHECK ERROR IN RECOVERY CODE
6830	033112 032704 000100	BIT #ECH, R4 ; TEST IF ECC IS HARD. IF
6831	033116 001406	BEQ 34\$; YES SET UNCORRECTABLE IN
6832	033120 052737 000040 005424	BIS #ECCNC, RECODE ; RECOVERY FLAG AND A 0 IN
6833	033126 005037 001206	CLR \$REG12 ; REG12 TO INDICATE UNCORRECTABLE,
6834	033132 000403	BR 35\$; A 1 IN REG 12 FOR CORRECTABLE
6835	033134 012737 000001 001206	MOV #1, \$REG12
6836	033142 105737 003130	TSTB DERCNT ; TEST IF FIRST ERROR
6837	033146 001033	BNE 37\$; NO SKIP REPORT
6838	033150 004737 034310	JSR PC, GTPKAD ; GO GET PACK ADDRESS OF ERROR
6839	033154 016537 000060 001202	MOV P.EPOS(R5), \$REG10 ; STORE ECC POSITION &
6840	033162 016537 000062 001204	MOV P.EPAT(R5), \$REG11 ; PATTERN
6841	033170 104032	ERROR 32 ; REPORT DCK ERROR
6842	033172 000137 033236	JMP 37\$
6843	033176 032765 040000 000020	BIT #WCE, P.CS2(R5) ; TEST WRITE CHECK ERROR
6844	033204 001414	BEQ 37\$
6845	033206 042737 000200 005424	BIC #ABORT, RECODE ; CLEAR ABORT & SET WRITE
6846	033214 052737 000100 005424	BIS #WCERR, RECODE ; CHECK ERROR IN RECODE
6847	033222 105737 003130	TSTB DERCNT ; TEST IF FIRST ERROR
6848	033226 001003	BNE 37\$; NO - SKIP
6849	033230 004737 034310	JSR PC, GTPKAD ; GO GET ADDRESS OF ERROR
6850	033234 104033	ERROR 33 ; REPORT WCE
6851		
6852	033236 032765 000020 000014	BIT #DRVHRD, P.PRST(R5) ; TEST HARD ERROR
6853	033244 001404	BEQ 43\$
6854	033246 104036	ERROR 36
6855	033250 052737 000200 005424	BIS #ABORT, RECODE
6856		

```

6857 033256 032765 000040 000014 43$: BIT #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
6858 033264 001404 BEQ 44$
6859 033266 104037 ERROR 37
6860 033270 052737 000200 005424 BIS #ABORT,RECODE
6861
6862 033276 032765 004000 000014 44$: BIT #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
6863 033304 001404 BEQ 46$
6864 033306 104040 ERROR 40
6865 033310 052737 000200 005424 BIS #ABORT,RECODE
6866
6867 033316 032765 000010 000014 46$: BIT #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
6868 033324 001404 BEQ ALLTRM
6869 033326 104042 ERROR 42
6870 033330 052737 000200 005424 BIS #ABORT,RECODE
6871
6872
6873 ;ALL ERRORS MUST EXIT THROUGH THIS POINT
6874
6875 033336 012705 002640 ALLTRM: MOV #PARMO,R5 ;RESTORE PARAMETER BLOCK SELECTION
6876 033342 012737 020556 003056 MOV #DCKHDL,A.ABNL ;SET READ ERROR HANDLER
6877 033350 105737 003143 TSTB TSTYP ;SEE IF READING WITH OFFSETS
6878 033354 001003 BNE 20$ ;BR IF YES
6879 033356 012737 031504 003056 MOV #ERRHDL,A.ABNL ;RESTORE INTERRUPT VECTORS FOR RETRY
6880 033364 012737 030272 003054 20$: MOV #ERRFRE,A.NORM ;DRIVER OPERATIONS, IF ANY
6881 033372 032737 000200 005424 BIT #ABORT,RECODE ;IF ABORT IS NOT SET AND
6882 033400 001057 BNE 48$ ;THE DRIVE IS READY (HAS NOT
6883 ;CYCLED DOWN)
6884 033402 013702 003046 MOV RKBAS,R2 ;GET BASE ADDRESS
6885 033406 032762 000200 000012 BIT #RDY,RKDS(R2) ;TEST IF DRIVE READY SET
6886 033414 001004 BNE 47$ ;RECALIBRATE REQUIRED BIT IS SET
6887 033416 052737 000200 005424 BIS #ABORT,RECODE ;ELSE ABORT WITH ABORT MESSAGE
6888 033424 000445 BR 48$
6889 033426 032737 004000 005424 47$: BIT #RCLREQ,RECODE ;IF RECALIBRATE IS REQUIRED
6890 033434 001451 BEQ BGNRTY ;FOR RETRY SET UP PARAM
6891 033436 013737 002640 002724 MOV PARMO,PARM1 ;SAVE COMMAND
6892 033444 112765 000113 000001 MOVVB #RECAL,P.CMND(R5) ;BLOCK TO DO IT.
6893 033452 012737 033736 003056 MOV #RETANL,A.ABNL
6894 033460 004737 030146 JSR PC,DRVCAL
6895 033464 013737 002724 002640 MOV PARM1,PARMO ;RESTORE COMMAND
6896 033472 012737 020556 003056 MOV #DCKHDL,A.ABNL ;SET READ ERROR HANDLER
6897 033500 105737 003143 TSTB TSTYP ;SEE IF READING WITH OFFSETS
6898 033504 001003 BNE 50$ ;BR IF YES
6899 033506 012737 031504 003056 MOV #ERRHDL,A.ABNL ;RESTORE ERROR RETURN
6900 033514 032737 000400 005424 50$: BIT #LEVZER,RECODE ;IF AN ERROR OCCURRED IN THE
6901 033522 001416 BEQ BGNRTY ;RECAL ATTEMP SET ABORT,
6902 033524 052737 000200 005424 BIS #ABORT,RECODE ;PRINT THE RECAL ERROR MESSAGE, AND
6903 033532 104060 ERROR 60 ;GO REPORT DETAILS
6904 033534 000137 031524 JMP ERZENT
6905 033540 104061 48$: ERROR 61 ;REPORT ABORT-RETRY FAILED
6906
6907 ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
6908 ;IF THE DRIVE READY BIT IS RESET.
6909
6910 033542 000005 HLTPRG: RESET ;DISABLE ALL DEVICES
6911 033544 000000 HALT ;HALT THE CPU
6912 033546 105037 003135 CLRB ERRCNT ;CLEAR ERROR COUNT

```

```

6913 033552 000005          RESET          ;RESET ALL DEVICES
6914 033554 000137 012236  JMP          CMSTRT
6915
6916
6917 ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
6918 ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
6919 ;THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
6920 ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
6921 ;FLAG IS SET AND PROGRAM HALTS.
6922
6923 033560 032737 000136 005424 BGNRTY: BIT      #BSERR!HVR CER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
6924 033566 001404          BEQ          3$
6925 033570 052737 100000 005424 9$:      BIS      #ANYDER,RECODE
6926 033576 000453          BR          2$
6927 033600          3$:
6928 033600 105737 003150          TSTB      NORTRY          ;SEE IF "NO-RETRY" FLAG SET
6929 033604 001371          BNE          9$          ;BR IF YES
6930 033606 032737 001000 005424          BIT      #BADSEC,RECODE ;TEST IF BAD SECTOR FLAG SET
6931 033614 001044          BNE          2$          ;IF YES-EXIT TO CALLER
6932 033616 123737 003135 003136          CMPB      ERRCNT,ERRLMT ;BEGIN RETRY IF ERROR COUNT HAS
6933 033624 001012          BNE          1$          ;NOT BEEN EXCEEDED
6934 033626 005037 001174          CLR          $REGS
6935 033632 113737 003135 001174          MOVB      ERRCNT,$REGS ;GET ERROR RETRY COUNT
6936 033640 104102          ERROR      102          ;REPORT RETRY UNSUCCESSFUL
6937 033642 052737 000200 005424          BIS      #ABORT,RECODE ;SET ABORT & QUIT
6938 033650 000734          BR          HLTORG
6939 033652 013702 003046          1$:      MOV      RKBAS,R2          ;GET RK BASE ADDRESS
6940 033656 112765 000177 000001          MOVB      #SUBCLR.P.CMND(R5) ;SET UP TO CLEAR SUBSYSTEM
6941 033664 004737 030146          JSR          PC,DRVCAL          ;GO DO IT
6942 033670 012700 005176          MOV      #COMSTR,R0          ;GO AND REESTABLISH THE COMMAND
6943 033674 012025          MOV      (R0)+,(R5)+          ;AS IT WAS ENTERED INTO THE
6944 033676 012025          MOV      (R0)+,(R5)+          ;PARAMETER BLOCK
6945 033700 012025          MOV      (R0)+,(R5)+
6946 033702 012025          MOV      (R0)+,(R5)+
6947 033704 012025          MOV      (R0)+,(R5)+
6948 033706 012025          MOV      (R0)+,(R5)+
6949 033710 012705 002640          MOV      #PARMO,R5
6950 033714 012737 000000 177776          MOV      #PRO,2#PSW          ;LOWER PRIORITY TO ALLOW INTERRUPT
6951 033722 004737 030146          JSR          PC,DRVCAL          ;CALL DRIVER
6952 033726 105037 003135          CLRB      ERRCNT          ;CLEAR ERROR COUNT
6953 033732 104410          RESREG
6954 033734 000207          RTS          PC          ;IF RETURN GETS HERE, NO ERROR
6955                                     ;OCCURRED, RECOVERY WAS SUCCESSFUL
6956
6957 033736 152737 000377 003132 RETANL: BISB      #377,DONE          ;SET DONE
6958 033744 052737 000400 005424          BIS      #LEV2ER,RECODE ;SET LEVEL TWO ERROR
6959 033752 000207          RTS          PC
6960 033754 152737 000377 003132 RETNML: BISB      #377,DONE          ;SET DONE
6961 3762 000207          RTS          PC
6962
6963
6964 ;*****
6965 ;SBTTL TIME OUT PROCESSOR ROUTINE
6966 ;THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
6967 ;GATHERING DUTIES.
6968 ;*****

```

```

6969 033764
6970 033764 104407
6971 033766 013702 003046
6972 033772 012701 001174
6973 033776 016221 000000
6974 034002 016221 000010
6975 034006 016221 000020
6976 034012 016221 000006
6977 034016 016221 000002
6978 034022 016221 000004
6979 034026 016221 000016
6980 034032 016221 000012
6981 034036 016221 000014
6982 034042 005000
6983
6984
6985 034044 012705 002724
6986 034050 112765 000141 000001
6987 034056 004737 030146
6988 034062 032765 000100 000014
6989 034070 001403
6990 034072 052737 002000 005424
6991 034100 062705 000040 1$:
6992 034104 012521
6993 034106 012521
6994 034110 012521
6995 034112 012521
6996 034114 012521
6997 034116 012521
6998 034120 012521
6999 034122 012521
7000
7001 034124 012705 002640
7002 034130 104410
7003 034132 000207
7004
7005
7006
7007
7008
7009
7010 034134
7011 034134 104407
7012 034136 016501 000026
7013 034142 016500 000052
7014 034146 042700 160017
7015 034152 006200
7016 034154 006200
7017 034156 006200
7018 034160 006200
7019 034162 012705 002724
7020 034166 010165 000004
7021 034172 010065 000002
7022 034176 112765 000177 000001
7023 034204 012737 000000 177776
7024 034212 004737 030146

TOPROC:
SAVREG
MOV RKBAS,R2
MOV #SREG5,R1 ;SET UP R1 FOR RK REGISTER STORAGE
MOV RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
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 #PARI,R5 ;SET UP TO USE PARI
MOV #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
JSR PC,DRVCAL ;CALL DRIVER
BIT #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
BEQ 1$ ;NO - SKIP
BIS #TWOTOS,RECODE ;SET TWO TIMEOUTS FLAG
ADD #P.AOD,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
;*****
ROHDO:
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 #PARI,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.
MOVB #SUBCLR,P.CMND(R5) ;SET S.S. CLEAR CMND
MOV #PRO,APSW ;ALLOW INTERRUPTS
JSR PC,DRVCAL ;INIT THE S.S.

```

```

7025 034216 112765 000125 000001      MOVB      #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
7026 034224 004737 030146              JSR      PC,DRVCAL      ;DO A READ HEADER
7027 034230 013762 024406 000000      MOV      HOLD1,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 0
7028 034236 032762 000200 000000 24$:  BIT      #RDY,RKCS1(R2) ;WAIT FOR READY
7029 034244 001774              BEQ      24$
7030 034246 013762 C 74410 000000      MOV      HOLD2,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
7031 034254 032762 0L3200 000000 26$:  BIT      #RDY,RKCS1(R2) ;WAIT FOR READY
7032 034262 001774              BEQ      26$
7033 034264 142765 000020 000007      BICB      #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
7034 034272 153765 003134 000007      BISB      FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
7035 034300 012705 002640              MOV      #PARMO,R5 ;RESTORE P.B. 0 ADDRESS
7036 034304 104410              RESREG    ;RESTORE R0-R5
7037 034306 000207              RTS      PC ;RETURN
7038
7039
7040
7041
7042
7043
7044 034310 016537 000030 001174      ;*****
7045 034316 005037 001176              ;SBTTL GET PACK ADDRESS ROUTINE
7046 034322 005037 001200              ;*****
7047 034326 116537 000026 001200      GTPKAD: MOV      P.DCYL(R5),SREG5 ;GET CYLINDER NUMBER
7048 034334 116537 000027 001176      CLR      SREG6 ;CLEAR REGISTERS FOR
7049 034342 032737 000004 005424      CLR      SREG7 ;TRACK & SECTOR STORAGE
7050 034350 001034              MOVB      P.DTS(R5),SREG7 ;STORE THE TRACK AND SECTOR
7051 034352 005737 001200              MOVB      P.DTS+1(R5),SREG6
7052              BIT      #HVRCE,RECODE ;SEE IF HVRCE ERROR
7053              BNE      5$ ;BR IF YES
7054              TST      SREG7 ;ADJUST THE ADDRESS CONTAINED IN
7055              ;THE RK REGISTERS FOR THE AUTOMATIC
7056              ;INCREMENT
7057 034356 001403              BEQ      1$
7058 034360 005337 001200              DEC      SREG7
7059 034364 000426              BR      5$
7060 034366 032765 010000 000016 1$:  BIT      #CFMT,P.CS1(R5)
7061 034374 001404              BEQ      2$
7062 034376 012737 000023 001200      MOV      #19.,SREG7
7063 034404 000403              BR      3$
7064 034406 012737 000025 001200 2$:  MOV      #21.,SREG7
7065 034414 005737 001176 3$:  TST      SREG6
7066 034420 001403              BEQ      4$
7067 034422 005337 001176              DEC      SREG6
7068 034426 000405              BR      5$
7069 034430 012737 000002 001176 4$:  MOV      #2,SREG6
7070 034436 005337 001174              DEC      SREG5
7071 034442 000207 5$:  RTS      PC
7072
7073
7074
7075
7076
7077 034444 104407              ;*****
7078 034446 016537 000030 001174      ;SBTTL BUILD EXPECTED HEADER
7079 034454 016501 000026              ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
7080 034460 042701 174377              ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN SREG5, 6, AND
7081              ;*7 FOR REPORTING.
7082              ;*****
7083              BLDEXH: SAVREG
7084              MOV      P.DCYL(R5),SREG5 ;CONSTRUCT EXPECTED HDR
7085              MOV      P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
7086              BIC      #174377,R1 ;CLEAR ALL BUT TRACK BITS

```

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 138
CZR6QB.P11 02-DEC-77 10:46 BUILD EXPECTED HEADER

Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

7100 .SBTTL RK611/RK06-RK07 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.10)

7101 ;*COPYRIGHT (C) 1975,1976,1977
7102 ;*DIGITAL EQUIPMENT CORP.
7103 ;*MAYNARD, MA. 01754
7104 ;*AUTHOR: ROY SPITZER
7105

7106 .SBTTL *WATCH-DOG TIMER
7107

7108 *****
7109 ;*
7110 ;* THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06-RK07 UNIBUS
7111 ;* SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
7112 ;* REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
7113 ;* THE RK06-RK07 DRIVER WILL USE THE LOCATION W.MTIM FOR
7114 ;* MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
7115 ;* WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
7116 ;* DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
7117 ;* (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
7118 ;* IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
7119 ;* TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
7120 ;* REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
7121 ;*

7122 ;* THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
7123 ;* OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
7124 ;* THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
7125 ;* LIMIT FOR ALL OTHER COMMANDS.
7126 ;*

7127 ;* FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
7128 ;* WATCH UP TO 8 OPERATIONS SIMULTANEOUSLY. FOR SEQUENTIAL
7129 ;* OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
7130 ;*

7131 ;*CALL JSR PC,W.WTCH
7132 ;* RETURN IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
7133 ;*
7134 ;* OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
7135 ;* BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
7136 ;* IN THE PROGRAM DEVICE STATUS REGISTER OF THE
7137 ;* APPROPRIATE PARAMETER BLOCK WILL BE SET.
7138 ;*

7139 *****
7140 ;*

7141									
7142	034572	010546							
7143	034574	010446							
7144	034576	010346							
7145	034600	010246							
7146	034602	013746	177776						
7147	034606	005337	003066						
7148	034612	001034							
7149	034614	013737	003070	003066					
7150	034622	105737	003110						
7151	034626	001426							
7152	034630	013737	003052	177776					
7153	034636	013702	003046						
7154	034642	005337	003124						
7155	034646	001016							

W.WTCH:	MOV	R5,-(SP)	;SAVE R5 ON THE STACK
	MOV	R4,-(SP)	;SAVE R4 ON THE STACK
	MOV	R3,-(SP)	;SAVE R3 ON THE STACK
	MOV	R2,-(SP)	;SAVE R2 ON STACK
	MOV	PS,-(SP)	;SAVE PROGRAM STATUS WORD ON STACK
	DEC	W.MTIM	;DECREMENT MILLISECOND TIMER
	BNE	20\$	;IF NOT ZERO RETURN
	MOV	W.MILI,W.MTIM	;REINITIALIZE MILLISECOND TIMER
	TSTB	W.TIME	;CHECK IF DRIVE IS BEING TIMED
	BEQ	20\$	;NO RETURN
	MOV	RKPRI,PS	;LOCK OUT RK06-RK07 INTERRUPTS
	MOV	RKBAS,R2	;LOAD BASE OF RK06-RK07 REGISTERS
	DEC	W.DRV	;DECREMENT COMMAND TIMER
	BNE	20\$	;RETURN IF NO TIME OUT

J11

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 140
 CZR6QB.P11 02-DEC-77 10:46 *WATCH-DOG TIMER

SEQ 0139

7156	034650	105037	003110		CLRB	W.TIME	;RESET TIMING INDICATOR
7157	034654	013705	003122		MOV	PBLKT,R5	;LOAD ADDRESS OF PARAMETER BLOCK
7158							TABLE FOR INDEXING
7159	034660	052765	000100	000014	BIS	%CMDTO,P.PRST(R5)	;SET COMMAND TIME OUT
7160	034666	020537	003064		CMP	R5,0.WAIT	;CHECK IF DRIVER IS WAITING FOR
7161							COMMAND COMPLETION
7162	034672	001002			BNE	5\$	;NO, DO NOT ALTER WAITING FOR
7163							COMMAND COMPLETION
7164	034674	005037	003064		CLR	0.WAIT	;CLEAR WAIT FOR COMMAND COMPLETION
7165	034700	004737	040154	5\$:	JSR	PC,R.ABNL	;BRANCH TO ERROR ROUTINE
7166	034704	012637	177776	20\$:	MOV	(SP)+,PS	;RESTORE PSW
7167	034710	012602			MOV	(SP)+,R2	;RESTORE R2
7168	034712	012603			MOV	(SP)+,R3	;RESTORE R3
7169	034714	012604			MOV	(SP)+,R4	;RESTORE R4
7170	034716	012605			MOV	(SP)+,R5	;RESTORE R5
7171	034720	000207			RTS	PC	;RETURN

K11

.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 (UNSUCESSFUL 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

7172
7173
7174
7175
7176
7177
7178
7179
7180
7181
7182
7183
7184
7185
7186
7187
7188
7189
7190
7191
7192
7193
7194
7195
7196
7197
7198
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216 034722 010546
7217 034724 010446
7218 034726 010346
7219 034730 010246
7220 034732 010146
7221 034734 010046
7222 034736 013702 003046
7223 034742 016237 000010 003012
7224 034750 032737 001000 003012
7225 034756 001407
7226 034760 052737 100000 003062
7227 034766 004737 040200

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

```

7228 034772 000137 037152      JMP      I.RTRN      ;RETURN
7229
7230 034776 105737 003104      1$:  TSTB      I.ISRL      ;CHECK IF INTERRUPT OR RELEASE
7231 035002 001410              BEQ        6$          ;NO, CHECK IF DRIVE AVAILABLE
7232 035004 100403              BMI        5$          ;CHECK IF RELEASE COMMAND
7233 035006 105037 003104      CLRB      I.ISRL      ;YES, CLEAR FLAG
7234 035012 000473              BR         I.ICO      ;CONTINUE PROCESSING INTERRUPT
7235
7236 035014 105037 003104      5$:  CLRB      I.ISRL      ;CLEAR FLAG
7237 035020 000137 036134      JMP      I.ATTN      ;GO PROCESS DRIVE ATTENTIONS
7238
7239 035024 032737 010400 003012 6$:  BIT        #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
7240                                UNIT FIELD ERROR
7241 035032 001413              BEQ        7$          ;NO, WAIT FOR DUAL ACCESS INTERRUPT
7242 035034 013704 003012      MOV        T.CS2,R4      ;LOAD R4 FOR DRIVE NUMBER
7243 035040 042704 177770      BIC        #IC<DRVMSK>,R4 ;KEEP DRIVE BITS
7244 035044 013705 003122      MOV        PBLKT,R5      ;STORE PARAMETER BLOCK ADDRESS
7245 035050 016237 000000 003010 MOV        RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
7246 035056 000137 035344      JMP      I.ERRC      ;REPORT ERROR
7247
7248 035062 016237 000012 003030 7$:  MOV        RKDS(R2),T.DS ;STORE STATUS REGISTER FOR COMPARISON
7249 035070 032737 000001 003030 BIT        #DRA,T.DS ;CHECK IF DRIVE SEIZED BY OTHER
7250                                PORT
7251 035076 001041              BNE        I.I00      ;NO, CONTINUE PROCESSING INTERRUPT
7252
7253                                ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
7254 035100 032737 164000 003012 BIT        #DLT!WCE!UPE!NEM,T.CS2
7255
7256 035106 001007              BNE        10$         ;INDICATE ERROR
7257 035110 016237 000014 003026 MOV        RKER(R2),T.ER ;STORE ERROR REGISTER
7258
7259                                ;CHECK FOR DATA TRANSFER ERROR TYPE ERROR
7260 035116 032737 125700 003026 BIT        #DCK!OPI!WLE!COE!4VRC!BSE!ECH,T.ER
7261
7262 035124 001407              BEQ        11$         ;NO, WAIT FOR RELEASE OF RK06 DRIVE
7263
7264 035126 052737 000010 003062 10$: BIS        #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
7265 035134 004737 040200      JSR        PC,R.CONT ;REPORT ERROR
7266 035140 000137 037152      JMP      I.RTRN      ;RESTORE REGISTERS
7267
7268 035144 105037 003110      11$: CLRB      W.TIME      ;RESET TIMING ON THIS DRIVE
7269 035150 005037 003124      CLR        W.DRV      ;CLEAR TIMING COUNT FOR THIS DRIVE
7270 035154 013705 003122      MOV        PBLKT,R5      ;LOAD R5 WITH PARAMETER BLOCK
7271                                ADDRESS
7272 035160 052765 010000 000014 BIS        #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
7273                                PROGRAM DRIVE STATUS REGISTER
7274 035166 005037 003064      CLR        O.WAIT      ;CLEAR WAIT FOR COMMAND COMPLETION
7275 035172 004737 040154      JSR        PC,R.ABNL ;INDICATE ABNORMAL TERMINATION
7276 035176 000137 037152      JMP      I.RTRN      ;GO RESTORE REGISTERS
7277
7278 035202 013705 003064      I.I00: MOV        O.WAIT,R5 ;LOAD PARAMETER BLOCK ADDRESS INTO R5
7279 035206 001002              BNE        2$          ;IS COMMAND WAITING PROCESSING
7280                                ;YES, DO PROCESSING
7281 035210 000137 036134      JMP      I.ATTN      ;NO, PROCESS ATTENTION
7282
7283 035214 013704 003012      2$:  MOV        T.CS2,R4      ;STORE RKCS2 FOR DRIVE NUMBER

```

```

7284 035220 042704 177770      BIC      #1C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
7285
7286
7287 035224 126504 000000      CMPB     P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
7288 035230 001401      BEQ      3$ ;YES, CONTINUE
7289 035232 000000      HALT     ;NO, DRIVER ERROR
7290 035234 122765 000164 000001 3$: CMPB     #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
7291 035242 001002      BNE     10$ ;NO, EXECUTE NORMAL DATA TRANSFER
7292 035244 000137 035602      JMP     I.HDAL ;GO EXECUTE SPECIAL HEADER SEQUENCE
7293
7294 035250 005037 003064      CLR      0.WAIT ;CLEAR WAIT FOR COMMAND COMPLETION
7295 035254 005037 003124      CLR      W.DRV ;CLEAR WATCH-DOG TIME
7296 035260 105037 003110      CLRB     W.TIME ;RESET TIMING ON THIS DRIVE
7297 035264 016237 000000 003010 MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7298 035272 032737 100000 003010 BIT      #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
7299 035300 001021      BNE     I.ERRC ;YES, PROCESS ERROR
7300 035302 016237 000016 003024 MOV      RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7301 035310 133737 003111 003025 BITB     INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
7302 035316 001004      BNE     15$ ;YES, REPORT ERROR
7303 035320 004737 040166      JSR      PC,R.NORM ;INDICATE NORMAL RETURN
7304 035324 000137 037152      JMP     I.ATRN ;RESTORE REGISTERS
7305
7306 035330 052765 000010 000014 15$: BIS      #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
7307
7308 035336 004737 037622      I.ERRA: JSR      PC,I.CSTS ;STORE CONTROLLER STATUS
7309 035342 000405      BR      I.ERR ;STORE PATTERN AND POSITION INFORMATION
7310
7311 035344 013765 003010 000016 I.ERRC: MOV      T.CS1,P.CS1(R5) ;GET ERROR RKCS1
7312 035352 004737 037644      JSR      PC,I.CST1 ;GET REST OF CONTROLLER STATUS
7313 035356 016265 000032 000062 I.ERR: MOV      RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
7314 035364 016265 000030 000060 MOV      RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
7315 035372 004037 037170      JSR      RO,I.CCLR ;CLEAR CONTROLLER
7316 035376 037152      I.RTRN ;ERROR RETURN
7317 035400 032765 010400 000020 BIT      #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
7318 ; UNIT FIELD ERROR
7319 035406 001046      BNE     5$ ;YES, REPORT ERROR
7320 035410 004037 037726      JSR      RO,I.STAT ;GATHER DRIVE STATUS
7321 035414 037152      I.RTRN ;ERROR RETURN
7322 035416 112737 000005 003010 MOVB     #DR.CLR,T.CS1 ;LOAD COMMAND
7323 035424 004037 037252      JSR      RO,I.ISSU ;ISSUE DRIVE CLEAR
7324 035430 037152      I.RTRN ;ERROR RETURN
7325 035432 133737 003111 003025 BITB     INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
7326 035440 001407      BEQ     2$ ;NO, INDICATE DRIVE ERROR
7327 035442 052737 000020 003062 BIS      #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
7328 ; WITH CLEAR
7329 035450 004737 040200      JSR      PC,R.CONT ;REPORT CONTROLLER ERROR
7330 035454 000137 037152      JMP     I.ATRN ;GO RESTORE REGISTERS
7331
7332 035460 032737 040000 003034 2$: BIT      #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE CLEARED
7333 035466 001403      BEQ     3$ ;YES, CHECK FAULT
7334 035470 052765 000040 000014 BIS      #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
7335 035476 032737 001000 003036 3$: BIT      #S.PAR,T.MR3 ;CHECK IF DRIVE PARITY ERROR
7336 035504 001407      BEQ     5$ ;NO, INDICATE ABNORMAL TERMINATION
7337 035506 052737 002000 003062 BIS      #E.OPAR,E.CONT ;SET DRIVE PARITY ERROR
7338 035514 004737 040200      JSR      PC,R.CONT ;INDICATE CONTROLLER ERROR
7339 035520 000137 037152      JMP     I.ATRN ;RETURN

```

N11

CZR6Q80 Rk6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 144
 CZR6Q8.P11 02-DEC-77 10:46 *RK06 INTERRUPT SERVICE ROUTINE

SEQ 0143

```

7340
7341 035524 032765 000020 000014 5$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
7342 035532 001017 10$ BNE 10$ ;YES, GO REPORT ERROR
7343 035534 032737 020000 003034 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
7344 035542 001413 10$ BEQ 10$ ;NO, REPORT ERROR
7345 035544 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
7346 035552 113737 003111 000010 MOV INTMSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
7347 035560 013737 003074 000014 MOV W.8SEC,W.DRV
7348 035566 000137 037152 JMP I.RTRN ;GO RESTORE REGISTERS
7349
7350 035572 004737 040154 10$: JSR PC,R.ABNL ;GO REPORT ERROR
7351 035576 000137 037152 JMP I.RTRN ;GO RESTORE REGISTERS
7352
7353 .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
7354
7355 035602 016237 000000 003010 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
7356 035610 032737 100000 003010 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
7357 035616 001422 5$ BEQ 5$ ;NO, CHECK FOR ATTENTION
7358
7359 035620 005037 003064 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
7360 035624 105037 003110 CLRB W.TIME ;RESET TIMING ON DRIVE
7361 035630 005037 003124 CLR W.DRV ;CLEAR TIME OUT COUNT
7362 035634 013765 003010 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
7363 035642 004737 037644 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
7364 035646 004037 037170 JSR RD,I.CCLR ;CLEAR CONTROLLER
7365 035652 037152 I.RTRN ;ERROR RETURN
7366 035654 004737 040154 JSR PC,R.ABNL ;INDICATE ERROR RETURN
7367 035660 000137 037152 JMP I.RTRN ;RESTORE REGISTERS
7368
7369 035664 016237 000016 003024 5$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7370 035672 133737 003111 003025 BITB INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
7371 035700 001410 7$ BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
7372 035702 005037 003064 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
7373 035706 105037 003110 CLRB W.TIME ;RESET TIMING ON DRIVE
7374 035712 005037 003124 CLR W.DRV ;CLEAR TIME OUT COUNT
7375 035716 000137 035336 JMP I.ERRA ;GO REPORT ERROR
7376
7377 035722 013701 003100 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
7378 035726 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
7379 035732 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
7380 035736 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
7381 035742 010137 003100 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
7382 035746 016237 000010 003012 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
7383 035754 032737 100000 003012 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
7384 035762 001055 3$ BNE 3$ ;YES, REPORT ERROR
7385 035764 005337 003102 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
7386 035770 001026 2$ BNE 2$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
7387 035772 005037 003064 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
7388 035776 005037 003124 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
7389 036002 105037 003110 CLRB W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
7390 036006 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
7391 036014 112737 000001 003010 MOVB #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
7392 036022 004037 037252 JSR RD,I.ISSU ;GET SECTOR COUNT
7393 036026 037152 I.RTRN ;ERROR RETURN
7394 036030 013765 003036 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P.1 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 145
*READ ALL HEADERS INTERRUPT SEQUENCE

SEQ 0144

7396	036036	004737	040166			JSR	PC.R.NORM	;INDICATE NORMAL TERMINATION
7397	036042	000137	037152			JMP	I.ATRN	;RESTORE REGISTERS
7398								
7399	036046	016562	000002	000020	25\$:	MOV	P.CYL(R5),RKDCYL(R2)	;LOAD CYLINDER ADDRESS REGISTER
7400	036054	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	;LOAD SECTOR AND TRACK
7401	036062	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5)	;STORE BITS 8-15 OF CS1
7402	036070	042765	165777	000016		BIC	*C<CDT!CFMT>,P.CS1(R5)	;CLEAR ALL BITS EXCEPT FORMAT AND
7403								DRIVE TYPE
7404	036076	112765	000125	000016		MOVB	*RDHEAD,P.CS1(R5)	;STORE COMMAND ISSUED
7405	036104	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2)	;ISSUE READ HEADER
7406	036112	000137	037152			JMP	I.ATRN	;RESTORE REGISTERS
7407								
7408	036116	052737	000400	003062	35\$:	BIS	*E.DLT,E.CONT	;SET DATA LATE WHILE UNLOADING HEADER
7409	036124	004737	040200			JSR	PC.R.CONT	;REPORT ERROR
7410	036130	000137	037152			JMP	I.ATRN	;RESTORE REGISTERS
7411								
7412						.SBTTL	*DRIVE ATTENTION SCANNER	
7413								
7414	036134	016237	000000	003010	I.ATTN:	MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS
7415								REGISTER 1 FOR COMPARISON
7416	036142	032737	100000	003010		BIT	*CERR,T.CS1	;CHECK IF CONTROLLER ERROR OCCURRED
7417	036150	001441				BEQ	\$S	;NO, CHECK IF ATTENTION
7418								
7419								;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
7420	036152	032737	164000	003012		BIT	*DLT!WCE!UPE!NEM,T.CS2	
7421								
7422	036160	001007				BNE	\$S	;INDICATE ERROR
7423	036162	016237	000014	003026		MOV	RKER(R2),T.ER	;STORE ERROR REGISTER
7424								
7425								CHECK FOR DATA TRANSFER ERROR TYPE
7426	036170	032737	125700	003026		BIT	*DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER	
7427								
7428	036176	001407				BEQ	\$S	;NO DATA TRANSFER ERROR
7429								
7430	036200	052737	000010	003062	1\$:	BIS	*E.UDAT,E.CONT	;SET UNEXPECTED DATA TYPE ERROR
7431	036206	004737	040200			JSR	PC.R.CONT	;REPORT ERROR
7432	036212	000137	037152			JMP	I.ATRN	;RESTORE REGISTERS
7433								
7434	036216	013704	003012		2\$:	MOV	T.CS2,R4	;SAVE CS2 FOR REGISTER NUMBER
7435	036222	042704	177770			BIC	*C<DRVMSK>,R4	;STRIP OFF JUNK
7436	036226	105037	003110			CLRB	W.TIME	;CLEAR WATCH DOG TIMER
7437	036232	005037	003124			CLR	W.DRV	;RESET TIMER VALUE
7438	036236	013705	003122			MOV	PBLKT,R5	;STORE PARAMETER BLOCK ADDRESS IN R5
7439								
7440								CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
7441								IN PROGRAM DEVICE STATUS REGISTER
7442	036242	042765	000006	000014		BIC	*DRVPOS!DRVPDT,P.PRST(R5)	
7443								
7444	036250	000137	035344			JMP	I.ERRC	;GO REPORT ERROR
7445								
7446	036254	032737	040000	003010	5\$:	BIT	*DI,T.CS1	;CHECK IF ANY DRIVE ATTENTION
7447	036262	001002				BNE	\$S	;YES, PROCESS INTERRUPT
7448	036264	000137	037152			JMP	I.ATRN	;RESTORE REGISTERS
7449								
7450	036270	016237	000016	003024	6\$:	MOV	RKASOF(R2),T.ASOF	;STORE ATTENTION SUMMARY
7451	036276	105737	003025			TSTB	T.ASOF+1	;CHECK IF ANY ATTENTIONS SET

7452	036302	001007			BNE	7\$	;YES GO PROCESS INTERRUPT
7453	036304	052737	000002	003062	BIS	#E.NOAT,E.CONT	;SET NO ATTENTION IN ATTENTION SUMMARY
7454	036312	004737	040200		JSR	PC,R.CONT	;GO REPORT ERROR
7455	036316	000137	037152		JMP	I.ATRN	;GO RESTORE REGISTERS
7456							
7457	036322	133737	003111	003025	7\$: BITB	INTMSK,T.ASOF+1	;CHECK IF DESIRED INTERRUPT
7458	036330	001007			BNE	8\$	;YES, GO PROCESS IT
7459	036332	052737	000004	003062	BIS	#E.UATT,E.CONT	;SET UNSOLICATED ATTENTION
7460	036340	004737	040200		JSR	PC,R.CONT	;GO REPORT ERROR
7461	036344	000137	037152		JMP	I.ATRN	;GO RESTORE REGISTERS
7462							
7463	036350	013705	003122		8\$: MOV	PBLKT,R5	;STORE PARAMETER BLOCK TABLE
7464	036354	116504	000000		MOVB	P.DRVN(R5),R4	;STORE DRIVE NUMBER
7465	036360	032765	020000	000014	BIT	#E.UNLD,P.PRST(R5)	;CHECK IF DRIVE UNLOADING
7466	036366	001402			BEQ	11\$	;NO, CONTINUE
7467	036370	000137	037052		JMP	I.UNLD	;SERVICE DRIVE IN POSITION AFTER ERROR
7468							
7469	036374	042765	000002	000014	11\$: BIC	#DRVPOS,P.PRST(R5)	;RESET DRIVE POSITIONING
7470	036402	005062	000026		CLR	RKMRI(R2)	;CLEAR MAINTENANCE REGISTER 1
7471	036406	112737	000001	003010	MOVB	#DR.SEL,T.CS1	;LOAD COMMAND
7472	036414	004037	037252		JSR	RO,I.ISSU	;SELECT DRIVE WITH ATTENTION HIGH
7473	036420	037152			I.ATRN		;ERROR RETURN
7474	036422	013765	003036	000042	MOV	T.MR3,P.800(R5)	;STORE STATUS BYTE 00 MESS B
7475	036430	032765	000200	000042	BIT	#S.FLT,P.800(R5)	;CHECK IF DRIVE FAULT
7476	036436	001401			BEQ	12\$	;NO, CHECK FOR DRIVE STATUS CHANGE
7477	036440	000461			BR	I.F-RR	;PROCESS ERROR
7478							
7479	036442	013765	003034	000040	12\$: MOV	T.MR2,P.800(R5)	;STORE MAINTENANCE REGISTER 2
7480	036450	032765	040000	000040	BIT	#S.DSC,P.800(R5)	;CHECK FOR DRIVE STATUS CHANGE
7481	036456	001004			BNE	13\$	;YES, PROCESS DRIVE STATUS CHANGE
7482	036460	052765	004000	000014	BIS	#NODSC,P.PRST(R5)	;SET NO DRIVE STATUS CHANGE
7483	036466	000446			BR	I.AERR	;PROCESS ERROR
7484							
7485	036470	112737	000005	003010	13\$: MOVB	#DR.CLR,T.CS1	;LOAD COMMAND
7486	036476	004037	037252		JSR	RO,I.ISSU	;CLEAR DRIVE STATUS CHANGE
7487	036502	037152			I.ATRN		;ERROR RETURN
7488	036504	013765	003024	000032	MOV	T.ASOF,P.ASOF(R5)	;STORE ATTENTION SUMMARY
7489	036512	133765	003111	000033	BITB	INTMSK,P.ASOF+1(R5)	;CHECK IF ATTENTION RESET
7490	036520	001407			BEQ	15\$	;YES, CONTINUE INTERRUPT PROCESSING
7491	036522	052737	000020	003062	BIS	#E.CLAT,E.CONT	;SET ATTENTION DID NOT RESET
7492							;WITH DRIVE CLEAR
7493	036530	004737	040200		JSR	PC,R.CONT	;FLAG ERROR
7494	036534	000137	037152		JMP	I.ATRN	;RESTORE REGISTERS
7495							
7496	036540	013765	003034	000040	15\$: MOV	T.MR2,P.800(R5)	;STORE MAINTENANCE REGISTER 2
7497	036546	032765	040000	000040	BIT	#S.DSC,P.800(R5)	;CHECK IF DRIVE STATUS CHANGE
7498							;RESET
7499	036554	001404			BEQ	16\$	;YES, CONTINUE INTERRUPT PROCESSING
7500	036556	052765	000040	000014	BIS	#DRVDSC,P.PRST(R5)	;SET DRIVE STATUS CHANGE DID NOT CLEAR
7501	036564	000407			BR	I.AERR	;GO PROCESS ERROR
7502							
7503	036566	105037	003110		16\$: CLRB	W.TIME	;RESET TIMING ON THIS DRIVE
7504	036572	005037	003124		CLR	W.DRV	;CLEAR DRIVE TIMING COUNT
7505	036576	004737	040166		JSR	PC,R.NORM	;REPORT SUCCESSFUL COMMAND COMPLETION
7506	036602	000563			BR	I.ATRN	;RESTORE REGISTERS
7507							

```

7508 .SBTTL *ATTENTION ERROR HANDLER
7509
7510 036604 042765 000004 000014 I.AERR: BIC #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
7511 ; OF DATA TRANSFER
7512 036612 105037 003110 CLR B W.TIME ;CLEAR TIMING FOR THIS DRIVE
7513 036616 005037 003124 CLR W.DRV ;RESET WATCH-DOG TIME
7514 036622 042765 177741 000016 BIC #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
7515 036630 042737 000036 003010 BIC #36,T.CS1 ;KEEP CURRENT CONTROLLER STATUS
7516 036636 053765 003010 000016 BIS T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
7517 036644 013765 003012 000020 MOV T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
7518 036652 013765 003014 000022 MOV T.WCR,P.WCR(R5)
7519 036660 013765 003016 000024 MOV T.BA,P.BAR(R5)
7520 036666 013765 003020 000026 MOV T.DA,P.DTS(R5)
7521 036674 013765 003022 000030 MOV T.DC,P.DCYL(R5)
7522 036702 013765 003024 000032 MOV T.ASOF,P.ASOF(R5)
7523 036710 013765 003026 000034 MOV T.ER,P.ER(R5)
7524 036716 013765 003030 000036 MOV T.DS,P.DS(R5)
7525 036724 004037 037726 JSR RO,I.STAT ;GATHER DRIVE STATUS
7526 036730 037152 I.RTRN ;ERROR RETURN
7527 036732 112737 000005 003010 MOV B #DR,CLR,T.CS1 ;LOAD COMMAND
7528 036740 004037 037252 JSR RO,I.ISSU ;CLEAR DRIVE ERRORS
7529 036744 037152 I.RTRN ;ERROR RETURN
7530 036746 133737 003111 003025 BITB INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
7531 036754 001407 BEQ 2$ ;YES, FLAG DRIVE ERROR
7532 036756 052737 000020 003062 BIS #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
7533 036764 004737 040200 JSR PC,R.CONT ;REPORT ERROR
7534 036770 000137 037152 JMP I.RTRN ;RESTORE REGISTERS
7535
7536 036774 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
7537 037002 001017 BNE 10$ ;YES, REPORT ERROR
7538 037004 032737 020000 003034 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS UNLOADING
7539 037012 001413 BEQ 10$ ;NO, REPORT ERROR
7540 037014 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
7541 037022 113737 003111 003110 MOVB INTMSK,W.TIME ;SET TIMING ON THIS DRIVE
7542 037030 013737 003074 003124 MOV W.BSEC,W.DRV ;LOAD 8 SECONDS FOR CYCLE UP TIME
7543 037036 000137 037152 JMP I.RTRN ;RESTORE REGISTERS
7544
7545 037042 004737 040154 10$: JSR PC,R.ABNL ;REPORT ERROR
7546 037046 000137 037152 JMP I.RTRN ;RESTORE REGISTERS
7547
7548 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
7549
7550 037052 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
7551 037060 112737 000005 003010 MOV B #DR,CLR,T.CS1 ;LOAD IN DRIVE CLEAR
7552 037066 004037 037252 JSR RO,I.ISSU ;GO ISSUE DRIVE CLEAR
7553 037072 037152 I.RTRN ;ERROR RETURN
7554 037074 136437 003111 003025 BITB INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
7555 037102 001406 BEQ 15$ ;YES, CONTINUE
7556 037104 012737 000020 003062 MOV #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
7557 037112 004737 040200 JSR PC,R.CONT ;REPORT ERROR
7558 037116 000415 BR I.RTRN ;RESTORE REGISTERS
7559
7560 037120 032737 040000 003034 15$: BIT #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE RESET
7561 037126 001403 BEQ 20$ ;YES, CONTINUE
7562 037130 052765 000040 000014 BIS #DRVDSO,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
7563 037136 105037 003110 20$: CLR B W.TIME ;RESET TIMING ON THIS DRIVE

```

E12

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 148
CZR6Q8.P11 02-DEC-77 10:46 *ERROR CAUSING DRIVE TO UNLOAD

SEQ 0147

7564	037142	005037	003124	CLR	W.DRV	;CLEAR TIME COUNT
7565	037146	004737	040154	JSR	PC,R.ABNL	;REPORT ERROR
7566						
7567	037152	012600		I.RTRN: MOV	(SP)+,R0	;RESTORE R0
7568	037154	012601		MOV	(SP)+,R1	;RESTORE R1
7569	037156	012602		MOV	(SP)+,R2	;RESTORE R2
7570	037160	012603		MOV	(SP)+,R3	;RESTORE R3
7571	037162	012604		MOV	(SP)+,R4	;RESTORE R4
7572	037164	012605		MOV	(SP)+,R5	;RESTORE R5
7573	037166	000002		RTI		;RETURN
7574						

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 149
*CONTROLLER CLEAR ROUTINE

SEQ 0148

```

7575 .SBTTL *CONTROLLER CLEAR ROUTINE
7576
7577 ;*****
7578 ;
7579 ; THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
7580 ; AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT
7581 ; CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
7582 ; E.CCLR SET IN E.CONT.
7583 ;
7584 ; REGISTER USE
7585 ; -----
7586 ;
7587 ; R2 ADDRESS OF RK06 REGISTERS
7588 ; R5 ADDRESS OF PARAMETER BLOCK
7589 ;
7590 ;CALL JSR R0,I.CCLR
7591 ; <ADDRESS OF ERROR RETURN>
7592 ; RETURN
7593 ;
7594 ;*****
7595
7596 037170 012762 100000 000000 I.CCLR: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
7597 037176 016237 000000 003010 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7598 037204 032737 100000 003010 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER CLEAR DID
7599 ; CLEAR ERROR
7600 037212 001407 BEQ S$ ;YES, RETURN TO DRIVER PROCESSING
7601 037214 052737 000001 003062 BIS #E.CCLR,E.CONT ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
7602 037222 004737 040200 JSR PC,R.CONT ;REPORT CONTROLLER ERROR
7603 037226 011000 MOV (R0),R0 ;SET UP ERROR RETURN
7604 037230 000200 RTS R0 ;RETURN
7605
7606 037232 012762 000100 000000 S$: MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
7607 037240 112737 177777 003104 MOVB #-1,I.ISRL ;SET INTERRUPT ENABLE ISSUED
7608 037246 005720 TST (R0)+ ;ADJUST FOR NORMAL RETURN
7609 037250 000200 RTS R0 ;RETURN

```

.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 RD.I.ISSU
;* <ADDRESS OF ERROR RETURN>
;* RETURN
;*;* ROUTINES USED:
;* -----;* I.CCLR
;* I.STOR
;*

;*****

7610				
7611				
7612				
7613				
7614				
7615				
7616				
7617				
7618				
7619				
7620				
7621				
7622				
7623				
7624				
7625				
7626				
7627				
7628				
7629				
7630				
7631				
7632				
7633				
7634				
7635				
7636				
7637				
7638	037252	013746	003010	
7639	037256	005037	003012	
7640	037262	116537	000000	003012
7641	037270	013762	003012	000010
7642	037276	116537	000007	003011
7643	037304	142737	177753	003011
7644				
7645	037312	013762	003010	000000
7646	037320	105762	000000	
7647	037324	100375		
7648	037326	004737	037474	
7649	037332	032737	100000	003010
7650	037340	001437		
7651	037342	032737	001000	003012
7652	037350	001406		
7653	037352	052737	100000	003062
7654	037360	004737	040200	
7655	037364	000440		
7656				
7657				
7658	037366	032737	024000	003010
7659	037374	001027		
7660	037376	032737	176400	003012
7661	037404	001023		
7662	037406	032737	131761	003026
7663	037414	001017		
7664				
7665	037416	122716	000005	

```

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   #T.CB.CDT!B.CFMT, T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                          ;FORMAT AND DRIVE TYPE
          MOV    T.CS1, RKCS1(R2) ;ISSUE COMMAND
1$:      TSTB    RKCS1(R2)        ;WAIT FOR READY
          BPL    1$
          JSR    PC, I.STOR      ;GO STORE REGISTERS
          BIT    #CERR, T.CS1    ;CHECK IF CONTROLLER ERROR OCCURED
          BEQ    5$              ;NO, RETURN
          BIT    #MDS, T.CS2     ;CHECK IF MULTIPLE DRIVE SELECT
          BEQ    2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
          BIS    #E.MDS, E.CONT  ;SET MULTIPLE DRIVE SELECT FLAG
          JSR    PC, R.CONT      ;REPORT CONTROLLER ERROR
          BR     10$             ;RETURN

          ;CHECK IF ANY CONTROLLER ERROR IS SET
2$:      BIT    #CTO!SPAR, T.CS1
          BNE    7$
          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

```

H12

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 151
*COMMAND ISSUED BY DRIVER SERVICE ROUTINE

SEQ 0150

7666	037422	001003				BNE	3\$		;NO, DO NOT SET DRIVE HARD ERROR
7667	037424	052765	000020	000014		BIS	*DRVHRD,P.PRST(R5)		;SET HARD DRIVE ERROR
7668	037432	004037	037170		3\$:	JSR	RD,I.CCLR		;GO ISSUE A CONTROLLER CLEAR
7669	037436	037466				10\$			;ERROR RETURN
7670	037440	012762	000100	000000	5\$:	MOV	*IE,RKCS1(R2)		;SET INTERRUPT ENABLE
7671	037446	005726				TST	(SP)+		;ADJUST STACK
7672	037450	005720				TST	(RO)+		;ADJUST RO FOR NORMAL RETURN
7673	037452	000200				RTS	RO		;RETURN
7674									
7675	037454	052737	001000	003062	7\$:	BIS	*E.CERR,E.CONT		;SET CONTROLLER ERROR DURING
7676									; DRIVER SERVICING
7677	037462	004737	040200			JSR	PC,R.CONT		;REPORT ERROR
7678	037466	005726			10\$:	TST	(SP)+		;ADJUST STACK
7679	037470	011000				MOV	(RO),RO		;ADJUST RO FOR ERROR RETURN
7680	037472	000200				RTS	RO		;RETURN

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6QB.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 152
*STORE RK611 UNIBUS REGISTERS

SEQ 0151

7681				
7682				
7683				
7684				
7685				
7686				
7687				
7688				
7689				
7690				
7691				
7692				
7693				
7694				
7695				
7696				
7697				
7698	037474	016237	000000	003010
7699	037502	016237	000010	003012
7700	037510	016237	000002	003014
7701	037516	016237	000004	003016
7702	037524	016237	000006	003020
7703	037532	016237	000012	003030
7704	037540	016237	000014	003026
7705	037546	016237	000016	003024
7706	037554	016237	000020	003022
7707	037562	016237	000026	003032
7708	037570	016237	000034	003034
7709	037576	016237	000036	003036
7710	037604	016237	000030	003040
7711	037612	016237	000032	003042
7712	037620	000207		

.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
;
*****
;

```

```

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

```

CZR6QBD RK6 DR CPT PRGG MACY11 30(1046)
CZR6QB.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 153
*STORE CONTROLLER STATUS

SEQ 0152

.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

```
*****
```

7743	037622	042765	177741	000016	I.CSTS: BIC	#177741,P.CS1(R5)	: CLEAR ALL BITS EXCEPT FUNCTION
7744							: OF LAST COMMAND ISSUED
7745	037630	042737	000036	003010	BIC	#36,T.CS1	: CLEAR FUNCTION OF CS1 STATUS
7746	037636	053765	003010	000016	BIS	T.CS1,P.CS1(R5)	: GENERATE CS1 STATUS INFORMATION
7747	037644	016265	000010	000020	I.CST1: MOV	RKCS2(R2),P.CS2(R5)	: STORE COMMAND AND STATUS REGISTER 2
7748	037652	016265	000002	000022	MOV	RKWC(R2),P.WCR(R5)	: STORE WORD COUNT REGISTER
7749	037660	016265	000004	000024	MOV	RKBA(R2),P.BAR(R5)	: STORE BUS ADDRESS REGISTER
7750	037666	016265	000006	000026	MOV	RKDA(R2),P.DTS(R5)	: STORE DESIRED TRACK AND SECTOR
7751	037674	016265	000012	000036	MOV	RKDS(R2),P.DS(R5)	: STORE DRIVE STATUS REGISTER
7752	037702	016265	000014	000034	MOV	RKER(R2),P.ER(R5)	: STORE ERROR REGISTER
7753	037710	016265	000016	000032	MOV	RKASOF(R2),P.ASOF(R5)	: STORE ATTENTION SUMMARY AND
7754							: OFFSET
7755	037716	016265	000020	000030	MOV	RKDCYL(R2),P.DCYL(R5)	: STORE CYLINDER ADDRESS
7756	037724	000207			RTS	PC	: RETURN
7757							

```

7758 .SBTTL *GATHER DRIVE STATUS
7759
7760 ;*****
7761 ;
7762 ; THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
7763 ; BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
7764 ; HAS PREVIOUSLY BEEN SEIZED. IT RUNS AT PRIORITY 7.
7765 ;
7766 ;CALL JSR RO,I,STAT
7767 ; <ADDRESS OF ERROR RETURN>
7768 ; RETURN
7769 ;
7770 ; THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
7771 ;
7772 ;
7773 ; REGISTER CONTENTS
7774 ; -----
7775 ;
7776 ; R2 RK06 BASE ADDRESS
7777 ; R5 ADDRESS OF PARAMETER BLOCK
7778 ;
7779 ; ROUTINES USED:
7780 ; I.ISSU
7781 ;
7782 ;*****
7783
7784 037726 012762 000001 000026 I,STAT: MOV #1,RKMR1(R2) ; LOAD MAINTENANCE REGISTER 1
7785 ; FOR STATUS BYTE 01
7786 037734 112737 000001 003010 MOV B #DR,SEL,T,CS1 ; LOAD COMMAND
7787 037742 004037 037252 JSR RO,I,ISSU ; GET STATUS BYTES 01
7788 037746 040136 3$ ; ERROR RETURN
7789 037750 013765 003034 000044 MOV T,MR2,P,A01(R5) ; STORE STATUS BYTE 01 MESS A
7790 037756 013765 003036 000046 MOV T,MR3,P,B01(R5) ; STORE STATUS BYTE 01 MESS B
7791 037764 012762 000002 000026 MOV #2,RKMR1(R2) ; LOAD MAINTENANCE REGISTER 1
7792 ; FOR STATUS BYTE 10
7793 037772 112737 000001 003010 MOV B #DR,SEL,T,CS1 ; LOAD COMMAND
7794 040000 004037 037252 JSR RO,I,ISSU ; GET STATUS BYTES 10
7795 040004 040136 3$ ; ERROR RETURN
7796 040006 013765 003034 000050 MOV T,MR2,P,A10(R5) ; STORE STATUS BYTE 10 MESS A
7797 040014 013765 003036 000052 MOV T,MR3,P,B10(R5) ; STORE STATUS BYTE 10 MESS B
7798 040022 012762 000003 000026 MOV #3,RKMR1(R2) ; LOAD MAINTENANCE REGISTER
7799 ; FOR STATUS BYTE 11
7800 040030 112737 000001 003010 MOV B #DR,SEL,T,CS1 ; LOAD COMMAND
7801 040036 004037 037252 JSR RO,I,ISSU ; GET STATUS BYTES 11
7802 040042 040136 3$ ; ERROR RETURN
7803 040044 013765 003034 000054 MOV T,MR2,P,A11(R5) ; STORE STATUS BYTE 11 MESS A
7804 040052 013765 003036 000056 MOV T,MR3,P,B11(R5) ; STORE STATUS BYTE 11 MESS B
7805 040060 005062 000026 CLR RKMR1(R2) ; LOAD MAINTENANCE REGISTER 1
7806 ; FOR STATUS BYTE 00
7807 040064 112737 000001 003010 MOV B #DR,SEL,T,CS1 ; LOAD COMMAND
7808 040072 004037 037252 JSR RO,I,ISSU ; GET STATUS BYTES 00
7809 040076 040136 3$ ; ERROR RETURN
7810 040100 013765 003034 000040 MOV T,MR2,P,A00(R5) ; STORE STATUS BYTE 00 MESS A
7811 040106 013765 003036 000042 MOV T,MR3,P,B00(R5) ; STORE STATUS BYTE 00 MESS B
7812 040114 032737 001000 003036 BIT #5,PAR,T,MR3 ; CHECK IF BAD PARITY DETECTED BY DRIVE
7813 040122 001407 BEQ 5$ ; NO, RETURN NORMALLY

```


CZR6080 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 156
 CZR608.P11 02-DEC-77 10:46 *COMMON DRIVER RETURNS

SEQ 0155

7823				.SBTTL	*COMMON DRIVER RETURNS	
7824						
7825	040154	105037	003111	R.ABNL:	CLRB INTMSK	; INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7826	040160	004777	142672		JSR PC,QA.ABNL	; INDICATE ABNORMAL RETURN
7827	040164	000207			RTS PC	; RETURN
7828						
7829	040166	105037	003111	R.NORM:	CLRB INTMSK	; INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7830	040172	004777	142656		JSR PC,QA.NORM	; INDICATE NORMAL RETURN
7831	040176	000207			RTS PC	; RETURN
7832						
7833	040200	105037	003111	R.CONT:	CLRB INTMSK	; INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7834	040204	105037	003110		CLRB W.TIME	; RESET WATCH DOG TIMING ON THIS DRIVE
7835	040210	005037	003124		CLR W.DRV	; CLEAR TIMING COUNT FOR THIS DRIVE
7836	040214	004777	142640		JSR PC,QA.CONT	; INDICATE CONTROLLER ERROR RETURN
7837	040220	000207			RTS PC	; RETURN

7838
7839
7840
7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893

.SBTTL *COMMAND INITATOR

```
*****
*
* THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
* BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
* SPECIAL COMMAND ARE ALSO EXECUTED:
*
*      RELEASE
*      CONROLLER CLEAR
*      SUBSYSTEM CLEAR
*      READ ALL DRIVE STATUS
*      READ SPECIFIED HEADER
*
* THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS
*CALL JSR PC.C.INIT
*      <ADDRESS OF PARAMETER BLOCK>
*      RETURN
*
* FOR THE SEQUENTIAL OPERATIONS, THE DRIVER WILL LOAD THE
* LOCATIONS, PBLKT AND INTMSK.
*
* ROUTINES USED:
*      W.WTCH
*      I.CSTS
*      I.STAT
*      I.CCLR
*****
```

```
C.INIT: MOV R5, -(SP) ;STORE R5 ON STACK
        MOV R4, -(SP) ;STORE R4 ON STACK
        MOV R3, -(SP) ;STORE R3 ON STACK
        MOV R2, -(SP) ;STORE R2 ON STACK
        MOV R1, -(SP) ;STORE R1 ON STACK
        MOV R0, -(SP) ;STORE R0 ON STACK
        MOV PS, -(SP) ;STORE PSW ON STACK
        MOV RKPRI, PS ;LOCK OUT RK06 INTERRUPTS
        MOV @16(SP), R5 ;STORE PARAMETER BLOCK ADDRESS
        ADD #2, 16(SP) ;ADJUST RETURN
        MOV P.DRVN(R5), R4 ;STORE DRIVE NUMBER
        BIC #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
```

```

7894 040324 042765 07517L 000014 BIC #C<DRVUSE!W.WCK!NOCHK!DRPDV!DTBAII>,P.PRST(R5)
7895
7896 040332 010500 MOV R5,R0 ;STORE PARAMETER BLOCK ADDRESS
7897 040334 062700 ADD #P.CS1,R0 ;CALCULATE FIRST LOCATION TO BE CLEARED
7898 040340 010501 MOV R5,R1 ;STORE PARAMETER BLOCK ADDRESS
7899 040342 062701 ADD #P.EPAT,R1 ;CALCULATE LAST LOCATION TO BE CLEARED
7900
7901 040346 005020 1$: CLR (R0)+ ;CLEAR RETURN PARAMETER
7902 040350 020001 CMP R0,R1 ;CHECK IF FINISHED
7903 040352 101775 BLOS 1$ ;NO, CLEAR NEXT RETURN PARAMETER
7904 040354 105037 003104 CLRB I.ISRL ;CLEAR RELEASE OR INTERRUPT ISSUED
7905 040360 010465 000020 MOV R4,P.CS2(R5) ;STORE DRIVE NUMBER
7906 040364 005062 000026 CLR PKMR1(R2) ;CLEAR RK06 MAINTENANCE REGISTER 1
7907 040370 132765 000040 000001 BITB #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
7908 040376 001402 BEQ 3$ ;NO, PROCESS
7909 040400 000137 041114 JMP C.SPEC ;JUMP TO SPECIAL COMMAND PROCESSOR
7910
7911 040404 122765 000107 000001 3$: CMPB #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
7912 ; START SPINDLE
7913 ; RECALIBRATE
7914 ; OFFSET
7915 ; SEEK
7916 ; UNLOAD
7917
7918 040412 101174 BHI 25$ ;NO, DRIVE COMMAND
7919 ; SELECT DRIVE
7920 ; PACK ACKNOWLEDGE
7921 ; CLEAR
7922
7923 040414 122765 000117 000001 CMPB #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
7924 040422 103540 BLO 20$ ;YES, DATA TRANSFER COMMAND
7925 ; READ DATA
7926 ; WRITE DATA
7927 ; READ HEADER
7928 ; WRITE HEADER
7929 ; WRITE CHECK
7930 040424 016562 000020 000010 MOV P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
7931 040432 052765 000002 000014 BIS #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
7932 040440 005037 003064 CLR O.WAIT ;CLEAR WAIT FOR COMMAND
7933 040444 122765 000117 000001 CMPB #SEEK,P.CMND(R5) ;CHECK IF SEEK
7934 040452 001007 BNE 5$ ;NO, CHECK FOR OFFSET
7935 040454 016562 000002 000020 MOV P.CYLN(R5),R.DCYL(R2) ;LOAD CYLINDER ADDRESS
7936 040462 016562 000004 000006 MOV P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
7937 040470 000431 BR 8$ ;GO ISSUE COMMAND
7938
7939 040472 122765 000115 000001 5$: CMPB #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
7940 040500 001007 BNE 6$ ;NO, CHECK FOR UNLOAD
7941 040502 116565 000006 000032 MOV P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
7942 040510 016562 000032 000016 MOV P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
7943 040516 000416 BR 8$ ;GO ISSUE COMMAND
7944
7945 040520 122765 000111 000001 6$: CMPB #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
7946 040526 001003 BNE 7$ ;NO, CHECK IF RECAL
7947 040530 013737 003076 003124 MOV W.MIN,W.DRV ;LOAD WATCH DOG TIME FOR 1 MINUTE
7948 040536 122765 000113 000001 7$: CMPB #RECAL,P.CMND(R5) ;CHECK IF RECAL
7949 040544 001003 BNE 8$ ;NO, CONTINUE

```

```

7950 040546 013737 003074 003124      MOV      W.BSEC,W.DRV      ;LOAD RECAL TIME FOR 8 SECONDS
7951 040554 116565 000007 000017 8$:      MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7952 040562 042765 165777 000016      BIC      #C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
7953                                     ; AND DRIVE TYPE
7954 040570 116565 000001 000016      MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
7955 040576 042765 000200 000014      BIC      #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
7956 040604 032765 000400 000014      BIT      #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
7957 040612 001533 000100 000016      BEQ      30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
7958 040614 042765 000100 000016      BIC      #IE,P.CS1(R5) ;CLEAR INTERRUPT ENABLE
7959 040622 016562 000016 000000      MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
7960 040630 004737 034572 003010 10$:      JSR      PC,W.WTCH ;CALL WATCH DOG TIMER
7961 040634 016237 000000 003010      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7962 040642 032737 000200 003010      BIT      #RDY,T.CS1 ;WAIT FOR READY
7963 040650 001767 000000 003010      BEQ      10$
7964 040652 032737 100000 003010      BIT      #CERR,T.CS1 ;CHECK FOR ERROR
7965 040660 001011 000000 003010      BNE      15$              ;YES, GIVE NORMAL RETURN
7966 040662 004737 034572 003024 11$:      JSR      PC,W.WTCH ;CALL WATCH DOG TIMER
7967 040666 016237 000016 003024      MOV      RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7968 040674 133737 003111 003025      BITB     INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
7969 040702 001767 000000 003025      BEQ      11$              ;WAIT FOR DRIVE INTERRUPT
7970 040704 105037 003110 003025 15$:      CLRB     W.TIME ;RESET TIMING ON THIS DRIVE
7971 040710 005037 003124 003025      CLR      W.DRV ;CLEAR DRIVE TIMING COUNT
7972 040714 004737 040166 003025      JSR      PC,R.NORM ;INDICATE COMMAND IS FINISHED
7973 040720 000137 042074 003025      JMP      C.ATRN ;RESTORE REGISTERS
7974
7975 040724 016562 000010 000004 20$:      MOV      P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
7976 040732 016562 000012 000002      MOV      P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
7977 040740 016562 000002 000020      MOV      P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
7978 040746 016562 000004 000006      MOV      P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
7979 040754 122765 000131 000001      CMPB     #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
7980 040762 001010 000200 000014      BNE      25$              ;NO, GO ISSUE THE COMMAND
7981 040764 032765 000200 000014      BIT      #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
7982 040772 001404 000123 000016      BEQ      25$              ;NO, GO ISSUE THE COMMAND
7983 040774 012765 000123 000016      MOV      #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
7984 041002 000406 000123 000016      BR       26$              ;GO ISSUE COMMAND
7985
7986 041004 116565 000001 000016 25$:      MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
7987 041012 042765 000200 000014      BIC      #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
7988 041020 116565 000007 000017 26$:      MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7989 041026 142765 177750 000017      BICB     #C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
7990                                     ; FORMAT, DRIVE TYPE, AND BUS ADDRESS
7991                                     ; BITS 16-17
7992 041034 010537 003064 000014      MOV      R5,0.WAIT ;LOAD WAITING FOR COMMAND
7993 041040 032765 100000 000014      BIT      #DIBAI,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
7994 041046 001403 000000 000014      BEQ      27$              ;NO, LOAD CS2
7995 041050 052765 000020 000020      BIS      #BAI,P.CS2(R5) ;SET INHIBIT BUS ADDRESS INCREMENT
7996 041056 016562 000020 000010 27$:      MOV      P.CS2(R5),RKCS2(R2) ;LOAD CS2
7997 041064 032765 000400 000014      BIT      #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
7998 041072 001403 000100 000016      BEQ      30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
7999 041074 042765 000100 000016      BIC      #IE,P.CS1(R5) ;CLEAR INTERRUPT ENABLE
8000 041102 016562 000016 000000 30$:      MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
8001 041110 000137 042074 000000      JMP      C.ATRN ;RESTORE REGISTERS
8002
8003 .SBTTL *SPECIAL COMMAND PROCESSING
8004
8005 041114 122765 000141 000001 C.SPEC: CMPB     #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS

```

8006	041122	001132				BNE	10\$	: NO, PROCESS OTHER COMMANDS
8007	041124	016562	000020	000010		MOV	P.CS2(R5), RKCS2(R2)	: LOAD CS2 FOR COMMAND
8008	041132	116565	000007	000017		MOVB	P.CS1H(R5), P.CS1+1(R5)	: STORE BITS 8-15 OF CS1
8009	041140	042765	165777	000016		BIC	#1C<CFMT!COT>, P.CS1(R5)	: CLEAR ALL BITS EXCEPT FORMAT
8010								: AND DRIVE TYPE
8011	041146	112765	000001	000016		MOVB	#DR.SEL, P.CS1(R5)	: STORE COMMAND
8012	041154	016562	000016	000000		MOV	P.CS1(R5), RKCS1(R2)	: ISSUE COMMAND
8013	041162	004737	034572		2\$:	JSR	PC.W.WTCH	: CALL WATCH-DOG TIMER
8014	041166	016265	000000	000016		MOV	RKCS1(R2), P.CS1(R5)	: STORE COMMAND AND STATUS REG. 1
8015	041174	032765	000200	000016		BIT	#RDY, P.CS1(R5)	: WAIT FOR READY
8016	041202	001767				BEQ	2\$	
8017	041204	004737	037644			JSR	PC.I.CST1	: STORE CONTROLLER REGISTERS
8018	041210	016265	000034	000040		MOV	RKMR2(R2), P.A00(R5)	: STORE STATUS BYTE 00 MESSAGE A
8019	041216	016265	000036	000042		MOV	RKMR3(R2), P.B00(R5)	: STORE STATUS BYTE 00 MESSAGE B
8020	041224	032765	100000	000016		BIT	#CERR, P.CS1(R5)	: CHECK IF CONTROLLER ERROR
8021	041232	001436				BEQ	6\$	: NO, GATHER DRIVE STATUS
8022	041234	105037	003110			CLRB	W.TIME	: RESET WATCH DOG TIMING ON THIS DRIVE
8023	041240	005037	003124			CLR	W.DRV	: CLEAR WATCH DOG COUNT
8024	041244	032765	000400	000014		BIT	#NOCHK, P.PRST(R5)	: CHECK IF NO CHECK MODE
8025	041252	001043				BNE	8\$	: YES, INDICATE NORMAL RETURN
8026	041254	032765	0C1000	000020		BIT	#MDS, P.CS2(R5)	: CHECK IF MULTIPLE DRIVE SELECT
8027	041262	001043				BNE	9\$	: YES, INDICATE CONTROLLER ERROR
8028	041264	004037	037170			JSR	RD, I.CCLR	: CLEAR ERROR
8029	041270	042074				C.ATRN		: ERROR RETURN
8030	041272	032765	010400	000020		BIT	#NED!UFE, P.CS2(R5)	: CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
8031	041300	001007				BNE	5\$	: REPORT ERROR
8032	041302	032765	000001	000036		BIT	#DRA, P.DS(R5)	: CHECK IF DRIVE AVAILIABLE
8033	041310	001003				BNE	5\$	: YES, REPORT ERROR
8034	041312	052765	010000	000014		BIS	#DRVSZD, P.PRST(R5)	: INDICATE DRIVE IS SEIZED BY OTHER PORT
8035	041320	004737	040154		5\$:	JSR	PC.R.ABNL	: INDICATE ABNORMAL RETURN
8036	041324	000137	042074			JMP	C.ATRN	: RESTORE REGISTERS
8037								
8038	041330	004037	037726		6\$:	JSR	RD, I.STAT	: GATHER DRIVE STATUS
8039	041334	042074				C.ATRN		: ERROR RETURN
8040	041336	105037	003110			CLRB	W.TIME	: STOP WATCH-DOG TIMING ON DRIVE
8041	041342	005037	003124			CLR	W.DRV	: RESET WATCH-DOG TIME
8042	041346	032765	000400	000014		BIT	#NOCHK, P.PRST(R5)	: CHECK IF NO CHECK MODE
8043	041354	001402				BEQ	8\$	: NO, REPORT ERROR
8044	041356	005062	000000			CLR	RKCS1(R2)	: CLEAR INTERRUPT ENABLE
8045	041362	004737	040166		8\$:	JSP	PC.R.NORM	: REPORT COMMAND COMPLETE
8046	041366	000137	042074			JMP	C.ATRN	: RESTORE REGISTERS
8047								
8048	041372	052737	100000	003062	9\$:	BIS	#E.MDS, E.CONT	: SET MULTIPLE DRIVE SELECT
8049	041400	004737	040200			JSR	PC.R.CONT	: INDICATE CONTROLLER ERROR
8050	041404	000137	042074			JMP	C.ATRN	
8051								
8052	041410	122765	000140	000001	10\$:	CMPB	#RELEAS, P.CMND(R5)	: CHECK IF RELEASE COMMAND
8053	041416	001040				BNE	13\$	: NO, CHECK IF READ ALL HEADERS
8054	041420	010537	003064			MOV	R5, O.WAIT	: STORE PARAMETER BLOCK ADDRESS IN
8055								: WAIT FOR COMMAND
8056	041424	052765	000010	000020		BIS	#RLS, P.CS2(R5)	: SET RELEASE BIT
8057	041432	016562	000020	000010		MOV	P.CS2(R5), RKCS2(R2)	: LOAD CS2 FOR DESELECT
8058	041440	112737	000001	003104		MOVB	#1, I.ISRL	: SET FLAG FOR RELEASE COMMAND
8059	041446	116565	000007	000017		MOVB	P.CS1H(R5), P.CS1+1(R5)	: STORE BITS 8-15 OF CS1
8060	041454	042765	165777	000016		BIC	#1C<CFMT!COT>, P.CS1(R5)	: CLEAR ALL BITS EXCEPT FORMAT
8061								: AND DRIVE TYPE

```

8062 041462 112765 000101 000016      MOVB  #SELDIV,P.CS1(R5) ;STORE COMMAND
8063 041470 032765 000400 000014      BIT   #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8064 041476 001403          BEQ    11$ ;NO DO NOT RESET INTERRUPT ENABLE
8065 041500 042765 000100 000016      BIC   #IE,P.CS1(R5) ;RESET INTERRUPT ENABLE
8066 041506 016562 000016 000000 11$:  MOV   P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
8067 041514 000137 042074          JMP   C.RTRN ;RESTORE REGISTERS
8068
8069 041520 122765 000164 000001 13$:  CMPB  #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
8070 041526 001053          BNE    30$ ;NO CHECK IF CONTROLLER CLEAR
8071 041530 010537 003064          MOV   R5,0.WAIT ;SET WAITING FOR COMMAND COMPLETION
8072 041534 016537 000010 003100      MOV   P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
8073 041542 132765 000020 000007      BITB  #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
8074 041550 001404          BEQ    14$ ;YES, LOAD 22 IN HEADER COUNT
8075 041552 012737 000024 003102      MOV   #20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
8076 041560 000403          BR     22$ ;GO ISSUE READ HEADER COMMAND
8077
8078 041562 012737 000026 003102 14$:  MOV   #22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
8079 041570 016562 000002 000020 22$:  MOV   P.CYLH(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
8080 041576 016562 000004 000006      MOV   P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
8081 041604 016562 000020 000010      MOV   P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
8082 041612 116565 000007 000017      MOVB  P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
8083 041620 042765 165777 000016      BIC   #IC(CFMT!COT),P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
8084                                AND  FORMAT
8085 041626 112765 000125 000016      MOVB  #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
8086 041634 032765 000400 000014      BIT   #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8087 041642 001027          BNE    34$ ;YES, INDICATE ILLEGAL DRIVER COMMAND
8088 041644 016562 000016 000000      MOV   P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
8089 041652 000137 042074          JMP   C.RTRN ;RESTORE REGISTERS
8090
8091 041656 122765 000176 000001 30$:  CMPB  #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
8092 041664 001012          BNE    32$ ;NO CHECK IF SUBSYSTEM CLEAR
8093 041666 004037 037170          JSR   RD,I.CCLR ;CLEAR CONTROLLER
8094 041672 042074          C.RTRN ;ERROR RETURN
8095 041674 032765 000400 000014      BIT   #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8096 041702 001472          BEQ    40$ ;NO, INDICATE NORMAL RETURN
8097 041704 005062 000000          CLR   RKCS1(R2) ;RESET INTERRUPT ENABLE
8098 041710 000467          BR     40$ ;INDICATE NORMAL RETURN
8099
8100 041712 122765 000177 000001 32$:  CMPB  #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
8101 041720 001406          BEQ    36$ ;YES, CLEAR SUBSYSTEM
8102 041722 052737 000100 003062 34$:  BIS   #E.ILLD,E.CONT ;SET ILLEGAL DRIVER COMMAND
8103 041730 004737 040200          JSR   PC,R.CONT ;REPORT ERROR
8104 041734 000457          BR     C.RTRN ;RESTORE REGISTERS
8105
8106 041736 012762 000040 000010 36$:  MOV   #SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
8107 041744 016265 000000 000016      MOV   RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
8108 041752 032765 100000 000016      BIT   #CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
8109 041760 001406          BEQ    37$ ;NO, FINISH COMMAND
8110 041762 052737 000001 003062      BIS   #BITD,E.CONT ;SET CLEAR SUBSYSTEM DID NOT CLEAR
8111                                ;CONTROLLER ERROR
8112 041770 004737 040200          JSR   PC,R.CONT ;REPORT ERROR
8113 041774 000437          BR     C.RTRN ;RESTORE REGISTERS
8114
8115 041776 013746 003070          37$:  MOV   W.MILI,-(SP) ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
8116                                ;TO DISAPPEAR
8117 042002 016265 000000 000016 38$:  MOV   RKCS1(R2),P.CS1(R5) ;STORE CS1

```

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 162
*SPECIAL COMMAND PROCESSING

SEQ 0161

8118	042010	032765	040000	000016	BIT	#DI,P.CS1(R5)	;CHECK IF ATTENTIONS CLEARED
8119	042016	001411			BEQ	39\$	;YES, FINISH COMMAND
8120	042020	005316			DEC	(SP)	;DECREMENT 16 MILLISECOND COUNT
8121	042022	001367			BNE	38\$	;CHECK DRIVE INTERRUPT AGAIN
8122	042024	005726			TST	(SP)+	;ADJUST STACK
8123	042026	052737	000040	003062	BIS	#E.SCLR,E.CONT	;SET SUBSYSTEM CLEAR DID NOT CLEAR
8124							;DRIVE ATTENTIONS
8125	042034	004737	040200		JSR	PC,R.CONT	;REPORT ERROR
8126	042040	000415			BR	C.ATRN	;RESTORE REGISTER
8127							
8128	042042	005726			39\$: TST	(SP)+	;ADJUST STACK
8129	042044	032765	000400	000014	BIT	#NOCHK,P.PRST(R5)	;CHECK IF NO CHECK MODE
8130	042052	001010			BNE	C.RTRN	;YES, RESTORE REGISTERS
8131	042054	112737	177777	003104	MOVB	#-1,I.ISRL	;SET INTERRUPT ENABLE SET
8132	042062	012762	000100	000000	MOV	#IE,RKCS1(R2)	;SET INTERRUPT ENABLE
8133	042070	004737	040166		40\$: JSR	PC,R.NORM	;INDICATE NORMAL TERMINATION
8134							
8135	042074	012637	177776		C.RTRN: MOV	(SP)+,PS	;RESTORE PSW
8136	042100	012600			MOV	(SP)+,R0	;RESTORE R0
8137	042102	012601			MOV	(SP)+,R1	;RESTORE R1
8138	042104	012602			MOV	(SP)+,R2	;RESTORE R2
8139	042106	012603			MOV	(SP)+,R3	;RESTORE R3
8140	042110	012604			MOV	(SP)+,R4	;RESTORE R4
8141	042112	012605			MOV	(SP)+,R5	;RESTORE R5
8142	042114	000207			RTS	PC	;RETURN
8143					.SBTTL	OCTAL TO BINARY CONVERSION ROUTINE	
8144							
8145							
8146							
8147							
8148							
8149							
8150							
8151							
8152							
8153							
8154							
8155							
8156							
8157							
8158							
8159							
8160	042116	010046			OCTBIN: MOV	R0,-(SP)	;SAVE R0
8161	042120	010146			MOV	R1,-(SP)	;SAVE R1
8162	042122	010246			MOV	R2,-(SP)	;SAVE R2
8163	042124	016600	000010		MOV	10(SP),R0	;GET ADDRESS OF ASCII STRING
8164	042130	005001			CLR	R1	;CLEAR DATA WORDS
8165	042132	005002			CLR	R2	
8166	042134	112046			2\$: MOVB	(R0)+,-(SP)	;PICK THIS CHARACTER
8167	042136	001423			BEQ	3\$	;IF ZERO GET OUT
8168	042140	121627	000054		CMPB	(SP),#'	;CHECK IF COMMA
8169	042144	001420			BEQ	3\$	;IF COMMA GET OUT
8170	042146	122716	000060		CMPB	#'0,(SP)	;MAKE SURE THIS CHARACTER IS
8171	042152	003020			BGT	4\$	;AN OCTAL DIGIT
8172	042154	122716	00006		CMPB	#'7,(SP)	
8173	042160	002425			BLT	4\$	

```

8174 042162 006301      ASL      R1      ; *2
8175 042164 006102      ROL      R2
8176 042166 006301      ASL      R1      ; *4
8177 042170 006102      ROL      R2
8178 042172 006301      ASL      R1      ; *8
8179 042174 006102      ROL      R2
8180 042176 042716 177770 BIC      #1C7,(SP) ; STRIP THE ASCII JUNK
8181 042202 062601      ADD      (SP)+,R1 ; ADD THIS DIGIT
8182 042204 000753      BR       2$      ; LOOP
8183 042206 005726      3$: TST      (SP)+ ; CLEAN PARTIAL FROM STACK
8184 042210 010166 000010 MOV      R1,10(SP) ; SAVE RESULT
8185 042214 010237 042250 MOV      R2,$HI OCT
8186 042220 012602      MOV      (SP)+,R2 ; RESTORE R2
8187 042222 012601      MOV      (SP)+,R1 ; RESTORE R1
8188 042224 012600      MOV      (SP)+,R0 ; RESTORE R0
8189 042226 062716 000002 ADD      #2,(SP) ; ADJUST RETURN
8190 042232 000207      RTS      PC      ; RETURN
8191
8192 042234 005726      4$: TST      (SP)+ ; CLEAN UP PARTIAL FROM STACK
8193 042236 012602      MOV      (SP)+,R2 ; RESTORE R2
8194 042240 012601      MOV      (SP)+,R1 ; RESTORE R1
8195 042242 012600      MOV      (SP)+,R0 ; RESTORE R0
8196 042244 013616      MOV      2(SP)+,(SP) ; PUT ADDRESS OF ERROR ROUTINE ON STACK
8197 042246 000207      RTS      PC      ; GO PROCESS ERROR
8198 042250 000000      $HI OCT: .WORD 0 ; HIGH ORDER BITS GO HERE
8199
8200      .SBTTL RANDOM NUMBER GENERATOR ROUTINE
8201
8202      ;*****
8203      ;THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
8204      ;WITH A RANGE OF 0 TO 2(+33)-1.
8205      ;CALL:
8206      ;*      JSR      PC,$RAND      ; CALL THE ROUTINE
8207      ;*      RETURN      ; RETURN HERE THE RANDOM
8208      ;*      ;NUMBER WILL BE IN
8209      ;*      ;$HINUM,$LONUM
8210
8211 042252      $RAND: MOV      R0,-(SP) ; PUSH R0 ON STACK
8212 042254 010146      MOV      R1,-(SP) ; PUSH R1 ON STACK
8213 042256 010246      MOV      R2,-(SP) ; PUSH R2 ON STACK
8214 042260 013700 042352 MOV      $LONUM,R0 ; SET R0 WITH LOW
8215 042264 013701 042350 MOV      $HINUM,R1 ; SET R1 WITH HIGH
8216 042270 012702 177771 MOV      #-7,R2 ; SET SHIFT COUNT
8217 042274 006300      1$: ASL      R0 ; SHIFT R0 LEFT AND
8218 042276 006101      ROL      R1 ; ROTATE CARRY INTO R1 AND
8219 042300 005202      INC      R2 ; CHECK FOR DONE
8220 042302 001374      BNE      1$      ; CONTINUE SHIFT LOOP
8221 042304 063700 042352 ADD      $LONUM,R0 ; ADD NUMBER TO MAKE X 129
8222 042310 005501      ADC      R1 ; PROPOGATE CARRY
8223 042312 063701 042350 ADD      $HINUM,R1 ; ADD NUMBER TO MAKE X 129
8224 042316 062700 001057 ADD      #1057,R0 ; ADD LOW CONSTANT
8225 042322 005501      ADC      R1 ; PROPOGATE CARRY
8226 042324 062701 047401 ADD      #47401,R1 ; ADD HIGH CONSTANT
8227 042330 010037 042352 MOV      R0,$LONUM ; SAVE R0
8228 042334 010137 042350 MOV      R1,$HINUM ; SAVE R1
8229 042340 012602      MOV      (SP)+,R2 ; POP STACK INTO R2

```

```

8230 042342 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
8231 042344 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
8232 042346 000207      RTS      PC          ;;RETURN
8233 042350 176543      $HINUM: .WORD 176543
8234 042352 123456      $LONUM: .WORD 123456
8235                                     .SBTTL  TYPE ROUTINE
8236
8237                                     ;*****
8238                                     ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8239                                     ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8240                                     ;NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
8241                                     ;NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
8242                                     ;NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
8243                                     ;*
8244                                     ;*CALL:
8245                                     ;*1) USING A TRAP INSTRUCTION
8246                                     ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
8247                                     ;*OR
8248                                     ;*      TYPE
8249                                     ;*      MESADR
8250                                     ;*
8251
8252 042354 105737 001157      $TYPE:  TSTB      $TPFLG      ;; IS THERE A TERMINAL?
8253 042360 100002          BPL      1$              ;; BR IF YES
8254 042362 000000          HALT          ;; HALT HERE IF NO TERMINAL
8255 042364 000430          BR      3$              ;; LEAVE
8256 042366 010046          1$:      MOV      R0,-(SP)      ;; SAVE R0
8257 042370 017600          MOV      32(SP),R0      ;; GET ADDRESS OF ASCIZ STRING
8258 042374 122737 000001 001350      CMPB      #APTENV,$ENV      ;; RUNNING IN APT MODE
8259 042402 001011          BNE      62$          ;; NO GO CHECK FOR APT CONSOLE
8260 042404 132737 000100 001351      BITB      #APTSPool,$ENVM      ;; SPOOL MESSAGE TO APT
8261 042412 001405          BEQ      62$          ;; NO GO CHECK FOR CONSOLE
8262 042414 010037 042424          MOV      R0,61$      ;; SETUP MESSAGE ADDRESS FOR APT
8263 042420 004737 044764          JSR      PC,$ATY3      ;; SPOOL MESSAGE TO APT
8264 042424 000000          61$:      .WORD 0          ;; MESSAGE ADDRESS
8265 042426 132737 000040 001351      62$:      BITB      #APTCSUP,$ENVM      ;; APT CONSOLE SUPPRESSED
8266 042434 001003          BNE      60$          ;; YES, SKIP TYPE OUT
8267 042436 112046          2$:      MOVB      (R0)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
8268 042440 001005          BNE      4$          ;; BR IF IT ISN'T THE TERMINATOR
8269 042442 005726          TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
8270 042444 012600          60$:      MOV      (SP)+,R0      ;; RESTORE R0
8271 042446 062716 000002          3$:      ADD      #2,(SP)      ;; ADJUST RETURN PC
8272 042452 000002          RTI          ;; RETURN
8273 042454 122716 000011          4$:      CMPB      #HT,(SP)      ;; BRANCH IF <HT>
8274 042460 001430          BEQ      8$          ;; BRANCH IF NOT <CRLF>
8275 042462 122716 000200          CMPB      #CRLF,(SP)
8276 042466 001006          BNE      5$          ;; POP <CR><LF> EQUIV
8277 042470 005726          TST      (SP)+      ;; TYPE A CR AND LF
8278 042472 104401          TYPE
8279 042474 001325          $CRLF
8280 042476 105037 042632          CLRB      $CHARCNT      ;; CLEAR CHARACTER COUNT
8281 042502 000755          BR      2$          ;; GET NEXT CHARACTER
8282 042504 004737 042566          5$:      JSR      PC,$TYPEC      ;; GO TYPE THIS CHARACTER
8283 042510 123726 001156          6$:      CMPB      $FILLC,(SP)+      ;; IS IT TIME FOR FILLER CHARS.?
8284 042514 001350          BNE      2$          ;; IF NO GO GET NEXT CHAR.
8285 042516 013746 001154          MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED

```

```

8286
8287 042522 105366 000001 7$: DEC B 1(SP) ; AND THE NULL CHAR.
8288 042526 002770 6$ ; DOES A NULL NEED TO BE TYPED?
8289 042530 004737 042566 JSR PC,$TYPEC ; BR IF NO--GO POP THE NULL OFF OF STACK
8290 042534 105337 042632 DEC B $CHARCNT ; GO TYPE A NULL
8291 042540 000770 BR 7$ ; DO NOT COUNT AS A COUNT
8292 ; LOOP
8293
8294 ; HORIZONTAL TAB PROCESSOR
8295 042542 112716 000040 8$: MOV B #' (SP) ; REPLACE TAB WITH SPACE
8296 042546 004737 042566 9$: JSR PC,$TYPEC ; TYPE A SPACE
8297 042552 132737 000007 042632 BIT B #',$CHARCNT ; BRANCH IF NOT AT
8298 042560 001372 BNE 9$ ; TAB STOP
8299 042562 005726 TST (SP)+ ; POP SPACE OFF STACK
8300 042564 000724 BR 2$ ; GET NEXT CHARACTER
8301 042566 105777 136356 $TYPEC: TST B @STPS ; WAIT UNTIL PRINTER IS READY
8302 042572 100375 BPL $TYPEC
8303 042574 116677 000002 136350 MOV B 2(SP),@STPB ; LOAD CHAR TO BE TYPED INTO DATA REG.
8304 042602 122766 000015 000002 CMP B #CR,2(SP) ; IS CHARACTER A CARRIAGE RETURN?
8305 042610 001003 BNE 1$ ; BRANCH IF NO
8306 042612 105037 042632 CLRB $CHARCNT ; YES--CLEAR CHARACTER COUNT
8307 042616 000406 BR $TYPEX ; EXIT
8308 042620 122766 000012 000002 1$: CMP B #LF,2(SP) ; IS CHARACTER A LINE FEED?
8309 042626 001402 BEQ $TYPEX ; BRANCH IF YES
8310 042630 105227 INCB (PC)+ ; COUNT THE CHARACTER
8311 042632 000000 $CHARCNT: .WORD 0 ; CHARACTER COUNT STORAGE
8312 042634 000207 $TYPEX: RTS PC
8313
8314
8315
8316
8317 ; *****
8318 ; SBTTL DOUBLE-PRECISION MULTIPLY SUBROUTINE
8319 ; * SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
8320 ; * USES ALL REGISTERS (R0-R5)
8321 ; *
8322 ; * ENTER WITH JSR PC,M.DPIM
8323 ; * MULTIPLIER IN R2-R3
8324 ; * MULTIPLICAND IN R4-R5
8325 ; * PRODUCT RETURNED IN R0-R1-R2-R3
8326 ; *
8327 ; *****
8328 042636 005000 M.DPIM: CLR R0 ; CLEAR HI ORDER WORDS
8329 042640 005001 CLR R1
8330 042642 012746 000041 MOV #41,-(SP) ; MOVE 33 (DEC) TO COUNTER
8331 042646 006000 M.DP01: ROR R0
8332 042650 006001 ROR R1
8333 042652 006002 ROR R2 ; SHIFT TO ADD
8334 042654 006003 ROR R3
8335 042656 103003 BCC M.DP02 ; NO CARRY NO ADD
8336 042660 060501 ADD R5,R1
8337 042662 005500 ADC R0 ; ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
8338 042664 060400 ADD R4,R0 ; PRODUCT
8339 042666 005316 M.DP02: DEC @SP ; DECREMENT COUNTER
8340 042670 001366 BNE M.DP01
8341 042672 005726 TST (SP)+ ; REMOVE THE COUNTER

```

8342 042674 000207

RTS PC

8343
8344
8345
8346
8347
8348
8349
8350
8351
8352
8353
8354
8355
8356
8357 042676 012746 000040
8358 042702 010446
8359 042704 010546
8360 042706 005466 000002
8361 042712 005416
8362 042714 005666 000002
8363 042720 061601
8364 042722 005500
8365 042724 066600 000002
8366 042730 103445
8367 042732 005046
8368 042734 006103
8369 042736 006102
8370 042740 006101
8371 042742 006100
8372 042744 005716
8373 042746 001410
8374 042750 005016
8375 042752 066601 000002
8376 042756 005500
8377 042760 005516
8378 042762 066600 000004
8379 042766 000404
8380 042770 060501
8381 042772 005500
8382 042774 005516
8383 042776 060400
8384 043000 005516
8385 043002 005716
8386 043004 001401
8387 043006 005203
8388 043010 005366 000006
8389 043014 003347
8390 043016 006003
8391 043020 103404
8392 043022 060501
8393 043024 005500
8394 043026 060400
8395 043030 000241
8396 043032 006103
8397 043034 062706 000010

```

*****
: 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 2(SP) ; VERSION OF THE DIVISOR
        SBC 2(SP)
        ADD 2(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 2(SP) ; TEST "CARRY INDICATOR"
        BEQ M.DP41 ; IF NO "CARRY" THEN ADD ELSE SUBTRACT
        CLR 2(SP) ; CLEAR UP FOR NEXT TIME
        ADD 2(SP), R1
        ADC R0 ; ADD -(DIVISOR)
        ADC 2(SP) ; SET "CARRY"
        ADD 4(SP), R0 ; I
        BR M.DP42
M.DP41: ADD R5, R1 ; ADD +(DIVISOR)
        ADC R0 ; SET "CARRY"
        ADC 2(SP) ; I
        ADD R4, R0 ; I
M.DP42: ADC 2(SP) ; SET "CARRY"
        TST 2(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
        CLC
M.DP44: ROL R3
        ADD #10, SP ; ADJUST STACK BY 4 WORDS

```

8398 043040 000242
8399 043042 000207
8400 043044 062706 000006
8401 043050 000262
8402 043052 000207

CLV
RTS PC
M.DPSO: 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

8417 043054 104407
8418 043056 016601 000002
8419 043062 012705 043173
8420 043066 012704 000014
8421 043072 012703 177770
8422 043076 012100
8423 043100 012101
8424 043102 005002
8425 043104 110245
8426 043106 010002
8427 043110 005304
8428 043112 003007
8429 043114 001405
8430 043116 005205
8431 043120 010566 000002
8432 043124 104410
8433 043126 000207
8434 043130 006203
8435 043132 006001
8436 043134 006000
8437 043136 006001
8438 043140 006000
8439 043142 006001
8440 043144 006000
8441 043146 040302
8442 043150 062702 000060
8443 043154 000753
8444 043156 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 #107, R3 ;; MASK
MOV (R1)+, R0 ;; LOWER WORD
MOV (R1)+, R1 ;; HIGH WORD
CLR R2 ;; TERMINATOR
1\$: MOV 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 R3, R2 ;; MASK OUT ALL JUNK
ADD #10, 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

```

8454          ;*      RETURN          ;;THE FIRST ADDRESS OF ASCII
8455          ;;IS ON THE STACK
8456
8457
8458 043174 104407 $DBED: SAVREG          ;;SAVE REGISTERS
8459 043176 016602 MOV 2(SP),R2          ;;PICKUP THE DATA POINTER
8460 043202 012700 MOV $DECVL,R0          ;;GET ADDRESS OF "$DECVL" STRING
8461 043206 010066 MOV R0,2(SP)          ;;PUT ADDRESS OF ASCII STRING ON STACK
8462 043212 012201 MOV (R2)+,R1          ;;PICKUP THE BINARY NUMBER
8463 043214 012202 MOV (R2)+,R2
8464 043216 012737 MOV #10.,4$          ;;SET UP TO DO 10 CONVERSIONS
8465 043224 012704 MOV $TNPWR,R4          ;;ADDRESS OF TEN POWER
8466 043230 012705 MOV $TNPWR+2,R5
8467 043234 005003 1$: CLR R3          ;;CLEAR PARTIAL
8468 043236 161401 2$: SUB (R4),R1          ;;SUBTRACT TEN POWER
8469 043240 005602 SBC R2
8470 043242 161502 SUB (R5),R2
8471 043244 002402 BLT 3$          ;;BR IF TEN POWER TO LARGE
8472 043246 005203 INC R3          ;;ADD 1 TO PARTIAL
8473 043250 000772 BR 2$          ;;LOOP
8474 043252 062401 3$: ADD (R4)+,R1          ;;RESTORE SUBTRACTED VALUE
8475 043254 005502 ADC R2
8476 043256 062402 ADD (R4)+,R2
8477 043260 022525 CMP (R5)+,(R5)+          ;;MOVE TO NEXT TEN POWER
8478 043262 052703 BIS #0,R3          ;;CHANGE PARTIAL TO ASCII
8479 043266 110320 MOVB R3,(R0)+          ;;SAVE IT
8480 043270 005327 DEC (PC)+          ;;DONE?
8481 043272 000000 4$: .WORD 0
8482 043274 001357 BNE 1$          ;;BR IF NO
8483 043276 105020 CLRB (R0)+          ;;TERMINATOR
8484 043300 104410 RESREG          ;;RESTORE REGISTERS
8485 043302 000207 RTS PC          ;;RETURN
8486 043304 145000 $TNPWR: 145000          ;;1.0E09
8487 043306 035632 35632          ;;1.0E08
8488 043310 160400 160400
8489 043312 002765 2765          ;;1.0E07
8490 043314 113200 113200
8491 043316 000230 230          ;;1.0E06
8492 043320 041100 041100
8493 043322 000017 17          ;;1.0E05
8494 043324 103240 103240
8495 043326 000001 1          ;;1.0E04
8496 043330 023420 23420
8497 043332 000000 0          ;;1.0E03
8498 043334 001750 1750
8499 043336 000000 0          ;;1.0E02
8500 043340 000144 144
8501 043342 000000 0          ;;1.0E01
8502 043344 000012 12
8503 043346 000000 0          ;;1.0E00
8504 043350 000001 1
8505 043352 000000 0
8506 043354 000014 $DECVL: .BLKB 12.          ;;RESERVE STORAGE FOR ASCII STRING
8507 .SBTTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
8508
8509 ;:*****

```

```

8510      ;*THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
8511      ;*LEADING NUMBERS.
8512      ;*CALL
8513      ;*      MOV      #NUMADR, -(SP)      ;; FIRST ADDRESS OF ASCII STRING
8514      ;*      JSR      PC, @#SSUPRS
8515
8516      $SUPRS: MOV      RO, -(SP)      ;; SAVE RO
8517      043370 010046      MOV      4(SP), RO      ;; PICKUP THE POINTER
8518      043372 016600 000004      TSTB      (RO)      ;; TERMINATE OR?
8519      043376 105710      BEQ      2$      ;; BR IF YES
8520      043400 001403      CMPB      #'0, (RO)+      ;; IS THIS AN ASCII "0" ?
8521      043402 122720 000060      BEQ      1$      ;; BR IF YES
8522      043406 001773      DEC      RO      ;; BACKUP BY "1"
8523      043410 005300      MOV      RO, 3$      ;; SAVE FOR TYPING
8524      043412 010037 043420      TYPE      0      ;; GO TYPE
8525      043416 104401      .WORD      0      ;; ASCII POINTER GOES HERE
8526      043420 000000      MOV      (SP)+, RO      ;; RESTORE RO
8527      043422 012600      MOV      (SP)+, (SP)      ;; RESTORE THE STACK
8528      043424 012616      RTS      PC      ;; RETURN
8529      043426 000207      .SBTTL      BINARY TO OCTAL (ASCII) AND TYPE
8530
8531      ;*****
8532      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
8533      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
8534      ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
8535      ;*CALL:
8536      ;*      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
8537      ;*      TYPOS      ;; CALL FOR TYPEOUT
8538      ;*      .BYTE      N      ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
8539      ;*      .BYTE      M      ;; M=1 OR 0
8540      ;*      ;; 1=TYPE LEADING ZEROS
8541      ;*      ;; 0=SUPPRESS LEADING ZEROS
8542
8543      ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
8544      ;*$TYPOS OR $TYPOC
8545      ;*CALL:
8546      ;*      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
8547      ;*      TYPON      ;; CALL FOR TYPEOUT
8548
8549      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
8550      ;*CALL:
8551      ;*      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
8552      ;*      TYPOC      ;; CALL FOR TYPEOUT
8553
8554      8555 043430 017646 000000 043653 $TYPOS: MOV      @ (SP), -(SP)      ;; PICKUP THE MODE
8556      043434 116637 000001      MOV      1(SP), $OFILL      ;; LOAD ZERO FILL SWITCH
8557      043442 112637 043655      MOV      (SP)+, $OMODE+1      ;; NUMBER OF DIGITS TO TYPE
8558      043446 062716 000002      ADD      #2, (SP)      ;; ADJUST RETURN ADDRESS
8559      043452 000406      BR      $TYPON
8560      043454 112737 000001 043653 $TYPOC: MOV      #1, $OFILL      ;; SET THE ZERO FILL SWITCH
8561      043462 112737 000006 043655      MOV      #6, $OMODE+1      ;; SET FOR SIX(6) DIGITS
8562      043470 112737 000005 043652 $TYPON: MOV      #5, $OCNT      ;; SET THE ITERATION COUNT
8563      043476 010346      MOV      R3, -(SP)      ;; SAVE R3
8564      043500 010446      MOV      R4, -(SP)      ;; SAVE R4
8565      043502 010546      MOV      R5, -(SP)      ;; SAVE R5

```

```

8566 043504 113704 043655      MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
8567 043510 005404      NEG      R4
8568 043512 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
8569 043516 110437 043654      MOVB    R4,$OMODE      ;;SAVE IT FOR USE
8570 043522 113704 043653      MOVB    $OFILL,R4      ;;GET THE ZERO FILL SWITCH
8571 043526 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
8572 043532 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
8573 043534 006105      1$:    ROL      R5      ;;ROTATE MSB INTO "C"
8574 043536 000404      BR       2$      ;;GO DO MSB
8575 043540 006105      2$:    ROL      R5      ;;FORM THIS DIGIT
8576 043542 006105      ROL      R5
8577 043544 006105      ROL      R5
8578 043546 010503      MOV      R5,R3
8579 043550 006103      3$:    ROL      R3      ;;GET LSB OF THIS DIGIT
8580 043552 105337 043654      DECB    $OMODE      ;;TYPE THIS DIGIT?
8581 043556 100016      BPL      7$      ;;BR IF NO
8582 043560 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
8583 043564 001002      BNE      4$      ;;TEST FOR 0
8584 043566 005704      TST      R4      ;;SUPPRESS THIS 0?
8585 043570 001403      BEQ      5$      ;;BR IF YES
8586 043572 005204      4$:    INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
8587 043574 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
8588 043600 052703 000040      5$:    BIS      #'',R3      ;;MAKE ASCII IF NOT ALREADY
8589 043604 110337 043650      MOVB    R3,$$      ;;SAVE FOR TYPING
8590 043610 104401 043650      TYPE     8$      ;;GO TYPE THIS DIGIT
8591 043614 105337 043652      7$:    DECB    $OCNT      ;;COUNT BY 1
8592 043620 003347      BGT      2$      ;;BR IF MORE TO DO
8593 043622 002402      BLT      6$      ;;BR IF DONE
8594 043624 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
8595 043626 000744      BR       2$      ;;GO DO THE LAST DIGIT
8596 043630 012605      6$:    MOV      (SP)+,R5      ;;RESTORE R5
8597 043632 012604      MOV      (SP)+,R4      ;;RESTORE R4
8598 043634 012603      MOV      (SP)+,R3      ;;RESTORE R3
8599 043636 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
8600 043644 012616      MOV      (SP)+,(SP)
8601 043646 000002      RTI
8602 043650      8$:    .BYTE    0      ;;RETURN
8603 043651      .BYTE    0      ;;STORAGE FOR ASCII DIGIT
8604 043652      .BYTE    0      ;;TERMINATOR FOR TYPE ROUTINE
8605 043653      .BYTE    0      ;;OCTAL DIGIT COUNTER
8606 043654 000000      .WORD    0      ;;ZERO FILL SWITCH
8607      .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE      ;;NUMBER OF DIGITS TO TYPE
8608
8609
8610      ;*****
8611      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
8612      ;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
8613      ;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
8614      ;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
8615      ;REPLACED WITH SPACES.
8616      ;CALL:
8617      ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
8618      ;*      TYPDS      ;;GO TO THE ROUTINE
8619
8620      $TYPDS:  MOV      R0,-(SP)      ;;PUSH R0 ON STACK
8621      MOV      R1,-(SP)      ;;PUSH R1 ON STACK

```

```

8622 043662 010246      MOV      R2,-(SP)      ;; PUSH R2 ON STACK
8623 043664 010346      MOV      R3,-(SP)      ;; PUSH R3 ON STACK
8624 043666 010546      MOV      R5,-(SP)      ;; PUSH R5 ON STACK
8625 043670 012746      MOV      #20200,-(SP)  ;; SET BLANK SWITCH AND SIGN
8626 043674 016605      MOV      20(SP),R5     ;; GET THE INPUT NUMBER
8627 043700 100004      BPL      1$          ;; BR IF INPUT IS POS.
8628 043702 005405      NEG      R5          ;; MAKE THE BINARY NUMBER POS.
8629 043704 112766      MOV      #'-,1(SP)     ;; MAKE THE ASCII NUMBER NEG.
8630 043712 005000      CLR      R0          ;; ZERO THE CONSTANTS INDEX
8631 043714 012703      MOV      $DBLK,R3      ;; SETUP THE OUTPUT POINTER
8632 043720 112723      MOV      #'',(R3)+     ;; SET THE FIRST CHARACTER TO A BLANK
8633 043724 005002      CLR      R2          ;; CLEAR THE BCD NUMBER
8634 043726 016001      MOV      $DTBL(R0),R1    ;; GET THE CONSTANT
8635 043732 160105      SUB      R1,R5        ;; FORM THIS BCD DIGIT
8636 043734 002402      BLT      4$          ;; BR IF DONE
8637 043736 005202      INC      R2          ;; INCREASE THE BCD DIGIT BY 1
8638 043740 000774      BR      3$          ;;
8639 043742 060105      ADD      R1,R5        ;; ADD BACK THE CONSTANT
8640 043744 005702      TST      R2          ;; CHECK IF BCD DIGIT=0
8641 043746 001002      BNE      5$          ;; FALL THROUGH IF 0
8642 043750 105716      TSTB     (SP)         ;; STILL DOING LEADING 0'S?
8643 043752 100407      BMI      7$          ;; BR IF YES
8644 043754 106316      ASLB     (SP)         ;; MSD?
8645 043756 103003      BCC      6$          ;; BR IF NO
8646 043760 116663      MOV      1(SP),-1(R3)    ;; YES--SET THE SIGN
8647 043766 052702      BIS      #'0,R2      ;; MAKE THE BCD DIGIT ASCII
8648 043772 052702      BIS      #' ,R2      ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
8649 043776 110223      MOV      R2,(R3)+     ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
8650 044000 005720      TST      (R0)+       ;; JUST INCREMENTING
8651 044002 020027      CMP      R0,#10      ;; CHECK THE TABLE INDEX
8652 044006 002746      BLT      2$          ;; GO DO THE NEXT DIGIT
8653 044010 003002      BGT      8$          ;; GO TO EXIT
8654 044012 010502      MOV      R5,R2        ;; GET THE LSD
8655 044014 000764      BR      6$          ;; GO CHANGE TO ASCII
8656 044016 105726      TSTB     (SP)+       ;; WAS THE LSD THE FIRST NON-ZERO?
8657 044020 100003      BPL      9$          ;; BR IF NO
8658 044022 116663      MOV      -1(SP),-2(R3)  ;; YES--SET THE SIGN FOR TYPING
8659 044030 105013      CLRB     (R3)         ;; SET THE TERMINATOR
8660 044032 012605      MOV      (SP)+,R5     ;; POP STACK INTO R5
8661 044034 012603      MOV      (SP)+,R3     ;; POP STACK INTO R3
8662 044036 012602      MOV      (SP)+,R2     ;; POP STACK INTO R2
8663 044040 012601      MOV      (SP)+,R1     ;; POP STACK INTO R1
8664 044042 012600      MOV      (SP)+,R0     ;; POP STACK INTO R0
8665 044044 104401      TYPE      $DBLK      ;; NOW TYPE THE NUMBER
8666 044050 016666      MOV      2(SP),4(SP)  ;; ADJUST THE STACK
8667 044056 012616      MOV      (SP)+,(SP)
8668 044060 000002      RTI
8669 044062 023420      $DTBL: 10000.
8670 044064 001750      1000.
8671 044066 000144      100.
8672 044070 000012      10.
8673 044072 000004      $DBLK: .BLKW 4
8674      .SBTTL  ERROR HANDLER ROUTINE
8675
8676      ;;*****
8677      ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,

```

```

8678 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
8679 ;*AND GO TO TYPERR ON ERROR
8680 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
8681 ;*SW15=1 HALT ON ERROR
8682 ;*SW13=1 INHIBIT ERROR TYPEOUTS
8683 ;*SW10=1 BELL ON ERROR
8684 ;*SW09=1 LOOF ON ERROR
8685 ;*CALL
8686 ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
8687
8688 $ERROR:
8689 7$: INCB      $ERFLG      ;;SET THE ERROR FLAG
8690      BEQ      7$          ;;DON'T LET THE FLAG GO TO ZERO
8691      MOV      $TSTNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
8692      BIT      $BIT'0,$SWR   ;;BELL ON ERROR?
8693      BEQ      1$          ;;NO - SKIP
8694      TYPE     $BELL        ;;RING BELL
8695      INC      $ERTTL       ;;COUNT THE NUMBER OF ERRORS
8696      MOV      (SP),$ERRPC  ;;GET ADDRESS OF ERROR INSTRUCTION
8697      SUB      #2,$ERRPC
8698      MOVB     $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
8699      BIT      $BIT13,$SWR  ;;SKIP TYPEOUT IF SET
8700      BNE      20$          ;;SKIP TYPEOUTS
8701      JSR      PC,TYPERR    ;;GO TO USER ERROR ROUTINE
8702      TYPE     $CRLF
8703
8704 20$: CMPB     $APTENV,$ENV  ;;RUNNING IN APT MODE
8705      BNE      2$          ;;NO SKIP APT ERROR REPORT
8706      MOVB     $ITEMB,$1$   ;;SET ITEM NUMBER AS ERROR NUMBER
8707      JSR      PC,$ATY4     ;;REPORT FATAL ERROR TO APT
8708
8709 21$: .BYTE    0
8710      .BYTE    0
8711 22$: BR      22$          ;;APT ERROR LOOP
8712 2$: TST      $SWR         ;;HALT ON ERROR
8713      BPL      3$          ;;SKIP IF CONTINUE
8714      HALT
8715 3$: BIT      $BIT09,$SWR  ;;HALT ON ERROR!
8716      BEQ      4$          ;;LOOP ON ERROR SWITCH SET?
8717      MOV      $LPERR,(SP)  ;;BR IF NO
8718 4$: TST      $ESCAPE      ;;FUDGE RETURN FOR LOOPING
8719      BEQ      5$          ;;CHECK FOR AN ESCAPE ADDRESS
8720      MOV      $ESCAPE,(SP) ;;BR IF NONE
8721 5$: CMP      $SENDAD,$#42  ;;FUDGE RETURN ADDRESS FOR ESCAPE
8722      BNE      6$          ;;ACT-11 AUTO-ACCEPT?
8723      HALT                ;;BRANCH IF NO
8724
8725 6$: RTI          ;;YES
8726      .SBTTL  TTY INPUT ROUTINE      ;;RETURN
8727
8728 ;*****
8729 .ENABL  LSB
8730
8731 .DSABL  LSB
8732
8733

```

```

8734      ;*****
8735      ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8736      ;CALL:
8737      ;      RDCHR
8738      ;      RETURN HERE
8739      ;
8740      ;
8741      ;
8742      ;
8743      ;
8744      ;
8745      ;
8746      ;
8747      ;
8748      ;
8749      ;
8750      ;
8751      ;
8752      ;
8753      ;
8754      ;
8755      ;
8756      ;
8757      ;
8758      ;
8759      ;
8760      ;
8761      ;
8762      ;
8763      ;
8764      ;
8765      ;
8766      ;
8767      ;
8768      ;
8769      ;
8770      ;
8771      ;
8772      ;
8773      ;
8774      ;
8775      ;
8776      ;
8777      ;
8778      ;
8779      ;
8780      ;
8781      ;
8782      ;
8783      ;
8784      ;
8785      ;
8786      ;
8787      ;
8788      ;
8789      ;

```

044276	011646	000004	000002	\$RDCHR: MOV	(SP), -(SP)	;; PUSH DOWN THE PC
044300	016666	134632		MOV	4(SP), 2(SP)	;; SAVE THE PS
044306	105777			1\$: TSTB	2\$TKS	;; WAIT FOR
044312	100375			BPL	1\$	;; A CHARACTER
044314	117766	134626	000004	MOVB	2\$TKB, 4(SP)	;; READ THE TTY
044322	042766	177600	000004	BIC	#1C<177>, 4(SP)	;; GET RID OF JUNK IF ANY
044330	026627	000004	000023	CMP	4(SP), #23	;; IS IT A CONTROL-S?
044336	001013			BNE	3\$	;; BRANCH IF NO
044340	105777	134600		2\$: TSTB	2\$TKS	;; WAIT FOR A CHARACTER
044344	100375			BPL	2\$	;; LOOP UNTIL ITS THERE
044346	117746	134574		MOVB	2\$TKB, -(SP)	;; GET CHARACTER
044352	042716	177600		BIC	#1C177, (SP)	;; MAKE IT 7-BIT ASCII
044356	022627	000021		CMP	(SP)+, #21	;; IS IT A CONTROL-Q?
044362	001366			BNE	2\$	;; IF NOT DISCARD IT
044364	000750			BR	1\$	;; YES, RESUME
044366	026627	000004	000140	3\$: CMP	4(SP), #140	;; IS IT UPPER CASE?
044374	002407			BLT	4\$	;; BRANCH IF YES
044376	026627	000004	000175	CMP	4(SP), #175	;; IS IT A SPECIAL CHAR?
044404	003003			BGT	4\$	;; BRANCH IF YES
044406	042766	000040	000004	BIC	#40, 4(SP)	;; MAKE IT UPPER CASE
044414	000002			4\$: RTI		;; GO BACK TO USER
044416	052536	005015	000	\$CNTLU: .ASCIZ	/U<15><12>	;; CONTROL "U"
044423	136	006507	000012	\$CNTLG: .ASCIZ	/G<15><12>	;; CONTROL "G"
044430	005015	053523	020122	\$MSWR: .ASCIZ	<15><12>/SWR = /	
044436	020075	000				
044441	040	047040	053505	\$MNEW: .ASCIZ	/ NEW = /	
044446	036440	000040				

```

8770      ;
8771      ;
8772      ;
8773      ;
8774      ;
8775      ;
8776      ;
8777      ;
8778      ;
8779      ;
8780      ;
8781      ;
8782      ;
8783      ;
8784      ;
8785      ;
8786      ;
8787      ;
8788      ;
8789      ;

```

044452	012737	044464	000024	\$PWRDN: MOV	#PWRUP, PWRVEC	;; SET VECTOR FOR POWER UP
044460	000000			HALT		;; HANG UP
044462	000776			BR	.-2	

```

8779      ;
8780      ;
8781      ;
8782      ;
8783      ;
8784      ;
8785      ;
8786      ;
8787      ;
8788      ;
8789      ;

```

044464	005037	044536		\$PWRUP: CLR	\$PWRCT	;; WAIT LOOP FOR TTY TO COME UP
044470	005237	044536		4\$: INC	\$PWRCT	
044474	001375			BNE	4\$	
044476	012737	044452	000024	MOV	#PWRDN, PWRVEC	;; SET VECTOR FOR POWER DOWN
044504	012737	000340	000026	MOV	#PR7, PWRVEC+2	;; RE-ESTABLISH POWER AND TRAP PRIORITIES
044512	012737	000340	000036	MOV	#PR7, TRAPVEC+2	
044520	012706	001100		MOV	#STACK, SP	;; RE-INITIALIZE THE STACK
044524	104401	044540		TYPE	, PWRMSG	;; TYPE "POWER FAILED"
044530	000005			RESET		;; CLEAR THE UNIBUS
044532	000177	134350		JMP	2\$LPADR	;; RESTART THE CURRENT TEST

CZR6080 RK6 DR CPT PROG MACY11 30(1046)
CZR608.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 174
POWER DOWN AND UP ROUTINES

SEQ 0173

8790 044536 000000
8791 044540 005015 047520 042527
8792 044546 020122 040506 046111
8793 044554 042105 005015 000
8794 044562
8795
8796
8797
8798
8799
8800

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

.EVEN

8801
8802 044562 105737 001103
8803 044566 001406
8804 044570 032777 001000 134342
8805 044576 001402
8806 044600 013716 001110
8807 044604 000002
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823 044606
8824 044606 032777 040000 134324
8825 044614 001052
8826
8827 044616 000416
8828
8829 044620 013746 000004
8830 044624 012737 044644 000004
8831 044632 005737 177060
8832 044636 012637 000004
8833 044642 000421
8834 044644 022626
8835 044646 012637 000004
8836 044652 000407
8837 044654
8838 044654 105737 001103
8839 044660 001412
8840 044662 032777 001000 134250
8841 044670 001404
8842 044672 013737 001110 001106
8843 044700 000420
8844 044702 105037 001103
8845 044706 105237 001102

* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
* CALLED BY "SCOPER"

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(\$STNM) 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
* SW09=1 LOOP ON ERROR
* CALL SCOPE ; ;SCOPE=IOT
*

\$SCOPE:
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
; ;THIS INSTRUCTION TO A "NOP" (NOP=240)
; ;SAVE THE CONTENTS OF THE ERROR VECTOR
MOV \$ERRVEC,-(SP) ; ;SET FOR TIMEOUT
MOV \$SS,\$ERRVEC ; ;TIME OUT ON XOR?
TST \$177060 ; ;RESTORE THE ERROR VECTOR
MOV (SP)+,\$ERRVEC ; ;GO TO THE NEXT TEST
BR \$SVLAD ; ;CLEAR THE STACK AFTER A TIME OUT
5\$: CMP (SP)+,(SP)+ ; ;RESTORE THE ERROR VECTOR
MOV (SP)+,\$ERRVEC ; ;LOOP ON THE PRESENT TEST
BR 7\$
6\$; *****END OF CODE FOR THE XOR TESTER*****
2\$: TSTB \$ERFLG ; ;HAS AN ERROR OCCURRED?
BEQ \$SVLAD ; ;BR IF NO
BIT \$BIT09,\$SWR ; ;LOOP ON ERROR?
BEQ 4\$; ;BR IF NO
7\$: MOV \$LPERR,\$LPADR ; ;SET LOOP ADDRESS TO LAST SCOPE
BR \$OVER
4\$: CLRB \$ERFLG ; ;ZERO THE ERROR FLAG
\$SVLAD: INCB \$STNM ; ;COUNT TEST NUMBERS

```

8846 044712 113737 001102 001334      MOV      $STNM,$TESTN      ;; SET TEST NUMBER IN APT MAILBOX
8847 044720 011637 001106      MOV      (SP), $LPADR      ;; SAVE SCOPE LOOP ADDRESS
8848 044724 011637 001110      MOV      (SP), $LPERR      ;; SAVE ERROR LOOP ADDRESS
8849 044730 005037 001316      CLR      $ESCAPE      ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
8850 044734 112737 000001 001115      MOV      #1,$ERMAX      ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
8851 044742 013777 001102 134172 $OVER: MOV      $STNM,$DISPLAY      ;; DISPLAY TEST NUMBER
8852 044750 013716 001106      MOV      $LPADR,(SP)      ;; FUDGE RETURN ADDRESS
8853 044754 000002      RTI      FIXES PS
8854      .SBTTL  APT COMMUNICATIONS ROUTINE
8855
8856      ;*****
8857 044756 112737 000001 045222 $ATY1: MOV      #1,$FFLG      ;; TO REPORT FATAL ERROR
8858 044764 112737 000001 045220 $ATY3: MOV      #1,$MFLG      ;; TO TYPE A MESSAGE
8859 044772 000403      BR      $ATYC
8860 044774 112737 000001 045222 $ATY4: MOV      #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
8861 045002      $ATYC:
8862 045002 010046      MOV      RO,-(SP)      ;; PUSH RO ON STACK
8863 045004 010146      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
8864 045006 105737 045220      TSTB     $MFLG      ;; SHOULD TYPE A MESSAGE?
8865 045012 001450      BEQ      5$      ;; IF NOT: BR
8866 045014 122737 000001 001350      CMPB     $APTENV,$ENV      ;; OPERATING UNDER APT?
8867 045022 001031      BNE      3$      ;; IF NOT: BR
8868 045024 132737 000100 001351      BITB     $APTPOOL,$ENVM      ;; SHOULD SPOOL MESSAGES?
8869 045032 001425      BEQ      3$      ;; IF NOT: BR
8870 045034 017600 000004      MOV      @4(SP),RO      ;; GET MESSAGE ADDR.
8871 045040 062766 000002 000004      ADD      #2,4(SP)      ;; BUMP RETURN ADDR.
8872 045046 005737 001330      1$: TST      $MSGTYPE      ;; SEE IF DONE W/ LAST XMISSION?
8873 045052 001375      BNE      1$      ;; IF NOT: WAIT
8874 045054 010037 001344      MOV      RO,$MSGAD      ;; PUT ADDR IN MAILBOX
8875 045060 105720      2$: TSTB     (RO)+      ;; FIND END OF MESSAGE
8876 045062 001376      BNE      2$
8877 045064 163700 001344      SUB      $MSGAD,RO      ;; SUB START OF MESSAGE
8878 045070 006200      ASR      RO      ;; GET MESSAGE LGTH IN WORDS
8879 045072 010037 001346      MOV      RO,$MSGLG      ;; PUT LENGTH IN MAILBOX
8880 045076 012737 000004 001330      MOV      #4,$MSGTYPE      ;; TELL APT TO TAKE MSG.
8881 045104 000413      BR      5$
8882 045106 017637 000004 045132 3$: MOV      @4(SP),4$      ;; PUT MSG ADDR IN JSR LINKAGE
8883 045114 062766 000002 000004      ADD      #2,4(SP)      ;; BUMP RETURN ADDRESS
8884 045122 013746 177776      MOV      177776,-(SP)      ;; PUSH 177776 ON STACK
8885 045126 004737 042354      JSR      PC,$TYPE      ;; CALL TYPE MACRO
8886 045132 000000      4$: .WORD      0
8887 045134      5$:
8888 045134 105737 045222      10$: TSTB     $FFLG      ;; SHOULD REPORT FATAL ERROR?
8889 045140 001416      BEQ      12$      ;; IF NOT: BR
8890 045142 005737 001350      TST      $ENV      ;; RUNNING UNDER APT?
8891 045146 001413      BEQ      12$      ;; IF NOT: BR
8892 045150 005737 001330      11$: TST      $MSGTYPE      ;; FINISHED LAST MESSAGE?
8893 045154 001375      BNE      11$      ;; IF NOT: WAIT
8894 045156 017637 000004 001332      MOV      @4(SP), $FATAL      ;; GET ERROR #
8895 045164 062766 000002 000004      ADD      #2,4(SP)      ;; BUMP RETURN ADDR.
8896 045172 005237 001330      INC      $MSGTYPE      ;; TELL APT TO TAKE ERROR
8897 045176 105037 045222      12$: CLRB     $FFLG      ;; CLEAR FATAL FLAG
8898 045202 105037 045221      CLRB     $LFLG      ;; CLEAR LOG FLAG
8899 045206 105037 045220      CLRB     $MFLG      ;; CLEAR MESSAGE FLAG
8900 045212 012601      MOV      (SP)+,R1      ;; POP STACK INTO R1
8901 045214 012600      MOV      (SP)+,RO      ;; POP STACK INTO RO

```

```

8902 045216 000207          RTS      PC          ;; RETURN
8903 045220          000          ;; MESSG. FLAG
8904 045221          000          ;; LOG FLAG
8905 045222          000          ;; FATAL FLAG
8906          045224          .EVEN
8907          000200      APTSIZE=200
8908          000001      APTENV=001
8909          000100      APTSP00L=100
8910          000040      APTCSUP=040
8911          .SBTTL      SAVE AND RESTORE RD-R5 ROUTINES
8912
8913          ;; *****
8914          ;; *SAVE RD-R5
8915          ;; *CALL:
8916          ;; *      SAVREG
8917          ;; *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
8918          ;; *
8919          ;; *TOP---(+16)
8920          ;; * +2---(+18)
8921          ;; * +4---R5
8922          ;; * +6---R4
8923          ;; * +8---R3
8924          ;; * +10---R2
8925          ;; * +12---R1
8926          ;; * +14---R0
8927
8928          $SAVREG:
8929          045224 010046      MOV      R0, -(SP)          ;; PUSH R0 ON STACK
8930          045226 010146      MOV      R1, -(SP)          ;; PUSH R1 ON STACK
8931          045230 010246      MOV      R2, -(SP)          ;; PUSH R2 ON STACK
8932          045232 010346      MOV      R3, -(SP)          ;; PUSH R3 ON STACK
8933          045234 010446      MOV      R4, -(SP)          ;; PUSH R4 ON STACK
8934          045236 010546      MOV      R5, -(SP)          ;; PUSH R5 ON STACK
8935          045240 016646 000022      MOV      22(SP), -(SP)      ;; SAVE PS OF MAIN FLOW
8936          045244 016646 000022      MOV      22(SP), -(SP)      ;; SAVE PC OF MAIN FLOW
8937          045250 016646 000022      MOV      22(SP), -(SP)      ;; SAVE PS OF CALL
8938          045254 016646 000022      MOV      22(SP), -(SP)      ;; SAVE PC OF CALL
8939          045260 000002      RTI
8940
8941          ;; *RESTORE RD-R5
8942          ;; *CALL:
8943          ;; *      RESREG
8944          $RESREG:
8945          045262 012666 000022      MOV      (SP)+, 22(SP)      ;; RESTORE PC OF CALL
8946          045266 012666 000022      MOV      (SP)+, 22(SP)      ;; RESTORE PS OF CALL
8947          045272 012666 000022      MOV      (SP)+, 22(SP)      ;; RESTORE PC OF MAIN FLOW
8948          045276 012666 000022      MOV      (SP)+, 22(SP)      ;; RESTORE PS OF MAIN FLOW
8949          045302 012605          MOV      (SP)+, R5          ;; POP STACK INTO R5
8950          045304 012604          MOV      (SP)+, R4          ;; POP STACK INTO R4
8951          045306 012603          MOV      (SP)+, R3          ;; POP STACK INTO R3
8952          045310 012602          MOV      (SP)+, R2          ;; POP STACK INTO R2
8953          045312 012601          MOV      (SP)+, R1          ;; POP STACK INTO R1
8954          045314 012600          MOV      (SP)+, R0          ;; POP STACK INTO R0
8955          045316 000002      RTI
8956          .SBTTL      TRAP DECODER
8957

```

```

8958      ;*****
8959      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8960      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8961      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8962      ;*GO TO THAT ROUTINE.
8963
8964      045320 010046      $TRAP:  MOV      RO, -(SP)      ;;SAVE RO
8965      045322 016600      MOV      2(SP),RO      ;;GET TRAP ADDRESS
8966      045326 005740      TST      -(RO)      ;;BACKUP BY 2
8967      045330 111000      MOV      (RO),RO      ;;GET RIGHT BYTE OF TRAP
8968      045332 006300      ASL      RO      ;;POSITION FOR INDEXING
8969      045334 016000      MOV      $TRAPD(RO),RO      ;;INDEX TO TABLE
8970      045340 000200      RTS      RO      ;;GO TO ROUTINE
8971
8972
8973      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8974
8975      045342 011646      $TRAP2: MOV      (SP), -(SP)      ;;MOVE THE PC DOWN
8976      045344 016666      MOV      4(SP), 2(SP)      ;;MOVE THE PSW DOWN
8977      045352 000002      RTI      ;;RESTORE THE PSW
8978
8979      .SBTTL  TRAP TABLE
8980
8981      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8982      ;*BY THE "TRAP" INSTRUCTION.
8983
8984      ;
8985      ROUTINE
8986      -----
8987      $TRAPD:  WORD      $TRAP2
8988      $TYPE    ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
8989      $TYPOC   ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8990      $TYPPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
8991      $TYPON   ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
8992      $TYPDS   ;;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
8993
8994      $RDCHR   ;;CALL=RDCHR     TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
8995      $SAVREG  ;;CALL=SAVREG    TRAP+7(104407)  SAVE RO-R5 ROUTINE
8996      $RESREG  ;;CALL=RESREG    TRAP+10(104410) RESTORE RO-R5 ROUTINE
8997      $SCOPE1  ;;CALL=SCOPER    TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
8998
8999      045400 020040 020040 042524 TSTMSG: .ASCIZ  / TEST /
9000      045406 052123 000040
9001      045412 025052 020040 000      AS2SP2: .ASCIZ  /** /
9002      045417 125 044516 052502 EM1: .ASCIZ  /UNIBUS PARITY ERROR/
9003      045424 020123 040520 044522
9004      045432 054524 042440 051122
9005      045440 051117 000
9006      045443 116 047117 042455 EM2: .ASCIZ  /NON-EXISTANT MEMORY/
9007      045450 044530 052123 047101
9008      045456 020124 042515 047515
9009      045464 054522 000
9010      045467 116 047117 042455 EM3: .ASCIZ  /NON-EXISTANT DRIVE/
9011      045474 044530 052123 047101
9012      045502 020124 051104 053111
9013      045510 000105

```

9014	045512	047125	052111	043040	EM4:	.ASCIZ	/UNIT FIELD ERROR/
9015	045520	042511	042114	042440			
9016	045526	051122	051117	000			
9017	045533	123	041125	054523	EM5:	.ASCIZ	/SUBSYS TIMEOUT/
9018	045540	020123	044524	042515			
9019	045546	052517	000124				
9020	045552	020104	047524	041440	EM6:	.ASCIZ	/D TO C PARITY ERROR/
9021	045560	050040	051101	052111			
9022	045566	020131	051105	047522			
9023	045574	000122					
9024	045576	051104	053111	020105	EM7:	.ASCIZ	/DRIVE DETECTED PARITY ERROR/
9025	045604	042504	042524	052103			
9026	045612	042105	050040	051101			
9027	045620	052111	020131	051105			
9028	045626	047522	000122				
9029	045632	041501	046040	053517	EM10:	.ASCIZ	/AC LCW/
9030	045640	000					
9031	045641	123	042520	042105	EM11:	.ASCIZ	/SPEED LOSS/
9032	045646	046040	051517	000123			
9033	045654	046111	042514	040507	EM12:	.ASCIZ	/ILLEGAL FUNCTION/
9034	045662	020114	052506	041516			
9035	045670	044524	047117	000			
9036	045675	120	047522	051107	EM13:	.ASCIZ	/PROGRAMMING ERROR/
9037	045702	046501	044515	043516			
9038	045710	042440	051122	051117			
9039	045716	000					
9040	045717	116	047117	042455	EM14:	.ASCIZ	/NON-EXISTANT FUNCTION/
9041	045724	044530	052123	047101			
9042	045732	020124	052506	041516			
9043	045740	044524	047117	000			
9044	045745	104	044522	042526	EM15:	.ASCIZ	/DRIVE TYPE ERROR/
9045	045752	052040	050131	020105			
9046	045760	051105	047522	000122			
9047	045766	047506	046522	052101	EM16:	.ASCIZ	/FORMAT ERROR/
9048	045774	042440	051122	051117			
9049	046002	000					
9050	046003	127	044522	042524	EM17:	.ASCIZ	/WRITE LOCK ERROR/
9051	046010	046040	041517	020113			
9052	046016	051105	047522	000122			
9053	046024	051104	053111	020105	EM20:	.ASCIZ	/DRIVE UNSAFE/
9054	046032	047125	040523	042506			
9055	046040	000					
9056	046041	123	042505	020113	EM21:	.ASCIZ	/SEEK INCOMPLETE/
9057	046046	047111	047503	050115			
9058	046054	042514	042524	000			
9059	046061	103	046131	047111	EM22:	.ASCIZ	/CYLINDER OVERFLOW/
9060	046066	042504	020122	053117			
9061	046074	051105	046106	053517			
9062	046102	000					
9063	046103	111	046114	043505	EM23:	.ASCIZ	/ILLEGAL CYLINDER ADDRESS/
9064	046110	046101	041440	046131			
9065	046116	047111	042504	020122			
9066	046124	042101	051104	051505			
9067	046132	000123					
9068	046134	051104	053111	020105	EM24:	.ASCIZ	/DRIVE OFF TRACK/
9069	046142	043117	020106	051124			

9070	046150	041501	000113		
9071	046154	051104	053111	020105	EM25: .ASCIZ /DRIVE TIMING ERROR/
9072	046162	044524	044515	043516	
9073	046170	042440	051122	051117	
9074	046176	000			
9075	046177	104	052101	020101	EM26: .ASCIZ /DATA LATE/
9076	046204	040514	042524	000	
9077	046211	103	047117	051124	EM27: .ASCIZ /CONTROLLER TIMEOUT/
9078	046216	046117	042514	020122	
9079	046224	044524	042515	052517	
9080	046232	000124			
9081	046234	050117	051105	052101	EM30: .ASCIZ /OPERATION INCOMPLETE/
9082	046242	047511	020116	047111	
9083	046250	047503	050115	042514	
9084	046256	042524	000		
9085	046261	110	040505	042504	EM31: .ASCIZ /HEADER VRC ERROR/
9086	046266	020122	051126	020103	
9087	046274	051105	047522	000122	
9088	046302	040504	040524	041440	EM32: .ASCIZ /DATA CHECK ERROR/
9089	046310	042510	045503	042440	
9090	046316	051122	051117	000	
9091	046323	127	044522	042524	EM33: .ASCIZ /WRITE CHECK ERROR/
9092	046330	041440	042510	045503	
9093	046336	042440	051122	051117	
9094	046344	000			
9095	046345	104	052101	020101	EM34: .ASCIZ /DATA MISCOMPARE/
9096	046352	044515	041523	046517	
9097	046360	040520	042522	000	
9098	046365	116	020117	051104	EM35: .ASCIZ /NO DRIVE RESPONSE-UFE & NXD/
9099	046372	053111	020105	042522	
9100	046400	050123	047117	042523	
9101	046406	052455	042506	023040	
9102	046414	047040	042130	000	
9103	046421	104	044522	042526	EM36: .ASCIZ /DRIVE ERROR WILL NOT CLEAR/
9104	046426	042440	051122	051117	
9105	046434	053440	046111	020114	
9106	046442	047516	020124	046103	
9107	046450	040505	000122		
9108	046454	051104	053111	020105	EM37: .ASCIZ /DRIVE STATUS CHANGE WILL NOT CLEAR/
9109	046462	052123	052101	051525	
9110	046470	041440	040510	043516	
9111	046476	020105	044527	046114	
9112	046504	047040	052117	041440	
9113	046512	042514	051101	000	
9114	046517	101	052124	047047	EM40: .ASCIZ /ATT'N BUT NO STATUS CHANGE OR FAULT/
9115	046524	041040	052125	047040	
9116	046532	020117	052123	052101	
9117	046540	051525	041440	040510	
9118	046546	043516	020105	051117	
9119	046554	043040	052501	052114	
9120	046562	000			
9121	046563	101	052124	047047	EM41: .ASCIZ /ATT'N BUT DRIVE NOT AVAILABLE/
9122	046570	041040	052125	042040	
9123	046576	044522	042526	047040	
9124	046604	052117	040440	040526	
9125	046612	046111	041101	042514	

K14

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 180
TRAP TABLE

SEQ 0179

9126	046620	000			
9127	046621	101	052124	047047	EM42: .ASCIZ /ATT'N WHEN NOT EXPECTED/
9128	046626	053440	042510	020116	
9129	046634	047516	020124	054105	
9130	046642	042520	052103	042105	
9131	046650	000			
9132	046651	105	051122	051117	EM43: .ASCIZ /ERROR GATHERING DRIVE STATUS/
9133	046656	043440	052101	042510	
9134	046664	044522	043516	042040	
9135	046672	044522	042526	051440	
9136	046700	040524	052524	000123	
9137	046706	052515	052114	050111	EM52: .ASCIZ /MULTIPLE DRIVE SELECT/
9138	046714	042514	042040	044522	
9139	046722	042526	051440	046105	
9140	046730	041505	000124		
9141	046734	042510	042101	051105	EM53: .ASCIZ /HEADER COMPARE ERROR/
9142	046742	041440	046517	040520	
9143	046750	042522	042440	051122	
9144	046756	051117	000		
9145	046761	123	041125	054523	EM56: .ASCIZ /SUBSYS TIMEOUT/
9146	046766	020123	044524	042515	
9147	046774	052517	000124		
9148	047000	051105	047522	020122	EM60: .ASCIZ /ERROR IN RECAL FOR RECOVERY/
9149	047006	047111	051040	041505	
9150	047014	046101	043040	051117	
9151	047022	051040	041505	053117	
9152	047030	051105	000131		
9153	047034	051120	043517	040522	EM61: .ASCIZ /PROGRAM ABORTING FATAL ERROR IN RETRY/
9154	047042	020115	041101	051117	
9155	047050	044524	043516	043040	
9156	047056	052101	046101	042440	
9157	047064	051122	051117	044440	
9158	047072	020116	042522	051124	
9159	047100	000131			
9160	047102	054503	044514	042116	EM62: .ASCIZ /CYLINDER MISCOMPARE/
9161	047110	051105	046440	051511	
9162	047116	047503	050115	051101	
9163	047124	000105			
9164	047126	046103	040505	020122	EM63: .ASCIZ /CLEAR CONTROLLER DID NOT CLEAR ERROR/
9165	047134	047503	052116	047522	
9166	047142	046114	051105	042040	
9167	047150	042111	047040	052117	
9168	047156	041440	042514	051101	
9169	047164	042440	051122	051117	
9170	047172	000			
9171	047173	116	020117	052101	EM64: .ASCIZ /NO ATT'N IN ATT'N SUMMARY REGISTER/
9172	047200	023524	020116	047111	
9173	047206	040440	052124	047047	
9174	047214	051440	046525	040515	
9175	047222	054522	051040	043505	
9176	047230	051511	042524	000122	EM65: .ASCIZ /UNSOLICITED ATTENTION/
9177	047236	047125	047523	044514	
9178	047244	044503	042524	020104	
9179	047252	052101	042524	052116	
9180	047260	047511	000116		
9181	047264	047125	054105	042520	EM66: .ASCIZ /UNEXPECTED DATA TYPE ERROR/

9182	047272	052103	042105	042040	
9183	047300	052101	020101	054524	
9184	047306	042520	042440	051122	
9185	047314	051117	000		
9186	047317	101	052124	047047	EM67: .ASCIZ /ATT'N DID NOT RESET WITH CLEAR/
9187	047324	042040	042111	047040	
9188	047332	052117	051040	051505	
9189	047340	052105	053440	052111	
9190	047346	020110	046103	040505	
9191	047354	000122			
9192	047356	052523	051502	051531	EM70: .ASCIZ /SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
9193	047364	041440	042514	051101	
9194	047372	042040	042111	047040	
9195	047400	052117	041440	042514	
9196	047406	051101	042040	044522	
9197	047414	042526	040440	052124	
9198	047422	047047	000		
9199	047425	104	052101	020101	EM71: .ASCIZ /DATA LATE WHEN UNLOADING HEADER/
9200	047432	040514	042524	053440	
9201	047440	042510	020116	047125	
9202	047446	047514	042101	047111	
9203	047454	020107	042510	042101	
9204	047462	051105	000		
9205	047465	103	047117	051124	EM72: .ASCIZ /CONTROLLER ERROR WHEN DRIVER SERVICING/
9206	047472	046117	042514	020122	
9207	047500	051105	047522	020122	
9208	047506	044127	047105	042040	
9209	047514	044522	042526	020122	
9210	047522	042523	053122	041511	
9211	047530	047111	000107		
9212	047534	051104	053111	020105	EM73: .ASCIZ /DRIVE DETECTED PARITY ERROR/
9213	047542	042504	042524	052103	
9214	047550	042105	050040	051101	
9215	047556	052111	020131	051105	
9216	047564	047522	000122		
9217	047570	047125	042504	044506	EM74: .ASCIZ /UNDEFINED ERROR/
9218	047576	042516	020104	051105	
9219	047604	047522	000122		
9220	047610	040515	045522	047111	EM75: .ASCIZ /MARKING THIS SECTOR BAD/
9221	047616	020107	044124	051511	
9222	047624	051440	041505	047524	
9223	047632	020122	040502	000104	
9224	047640	040502	020104	040504	EM76: .ASCIZ /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
9225	047646	040524	053040	051105	
9226	047654	043111	047047	053440	
9227	047662	052111	020110	042522	
9228	047670	042101	020056	041505	
9229	047676	020103	043117	046040	
9230	047704	051501	020124	042522	
9231	047712	051124	020131	051511	
9232	047720	000072			
9233	047722	042522	051124	020131	EM77: .ASCIZ /RETRY SUCCESSFUL/
9234	047730	052523	041503	051505	
9235	047736	043123	046125	000	
9236	047743	122	052105	054522	EM100: .ASCIZ /RETRY UNSUCCESSFUL/
9237	047750	052440	051516	041525	

9238	047756	042503	051523	052506	
9239	047764	000114			
9240	047766	040503	047116	052117	EM101: .ASCIZ /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
9241	047774	043040	047111	020104	
9242	050002	020101	040526	044514	
9243	050010	020104	042510	042101	
9244	050016	051105	044440	020116	
9245	050024	051124	041501	020113	
9246	050032	052512	052123	051040	
9247	050040	040505	000104		
9248	050044	040502	020104	042523	EM102: .ASCIZ /BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
9249	050052	052103	051117	042440	
9250	050060	051122	051117	047440	
9251	050066	020116	042523	052103	
9252	050074	051117	047040	052117	
9253	050102	046040	051511	042524	
9254	050110	020104	040502	000104	
9255	050116	044524	042515	026504	EM103: .ASCIZ /TIMED-OUT ON READ HDR/
9256	050124	052517	020124	047117	
9257	050132	051040	040505	020104	
9258	050140	042110	000122		
9259	050144	044524	042515	026504	EM104: .ASCIZ /TIMED-OUT ON SEEK/
9260	050152	052517	020124	047117	
9261	050160	051440	042505	000113	
9262	050166	051104	053111	020105	EM105: .ASCIZ /DRIVE SIEZED BY OTHER PORT/
9263	050174	044523	055105	042105	
9264	050202	041040	020131	052117	
9265	050210	042510	020122	047520	
9266	050216	052122	000		
9267	050221	104	052101	020101	EM106: .ASCIZ /DATA MISCMPR WHILE BAI SET/
9268	050226	044515	041523	050115	
9269	050234	020122	044127	046111	
9270	050242	020105	040502	020111	
9271	050250	042523	000124		
9272	050254	047516	047040	046505	EM107: .ASCIZ /NO NEM ERROR WHEN REF'ING LOC 760000/
9273	050262	042440	051122	051117	
9274	050270	053440	042510	020116	
9275	050276	042522	023506	047111	
9276	050304	020107	047514	020103	
9277	050312	033067	030060	030060	
9278	050320	000			
9279	050321	111	052116	050122	EM110: .ASCIZ /INTRPT WHEN CNTRLR NOT RDY/
9280	050326	020124	044127	047105	
9281	050334	041440	052116	046122	
9282	050342	020122	047516	020124	
9283	050350	042122	000131		
9284	050354	047516	040440	052124	EM111: .ASCIZ /NO ATT'N ON SEEK/
9285	050362	047047	047440	020116	
9286	050370	042523	045505	000	
9287	050375	104	044522	042526	EM112: .ASCIZ /DRIVE'S CYLINDER INCORRECT/
9288	050402	051447	041440	046131	
9289	050410	047111	042504	020122	
9290	050416	047111	047503	051122	
9291	050424	041505	000124		
9292	050430	041101	051117	026524	EM113: .ASCIZ /ABORT- CAN'T READ BSF/
9293	050436	041440	047101	052047	

Address	Offset	Value	Label	Comment
9294	050444	040505	020104	
9295	050452	051502	000106	
9296	050456	052113	030461	EM114: .ASCIZ /KT11 FAILURE/
9297	050464	044501	052514	042522
9298	050472	000		
9299	050473	115	046505	050040
9300	050500	051101	052111	020131
9301	050506	051105	000122	
9302	050512	052503	051122	047105
9303	050520	020124	047503	046515
9304	050526	047101	020104	000072
9305	050534	051120	053105	047511
9306	050542	051525	041440	046517
9307	050550	040515	042116	035040
9308	050556	000		
9309	050557	122	033113	030461
9310	050564	051040	043505	051511
9311	050572	042524	051522	035040
9312	050600	000		
9313	050601	122	046505	044501
9314	050606	044516	043516	051040
9315	050614	043505	051511	042524
9316	050622	051522	047040	052117
9317	050630	053040	046101	042111
9318	050636	000		
9319	050637	124	042510	043040
9320	050644	046117	047514	044527
9321	050652	043516	051040	043505
9322	050660	051511	042524	020122
9323	050666	040504	040524	046440
9324	050674	054501	041040	020105
9325	050702	047111	047503	051122
9326	050710	041505	020124	000072
9327	050716	051105	020122	041520
9328	050724	020040	051104	053111
9329	050732	020105	020040	046503
9330	050740	042116	020040	020040
9331	050746	054503	047114	051104
9332	050754	020040	051124	041501
9333	050762	020113	020040	042523
9334	050770	052103	051117	020040
9335	050776	042127	041440	052116
9336	051004	000		
9337	051005	110	020111	040502
9338	051012	020040	046040	020117
9339	051020	040502	000	
9340	051023	120	042522	027126
9341	051030	052440	044516	052502
9342	051036	020123	040515	020120
9343	051044	000072		
9344	051046	044510	051040	043505
9345	051054	020060	047514	051040
9346	051062	043505	000060	
9347	051066	052503	027122	052440
9348	051074	044516	052502	020123
9349	051102	040515	020120	000072

9406	051572	051117	031504	000	
9407	051577	102	042101	044040	DH607: .ASCIZ /BAD HEADER IS :/
9408	051604	040505	042504	020122	
9409	051612	051511	035040	000	
9410	051617	127	020104	020043	DH701: .ASCIZ /WD # GOOD BAD HI PHY LO PHY VRT AD KIPAR6/
9411	051624	020040	043440	047517	
9412	051632	020104	020040	041040	
9413	051640	042101	020040	020040	
9414	051646	044040	020111	044120	
9415	051654	020131	046040	020117	
9416	051662	044120	020131	053040	
9417	051670	052122	040440	020104	
9418	051676	045140	050111	051101	
9419	051704	000066			
9420	051706	047507	042117	020040	DH702: .ASCIZ /GOOD BAD/
9421	051714	020040	040502	000104	
9422	051722	040506	046111	047111	DH703: .ASCIZ /FAILING DATA WORD :/
9423	051730	020107	040504	040524	
9424	051736	053440	051117	020104	
9425	051744	000072			
9426	051746	051123	004460	051123	DH704: .ASCIZ /SR0 SR1 SR2 SR3/
9427	051754	004461	051123	004462	
9428	051762	051123	000063		
9429	051766	052113	030461	051040	DH705: .ASCIZ /KT11 REGS :/
9430	051774	043505	020123	000072	
9431	052002	042515	020105	042522	DH706: .ASCIZ /MEM REGS :/
9432	052010	051507	000040	000	
9433	052015	120	004503	047514	DH707: .ASCIZ /PC LOERAD HIERAD MEMSYS/
9434	052022	051105	042101	044011	
9435	052030	042511	040522	004504	
9436	052036	042515	051515	051531	
9437	052044	000			
9438	052045	120	004503	051503	DH710: .ASCIZ /PC CSR AD CSR/
9439	052052	020122	042101	041411	
9440	052060	051123	000		
9441	052063	103	046131	047111	DH711: .ASCIZ /CYLINDERS :/
9442	052070	042504	051522	035040	
9443	052076	000			
9444	052077	122	042113	020123	DH204: .ASCIZ /RKDS RKER RKMR2 RKMR3/
9445	052104	020040	051040	042513	
9446	052112	020122	020040	051040	
9447	052120	046513	031122	020040	
9448	052126	051040	046513	031522	
9449	052134	000			
9450	052135	105	051122	051117	DH502: .ASCIZ /ERROR TRYING TO GET FAILURE INFO/
9451	052142	052040	054522	047111	
9452	052150	020107	047524	043440	
9453	052156	052105	043040	044501	
9454	052164	052514	042522	044440	
9455	052172	043116	000117		
9456	052176	042523	047503	042116	DH503: .ASCIZ /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
9457	052204	052040	046511	047505	
9458	052212	052125	047440	041503	
9459	052220	051125	042522	020104	
9460	052226	042507	052124	047111	
9461	052234	020107	051104	053111	

9462	052242	020105	052123	052101		
9463	052250	051525	000			
9464	052253	116	046525	042502	DH800:	.ASCIZ /NUMBER OF RETRIES:/
9465	052260	020122	043117	051040		
9466	052266	052105	044522	051505		
9467	052274	000072				
9468						
9469	052276	001116	005430	005426	DT100:	.EVEN \$ERRPC,DRIVE,ERRCOM,\$REG0,\$REG1,\$REG2,\$REG3
9470	052304	001162	001164	001166		
9471	052312	001170				
9472	052314	001256	001260		DT102:	.WORD \$REG36,\$REG37
9473	052320	001174	001176	001200	DT201:	.WORD \$REG5,\$REG6,\$REG7,\$REG10,\$REG11,\$REG12,\$REG13
9474	052326	001202	001204	001206		
9475	052334	001210				
9476	052336	001212	001214		DT202:	.WORD \$REG14,\$REG15
9477	052342	001216	001220	001222	DT203:	.WORD \$REG16,\$REG17,\$REG20,\$REG21,\$REG22,\$REG23,\$REG24,\$REG25
9478	052350	001224	001226	001230		
9479	052356	001232	001234			
9480	052362	001174	001176	001200	DT601:	.WORD \$REG5,\$REG6,\$REG7
9481	052370	001202	001204	001206	DT602:	.WORD \$REG10,\$REG11,\$REG12,\$REG13,\$REG14,\$REG15,\$REG16
9482	052376	001210	001212	001214		
9483	052404	001216				
9484						
9485	052406	000006			DF01:	.WORD 6 ;NUMBER OF HEADER LINES
9486	052410	000				.BYTE 0 ;NUMBER OF COL FOR FIRST HDR
9487	052411	000				.BYTE 0 ;ALL CHARACTERS OCTAL
9488	052412	050716				.WORD DH101 ;SECOND HDR LINE
9489	052414	007	000			.BYTE 7,0 ;NUM OF COL-ALL OCTAL
9490	052416	051005				.WORD DH102
9491	052420	002	000			.BYTE 2,0
9492	052422	050557				.WORD DH200
9493	052424	000	000			.BYTE 0,0
9494	052426	051110				.WORD DH201
9495	052430	007	000			.BYTE 7,0
9496	052432	050601				.WORD DH500
9497	052434	000	000			.BYTE 0,0
9498						
9499	052436	000007			DF02:	.WORD 7
9500	052440	000	000			.BYTE 0,0
9501	052442	050716				.WORD DH101
9502	052444	007	000			.BYTE 7,0
9503	052446	051005				.WORD DH102
9504	052450	002	000			.BYTE 2,0
9505	052452	050557				.WORD DH200
9506	052454	000	000			.BYTE 0,0
9507	052456	051110				.WORD DH201
9508	052460	007	000			.BYTE 7,0
9509	052462	051177				.WORD DH202
9510	052464	002	000			.BYTE 2,0
9511	052466	051214				.WORD DH203
9512	052470	010	000			.BYTE 10,0
9513						
9514	052472	000010			DF03:	.WORD 10
9515	052474	000	000			.BYTE 0,0
9516	052476	050716				.WORD DH101
9517	052500	007	000			.BYTE 7,0

9518	052502	051005		.WORD	DH102
9519	052504	002	000	.BYTE	2,0
9520	052506	050557		.WORD	DH200
9521	052510	000	000	.BYTE	0,0
9522	052512	051110		.WORD	DH201
9523	052514	007	000	.BYTE	7,0
9524	052516	050637		.WORD	DH5C1
9525	052520	000	000	.BYTE	0,0
9526	052522	051177		.WORD	DH202
9527	052524	002	000	.BYTE	2,0
9528	052526	051214		.WORD	DH203
9529	052530	010	000	.BYTE	10,0
9530					
9531	052532	000003		DF04: .WORD	3
9532	052534	000	000	.BYTE	0,0
9533	052536	050716		.WORD	DH101
9534	052540	007	000	.BYTE	7,0
9535	052542	051005		.WORD	DH102
9536	052544	002	000	.BYTE	2,0
9537					
9538	052546	000007		DF05: .WORD	7
9539	052550	000	000	.BYTE	0,0
9540	052552	050716		.WORD	DH101
9541	052554	007	000	.BYTE	7,0
9542	052556	051005		.WORD	DH102
9543	052560	002	000	.BYTE	2,0
9544	052562	051310		.WORD	DH601
9545	052564	000	000	.BYTE	0,0
9546	052566	051551		.WORD	DH606
9547	052570	003	000	.BYTE	3,0
9548	052572	051332		.WORD	DH602
9549	052574	000	000	.BYTE	0,0
9550	052576	051551		.WORD	DH606
9551	052600	003	000	.BYTE	3,0
9552					
9553	052602	000007		DF07: .WORD	7
9554	052604	000	000	.BYTE	0,0
9555	052606	050716		.WORD	DH101
9556	052610	007	000	.BYTE	7,0
9557	052612	051005		.WORD	DH102
9558	052614	002	000	.BYTE	2,0
9559	052616	051377		.WORD	DH604
9560	052620	000	000	.BYTE	0,0
9561	052622	051432		.WORD	DH6041
9562	052624	003	000	.BYTE	3,0
9563	052626	051477		.WORD	DH605
9564	052630	000	000	.BYTE	0,0
9565	052632	051514		.WORD	DH6051
9566	052634	003	000	.BYTE	3,0
9567					
9568	052636	000005		DF10: .WORD	5
9569	052640	000	000	.BYTE	0,0
9570	052642	050716		.WORD	DH101
9571	052644	007	000	.BYTE	7,0
9572	052646	051005		.WORD	DH102
9573	052650	002	000	.BYTE	2,0

;"THE FOLLOWING REG DATA MB INCORRECT"

,OPI, HVRC

9574	052652	051377		.WORD	DH604
9575	052654	000	000	.BYTE	0,0
9576	052656	051432		.WORD	DH6041
9577	052660	003	000	.BYTE	3,0
9578					
9579	052662	000004		DF11: .WORD	4
9580	052664	000	000	.BYTE	0,0
9581	052666	051377		.WORD	DH604
9582	052670	000	000	.BYTE	0,0
9583	052672	051432		.WORD	DH6041
9584	052674	003	000	.BYTE	3,0
9585	052676	051617		.WORD	DH701
9586	052700	000	000	.BYTE	0,0
9587					
9588	052702	000006		DF12: .WORD	6
9589	052704	000	000	.BYTE	0,0
9590	052706	050716		.WORD	DH101
9591	052710	007	000	.BYTE	7,0
9592	052712	051005		.WORD	DH102
9593	052714	002	000	.BYTE	2,0
9594	052716	050557		.WORD	DH200
9595	052720	000	000	.BYTE	0,0
9596	052722	051110		.WORD	DH201
9597	052724	007	000	.BYTE	7,0
9598	052726	052077		.WORD	DH204
9599	052730	004	000	.BYTE	4,0
9600					
9601	052732	000006		DF13: .WORD	6
9602	052734	000	000	.BYTE	0,0
9603	052736	050716		.WORD	DH101
9604	052740	007	000	.BYTE	7,0
9605	052742	051005		.WORD	DH102
9606	052744	002	000	.BYTE	2,0
9607	052746	051310		.WORD	DH601
9608	052750	000	000	.BYTE	0,0
9609	052752	051551		.WORD	DH606
9610	052754	003	000	.BYTE	3,0
9611	052756	052135		.WORD	DH502
9612	052760	000	000	.BYTE	0,0
9613					
9614	052762	000006		DF14: .WORD	6
9615	052764	000	000	.BYTE	0,0
9616	052766	050716		.WORD	DH101
9617	052770	007	000	.BYTE	7,0
9618	052772	051005		.WORD	DH102
9619	052774	002	000	.BYTE	2,0
9620	052776	051310		.WORD	DH601
9621	053000	070	000	.BYTE	0,0
9622	053002	051551		.WORD	DH606
9623	053004	003	000	.BYTE	3,0
9624	053006	052135		.WORD	DH502
9625	053010	000	000	.BYTE	0,0
9626					
9627	053012	000011		DF15: .WORD	11
9628	053014	000	000	.BYTE	0,0
9629	053016	050716		.WORD	DH101

```

;FORMAT FOR 2ND LEVEL ERROR
;IN HEADER COMPARE ERROR
;AND 2ND LEVEL HEADER
;VRC ERROR

```

```

;FORMAT FOR 2ND LEVEL ERROR
;IN OPERATION INCOMPLETE ERROR

```

9630	053020	007	000	.BYTE	7,0
9631	053022	051005		.WORD	0H102
9632	053024	002	000	.BYTE	2,0
9633	053026	050557		.WORD	0H200
9634	053030	000	000	.BYTE	0,0
9635	053032	051110		.WORD	0H201
9636	053034	007	000	.BYTE	7,0
9637	053036	051177		.WORD	0H202
9638	053040	002	000	.BYTE	2,0
9639	053042	052176		.WORD	0H503
9640	053044	000	000	.BYTE	0,0
9641	053046	050637		.WORD	0H501
9642	053050	000	000	.BYTE	0,0
9643	053052	051214		.WORD	0H203
9644	053054	010	000	.BYTE	10,0
9645					
9646	053056	000011		DF16: .WORD	11
9647	053060	000	000	.BYTE	0,0
9648	053062	050716		.WORD	0H101
9649	053064	007	000	.BYTE	7,0
9650	053066	051005		.WORD	0H102
9651	053070	002	000	.BYTE	2,0
9652	053072	050557		.WORD	0H200
9653	053074	000	000	.BYTE	0,0
9654	053076	051110		.WORD	0H201
9655	053100	007	000	.BYTE	7,0
9656	053102	051177		.WORD	0H202
9657	053104	002	000	.BYTE	2,0
9658	053106	052135		.WORD	0H502
9659	053110	000	000	.BYTE	0,0
9660	053112	050637		.WORD	0H501
9661	053114	000	000	.BYTE	0,0
9662	053116	051214		.WORD	0H203
9663	053120	010	000	.BYTE	10,0
9664					
9665	053122	000005		DF17: .WORD	5
9666	053124	000	000	.BYTE	0,0
9667	053126	050716		.WORD	0H101
9668	053130	007	000	.BYTE	7,0
9669	053132	051005		.WORD	0H102
9670	053134	002	000	.BYTE	2,0
9671	053136	052063		.WORD	0H711
9672	053140	000	000	.BYTE	0,0
9673	053142	051706		.WORD	0H702
9674	053144	002	000	.BYTE	2,0
9675					
9676	053146	000002		DF20: .WORD	2
9677	053150	000	000	.BYTE	0,0
9678	053152	051046		.WORD	0H104
9679	053154	002	000	.BYTE	2,0
9680					
9681	053156	000002		DF21: .WORD	2
9682	053160	000	000	.BYTE	0,0
9683	053162	051514		.WORD	0H6051
9684	053164	003	000	.BYTE	3,0
9685					

9686	053166	000006		DF22:	.WORD	6
9687	053170	000	000		.BYTE	0,0
9688	053172	050512			.WORD	04100
9689	053174	000	000		.BYTE	0,0
9690	053176	050716			.WORD	04101
9691	053200	007	000		.BYTE	7,0
9692	053202	051005			.WORD	04102
9693	053204	002	000		.BYTE	2,0
9694	053206	051066			.WORD	04106
9695	053210	000	000		.BYTE	0,0
9696	053212	051046			.WORD	04104
9697	053214	002	000		.BYTE	2,0
9698						
9699	053216	000002		DF23:	.WORD	2
9700	053220	000	000		.BYTE	0,0
9701	053222	052253			.WORD	04800
9702	053224	001	000		.BYTE	1,0
9703						
9704	053226	000001		DF24:	.WORD	1
9705	053230	002	000		.BYTE	2,0
9706						
9707	053232	000000		DF25:	.WORD	0
9708	053234	007	000		.BYTE	7,0
9709						
9710	053236	000002		DF26:	.WORD	2
9711	053240	002	000		.BYTE	2,0
9712	053242	051214			.WORD	04203
9713	053244	010	000		.BYTE	10,0
9714						
9715	053246	000005		DF27:	.WORD	5
9716	053250	000	000		.BYTE	0,0
9717	053252	050716			.WORD	04101
9718	053254	007	000		.BYTE	7,0
9719	053256	051005			.WORD	04102
9720	053260	002	000		.BYTE	2,0
9721	053262	051722			.WORD	04703
9722	053264	000	000		.BYTE	0,0
9723	053266	051706			.WORD	04702
9724	053270	002	000		.BYTE	2,0
9725						
9726	053272	000005		DF30:	.WORD	5
9727	053274	000	000		.BYTE	0,0
9728	053276	050716			.WORD	04101
9729	053300	007	000		.BYTE	7,0
9730	053302	051005			.WORD	04102
9731	053304	002	000		.BYTE	2,0
9732	053306	051766			.WORD	04705
9733	053310	000	000		.BYTE	0,0
9734	053312	051746			.WORD	04704
9735	053314	004	000		.BYTE	4,0
9736						
9737	053316	000005		DF31:	.WORD	5
9738	053320	000	000		.BYTE	0,0
9739	053322	050716			.WORD	04101
9740	053324	007	000		.BYTE	7,0
9741	053326	051005			.WORD	04102

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 191
 CZR6Q8.P11 02-DEC-77 10:46 TRAP TABLE

SEQ 0190

9742	053330	002	000	.BYTE	2,0	
9743	053332	052002		.WORD	04706	
9744	053334	000	000	.BYTE	0,0	
9745	053336	052045		.WORD	04710	
9746	053340	003	000	.BYTE	3,0	
9747						
9748	053342	000400		RWBUF: .BLKW	256.	;READ/WRITE DATA BUFFER
9749						
9750	054342	005015	020040	020040	NOTMSG: .ASCII	<15><12>/ *** NOTE ***<15><12><12>
9751	054350	020040	020040	020040		
9752	054356	025052	020052	047516		
9753	054364	042524	025040	025052		
9754	054372	005015	012			
9755	054375	117	046116	020131	.ASCII	/ONLY ALL RK06 OR ALL RK07'S MUST BE UNDER TEST/<15><12><12>
9756	054402	046101	020114	045522		
9757	054410	033060	047440	020122		
9758	054416	046101	020114	045522		
9759	054424	033460	051447	046440		
9760	054432	051525	020124	042502		
9761	054440	052440	042116	051105		
9762	054446	052040	051505	006524		
9763	054454	005012			.ASCII	/ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
9764	054456	046101	020114	051104		
9765	054464	053111	051505	052040		
9766	054472	020117	042502	052040		
9767	054500	051505	042524	020104		
9768	054506	052515	052123	044040		
9769	054514	053101	020105	006472		
9770	054522	005012			.ASCII	/1. DESIRED PORT SELECTED/<15><12>
9771	054524	027061	042040	051505		
9772	054532	051111	042105	050040		
9773	054540	051117	020124	042523		
9774	054546	042514	052103	042105		
9775	054554	005015			.ASCII	/2. WRITE LOCK DISABLED/<15><12>
9776	054556	027062	053440	044522		
9777	054564	042524	046040	041517		
9778	054572	020113	044504	040523		
9779	054600	046102	042105	005015	.ASCII	/DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
9780	054606	051104	053111	051505		
9781	054614	047040	052117	052040		
9782	054622	020117	042502	052040		
9783	054630	051505	042524	020104		
9784	054636	052515	052123	044040		
9785	054644	053101	020105	047502		
9786	054652	044124	005015		.ASCII	/PORTS DE-SELECTED./<15><12>
9787	054656	047520	052122	020123		
9788	054664	042504	051455	046105		
9789	054672	041505	042524	027104		
9790	054700	005015			.ASCII	/THE CARTRIDGE MUST BE FORMATTED ON A KNOWN GOOD/<15><12>
9791	054702	044124	020105	040503		
9792	054710	052122	044522	043504		
9793	054716	020105	052515	052123		
9794	054724	041040	020105	047506		
9795	054732	046522	052101	042524		
9796	054740	020104	047117	040440		
9797	054746	045440	047516	047127		

CZREJ80 RK6 DR CPT PRG MACY11 30(1046)
CZREJ80.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 192
TRAP TABLE

SEQ 0191

9798	054754	043440	047517	006504
9799	054762	012		
9800	054763	127	046105	026514
9801	054770	046101	043511	042516
9802	054776	020104	051104	053111
9803	055004	026105	050040	044522
9804	055012	051117	052040	020117
9805	055020	042524	052123	047111
9806	055026	027107	005015	
9807	055032	044124	020105	047503
9808	055040	050115	042514	042524
9809	055046	051040	047125	052040
9810	055054	046511	020105	051511
9811	055062	034040	030455	020060
9812	055070	044515	027116	050040
9813	055076	051105	042040	044522
9814	055104	042526	006456	012
9815	055111	124	020117	041101
9816	055116	051117	020124	042524
9817	055124	052123	047111	026107
9818	055132	052040	050131	020105
9819	055140	047503	052116	047522
9820	055146	026514	020103	057050
9821	055154	024503	006456	005012
9822	055162	000		
9823				
9824				
9825				
9826				
9827				
9828				
9829				

000001

.END

.ASCII /WELL-ALIGNED DRIVE, PRIOR TO TESTING./<15><12>

.ASCII /THE COMPLETE RUN TIME IS 8-10 MIN. PER DRIVE./<15><12>

.ASCII /TO ABORT TESTING, TYPE CONTROL-C (↑C)./<15><12><12>

SEQ 0193

APTSIZ=	000200	3559	8907*											
APTSP0=	000100	8260	8868	8909*										
ASKSYS	013446	3697	3708	3713	3719*	3723	3724	3726	3731	3742	5062	5134	5811	
ASKTYP	010025	3281*	5482											
ASWREG=	000000	1593	1606											
AS2SP2	045412	6397	9001*											
ATESTN=	000000	1593	1597											
AUNIT =	000000	1593	1600											
AUSWR =	000000	1593	1607											
AVECT1=	120210	1496*	1593	1632	3599									
AVECT2=	000000	1593	1633											
AVSCOR	020024	4124	4193	4257	4577*									
A.ABNL	003056	2843*	3842*	3845*	4869*	4943*	4948*	5802*	5984*	6001*	6010*	6194*	6209*	6611*
		6876*	6879*	6893*	6896*	6899*	7826							
A.CONT	003060	2844*	7836											
A.NORM	003054	2842*	4870*	6612*	6880*	7830								
BADDRV	011303	3409*	4381*	4382*	4386	4946*	4951*	4952*	4953	4965*	4966*	4967	4973*	4974*
		4975	4996*	4997*	4998									
BADSEC=	001000	3246*	5726	5911	6274	6282	6818	6930						
BAD632	012036	3479*	4993											
BA1 =	000020	2521*	7995											
BA16 =	000400	2497*												
BA17 =	001000	2498*												
BDSC ID	027004	5984	6001	6015*										
BDSECT	012007	3473*	6033											
BDSACK	030042	5724	5909	6264*	6817									
BGNRTY	033560	6535	6890	6901	6923*									
BITO =	000001	1428*	2494	2511	2537	2559	2707	2851	2944	3794	5646	5695	5735	5922
		6494	8110											
BIT00 =	000001	1418*	1428											
BIT01 =	000002	1417*	1427											
BIT02 =	000004	1416*	1426											
BIT03 =	000010	1415*	1425											
BIT04 =	000020	1414*	1424											
BIT05 =	000040	1413*	1423											
BIT06 =	000100	1412*	1422											
BIT07 =	000200	1411*	1421											
BIT08 =	000400	1410*	1420											
BIT09 =	001000	1409*	1419	5537	8714	8840								
BIT1 =	000002	1427*	2512	2539	2708	2852	2945	3238	6498					
BIT10 =	002000	1408*	2499	2528	2550	2585	2599	2612	2627	2641	2720	2861	3247	6527
		8692												
BIT11 =	004000	1407*	2500	2529	2551	2570	2586	2600	2613	2628	2642	2721	3248	
BIT12 =	010000	1406*	2502	2530	2552	2587	2601	2614	2615	2629	2643	2722	3249	6409
		6437												
BIT13 =	020000	1405*	2503	2531	2553	2571	2588	2602	2616	2630	2644	2723	8699	
BIT14 =	040000	1404*	2504	2532	2554	2572	2589	2603	2617	2631	2645	2724	2862	8824
BIT15 =	100000	1403*	2505	2506	2533	2555	2573	2590	2650	2725	2863	3250	6530	
BIT2 =	000004	1426*	2513	2540	2541	2561	2709	2853	2946	3239	6502			
BIT3 =	000010	1425*	2519	2520	2542	2562	2710	2854	2947	3240	6509			
BIT4 =	000020	1424*	2514	2521	2543	2563	2564	2579	2621	2635	2711	2855	2948	3241
		6513												
BIT5 =	000040	1423*	2522	2523	2544	2565	2580	2594	2607	2622	2636	2712	2856	3242
		4741	6517	7907										
BIT6 =	000100	1422*	2495	2524	2545	2566	2581	2595	2608	2623	2637	2713	2858	3243
		4746	4779	5762	5949									

CZR6QBD RK6 DR CPT PROG MACY11 3D(1046) 02-DEC-77 11:48 PAGE 196
CZR6QB.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

N15

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 197
CZR6Q8.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0195

C. INIT	040222	6315	7869#											
C. RTN	042074	7973	8001	8029	8036	8039	8046	8050	8067	8089	8094	8104	8113	8126
		8130	8135#											
C. SPEC	041114	7909	8005#											
DCEFLG	003144	2931#	4725*	6192*										
DCK =	100000	2555#	4721	6015	6827	7260	7426	7662						
DCKERR=	000020	3241#	4724	5275	6132	6157	6829	6923						
DCKHDL	020556	4716#	6194	6876	6896									
DCL0 =	000020	2564#												
DDISP =	177570	1350#	1527	3547										
DDT =	000400	2569#												
DERCNT	003130	2919#	6757	6787	6836	6847								
DESL =	000010	2519#												
DFLTDR	013774	3607	3789#											
DFSTRY	012176	1454	3510#	5057	5129									
DF01	052406	1658	1664	1670	1676	1718	1826	2096	2132	9485#				
DF02	052436	1682	1700	1706	1712	1724	1730	1736	1742	1748	1754	1760	1766	1772
		1778	1784	1790	1844	1850	1856	1964	1970	1976	1982	1988	1994	2000
		2006	2012	2018	2060	2078	2102	9499#						
DF03	052472	1688	1694	2066	2072	9514#								
DF04	052532	1820	9531#											
DF05	052546	1796	1802	9538#										
DF07	052602	1808	9553#											
DF10	052636	1814	9568#											
DF11	052662	2114	9579#											
DF12	052702	1862	1868	1874	1880	1886	1892	1898	1904	9588#				
DF13	052732	1910	1922	9601#										
DF14	052762	1916	9614#											
DF15	053012	1928	9627#											
DF16	053056	1934	9646#											
DF17	053122	1952	9665#											
DF20	053146	2120	9676#											
DF21	053156	2130	9681#											
DF22	053166	2126	9686#											
DF23	053216	2036	2042	2048	9699#									
DF24	053226	2054	9704#											
DF25	053232	1958	5684#	5714*	5896*	5914*	9707#							
DF26	053236	2108	9710#											
DF27	053246	2084	9715#											
DF30	053272	2138	9726#											
DF31	053316	2144	9737#											
DH100	050512	1656	1662	1668	1674	1680	1686	1692	1698	1704	1710	1716	1722	1728
		1734	1740	1746	1752	1758	1764	1770	1776	1782	1788	1794	1800	1806
		1812	1818	1824	1842	1848	1854	1860	1866	1872	1878	1884	1890	1896
		1902	1908	1914	1920	1926	1932	1950	1962	1968	1974	1980	1986	1992
		1998	2004	2010	2016	2058	2064	2070	2076	2082	2094	2100	2130	2136
		2142	9302#	9688										
DH101	050716	6413	9327#	9488	9501	9516	9533	9540	9555	9570	9590	9603	9616	9629
		9648	9667	9690	9717	9728	9739							
DH102	051005	6424	9337#	9490	9503	9518	9535	9542	9557	9572	9592	9605	9618	9631
		9650	9669	9692	9719	9730	9741							
DH103	051023	2118	9340#											
DH104	051046	9344#	9678	9696										
DH105	050534	6411	9305#											
DH106	051066	9347#	9694											
DH200	050557	9309#	9492	9505	9520	9594	9633	9652						

[illegible]

DRVRS	003137	2926#	3567*	6381*										
DRVHRD=	000020	2711#	6852	7341	7536	7667								
DRVLST	005512	2985#	3620	3625	3671	3735	3739	3806	4016	4262	4282	4335	4414	4632
		5037	5160											
DRVMSK=	000007	2518#	7243	7284	7435	7880								
DRVNO	011725	3462#												
DRVPOI=	000004	2709#	7442	7510										
DRVPOS=	000002	2708#	7442	7469	7931									
DRVPTR	005554	2989#	3744*	3809*	4004*	4005	4016*	4329	4341	4380	4401*	4402	4414*	5034
		5067	5109											
DRVSEQ	011265	3406#												
DRVSER	023736	5093												
DRVSZO=	010000	2722#	5424*											
DRVTSI	013734	3700	6671	7272	8034									
DRVUSE=	000001	3700	3774	3779#										
DRY	= 000200	2707#	7894											
DR.CLR=	000005	2567#												
DR.SEL=	000001	2490#	7322	7485	7527	7551	7665							
DRDXOP	011610	2489#	7392	7471	7786	7793	7800	7807	8011					
OSC	= 040000	3448#												
DSWR	= 177570	2572#												
DTBAII=	100000	1349#	1526	3546										
DTE	= 010000	2725#	4095	4240	6140	7894	7993							
DTYE	= 000040	2552#	6015	6745	7662									
DT100	052276	2544#	6708	7662										
		1657	1663	1669	1675	1681	1687	1693	1699	1705	1711	1717	1723	1729
		1735	1741	1747	1753	1759	1765	1771	1777	1783	1789	1795	1801	1807
		1813	1819	1825	1843	1849	1855	1861	1867	1873	1879	1885	1891	1897
		1903	1909	1915	1921	1927	1933	1951	1963	1969	1975	1981	1987	1993
		1999	2005	2011	2017	2059	2065	2071	2077	2083	2095	2101	2125	2131
		2137	2143	9469#										
DT102	052314	9472#												
DT201	052320	9473#												
DT202	052336	2107	9476#											
DT203	052342	9477#												
DT601	052362	2029	2035	2041	2047	2053	2113	9480#						
DT602	052370	1957	9481#											
DZR6Q	007572	3252#	3571											
ECCNC	= 000040	3242#	6832											
ECCW	= 020000	2588#												
ECH	= 000100	2545#	6830	7260	7426	7662								
ECOBAD	021014	4793#												
EMTVEC=	000030	1438#	3531*											

CZR6Q80 RK6 DR CPT PRG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 200
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ C198

EM113	050430	2129	9292#	
EM114	050456	2135	9296#	
EM115	050473	2141	9299#	
EM12	045654	1709	9033#	
EM13	045675	1715	9036#	
EM14	045717	1721	9040#	
EM15	045745	1727	9044#	
EM16	045766	1733	9047#	
EM17	046003	1739	9050#	
EM2	045443	1661	9006#	
EM20	046024	1745	9053#	
EM21	046041	1751	9056#	
EM22	046061	1757	9059#	
EM23	046103	1763	9063#	
EM24	046134	1769	9068#	
EM25	046154	1775	9071#	
EM26	046177	1781	9075#	
EM27	046211	1787	9077#	
EM3	045467	1667	2147	9010#
EM30	046234	1793	1913	9081#
EM31	046261	1799	1919	9085#
EM32	046302	1805	9088#	
EM33	046323	1811	9091#	
EM34	046345	1817	2117	9095#
EM35	046365	1823	9098#	
EM36	046421	1829	9103#	
EM37	046454	1835	9108#	
EM4	045512	1673	9014#	
EM40	046517	1841	9114#	
EM41	046563	1847	9121#	
EM42	046621	1853	9127#	
EM43	046651	1859	9132#	
EM5	045533	1679	9017#	
EM52	046706	1901	9137#	
EM53	046734	1907	9141#	
EM56	046761	1925	1931	9145#
EM6	045552	1685	9020#	
EM60	047000	1937	9148#	
EM61	047034	1943	9153#	
EM62	047102	1949	9160#	
EM63	047126	1865	1961	9164#
EM64	047173	1871	1967	9171#
EM65	047236	1877	1973	9177#
EM66	047264	1883	1979	9181#
EM67	047317	1889	1985	9186#
EM7	045576	1691	9024#	
EM70	047356	1895	1991	9192#
EM71	047425	1997	9199#	
EM72	047465	2003	9205#	
EM73	047534	2009	9212#	
EM74	047570	2015	9217#	
EM75	047610	2021	9220#	
EM76	047640	2027	9224#	
EM77	047722	2033	2039	9233#
ENDTST	011170	3392#	4413	
EQUALS	012164	3501#		

ERRCNT	003135	2924#	3568#	5014#	6195#	6360#	6362	6365	6367#	6492#	6606#	6777#	6822#	6912#
ERRCOM	005426	6932	6935	6952#										
ERRFRE	030272	2964#	6543#	6544#	9469									
ERRHDL	031504	2842	4870	6357#	6880									
ERRLMT	003136	2843	3845	4691	4723	4869	4948	5802	6010	6017	6209	6604#	6879	6899
ERRVEC=	000004	2925#	6932											
ERZENT	031524	1431#	3544	3545#	3556#	3564#	3565#	8829	8830#	8832#	8835#			
E.CCLR=	000001	6608#	6670	6783	6808	6904								
E.CERR=	001000	2851#	7601											
E.CLAT=	000020	2860#	7675											
E.CMT0=	040000	2855#	7327	7491	7532	7556								
E.CONT	003062	2862#												
		2845#	6494	6498	6502	6509	6513	6517	6521	6524	6527	6530	6534#	7226#
		7264#	7327#	7337#	7408#	7430#	7453#	7459#	7491#	7532#	7556#	7601#	7653#	7675#
		7814#	8048#	8102#	8110#	8123#								
E.DLT =	000400	2859#	7408											
E.DPAR=	002000	2861#	7337	7814										
E.ILLD=	000100	2858#	8102											
E.MDS =	100000	2863#	7226	7653	8048									
E.NOAT=	000002	2852	7453											
E.SCLR=	000040	2857	8123											
E.UATT=	000004	2853#	7459											
E.UDAT=	000010	2854#	7264	7430										
E.UNLD=	020000	2723#	7345	7465	7540	7550								
FACTBS	011762	3469#	6032											
FINADR	024542	5589#	5719	5904										
FINCYL	005470	2977#	5619#	5721	5739	5745	5906	5926	5932					
FINMEM	027202	6076#	6127											
FINSEC	005473	2979#	5607#	5723	5743	5747	5908	5930	5934					
FINTRK	005472	2978#	5617#	5722	5741	5746	5907	5928	5933					
FMTE =	000020	2543#	6712	7662										
FORMAT	003134	2923#	4097	4242	4879	5082#	5086#	5461	5599	5978	5998	6103	7034	
GETPRM	021026	3748	3753	3762	4808#	4848								
GETTYP	024176	3592	5481#	5484	5485	5486	5491							
GNS =	***** U	1450	8987	8988	8989	8990	8991	8994	8995	8996	8997			
GO =	000001	2494#	3825	3827	4925	4928								
GTPKAL	034310	5278	6838	6849	7044#									
GTPRMS	027540	6169#	6206											
GTSWRG	021136	3589	4844#	5374	5822									
GTO	027552	6168	6172#											
HCRC =	000400	2547#												

CZR6Q80 RK6 DF CPT PRGG MACY11 30(1046)
CZR6QB.P11 02-DEC-77 10:46

02-DEC-77 11:48 PAGE 202
CROSS REFERENCE TABLE -- USER SYMBOLS

IOAE = 002000	2550#	6735												
IE = 000100	2495#	3821	4921	4959	5041	5113	6769	6799	7606	7670	7958	7999	8065	
	8132													
ILC = 000001	2537#	6696	7662											
ILF = 000004	2540#	6704												
INCRSC 027266	6088	6102#	6135											
INITSS 021170	3820	3840	3874	3897	3945	4000	4045	4079	4151	4228	4325	4869#	4920	
	4941	5040	5065	5073	5087	5112	5137	5425	5459	5972				
INTBLD 022642	3947	3967	4114	4237	5150#									
INTCHR 005442	2970#	4742*	4761	4778*	4793	5098*	5791	5795	5800*	5809	5813	5820		
INTMSK 003111	2894#	7301	7325	7346	7371	7457	7489	7530	7541	7554	7825*	7829*	7833*	
	7882*	7968												
INTR = 000300	2485#													
IOTVEC= 000020	1436#	3529*	3530*											
IR = 000100	2524#													
I.AERR 036604	7477	7483	7501	7510#										
I.ATTN 036134	7237	7281	7414#											
I.CCLR 037170	7315	7365	7596#	7668	8026	8093								
I.CSTS 037622	7308	7743#												
I.CST1 037644	3835	4936	7312	7364	7747#	8017								
I.DRV 003112	2899#	7882	7883											
I.ERR 035356	7309	7313#												
I.ERRA 035336	7308#	7376												
I.ERRC 035344	7246	7299	7311#	7444										
I.HDAL 035602	7292	7355#												
I.INTR 034722	3579	7216#												
I.ISRL 003104	2888#	7230	7233*	7236*	7607*	7904*	8058*	8131*						
I.ISSU 037252	7323	7393	7472	7486	7528	7552	7638#	7787	7794	7801	7808			
I.IIO 035202	7234	7251	7278#											
I.ATRN 037152	7228	7266	7276	7304	7316	7321	7324	7330	7339	7348	7351	7366	7368	
	7394	7397	7406	7410	7432	7448	7455	7461	7473	7487	7494	7506	7526	
	7529	7534	7543	7546	7553	7558	7567#							
I.STMT 037726	7320	7525	7784#	8038										
I.STOR 037474	7648	7698#												
I.UNLD 037052	7467	7550#												
KBDHOL 020634	3576	4734#												
LASTWC 005474	2980#	5590	5718*	5903*										
LDMEM 026042	5858#													
LDMEM 026054	5852#													
LDMO 026062	5861	5864#												
LEV2ER= 000400	3245#	6608	6663	6760	6766	6790	6796	6900	6958					
LF = 000012	1342#	8308	8314											

[illegible]

[illegible]

PSSCR2	007424	3203*	4158*	4549*													
PSW	= 177776	1346*	6950*	7023*													
PWRMSG	044540	8787*	8791*														
PWRVEC=	000024	1437*	3535*	3536*	8775*	8783*	8784*										
P.ASOF=	000032	2738*	6564	7488*	7489	7522*	7753*	7941*	7942								
P.A00 =	000040	2741*	4963	4971	6567	6991	7479*	7480	7496*	7497	7810*	8018*					
P.A01 =	000044	2743*	6569	7789*													
P.A10 =	000050	2745*	6571	7796*													
P.A11 =	000054	2747*	5430	6573	7803*												
P.BAHI=	000007	2700*	6151*	6553													
P.BALO=	000010	2701*	3917*	4094*	4163*	4239*	4986*	5467*	5842*	5983*	5995*	6148*	6556	7975			
		8072															
P.BAR =	000024	2735*	6563	7519*	7749*												
P.B00 =	000042	2742*	6568	7474*	7475	7811*	8019*										
P.B01 =	000046	2744*	6570	7790*													
P.B10 =	000052	2746*	6572	7013	7797*												
P.B11 =	000056	2748*	6574	7395*	7804*												
P.CMND=	000001	2694*	3833*	3841*	3920*	3923*	4084*	4088*	4176*	4179*	4471*	4507*	4510*	4513*			
		4516*	4875*	4934*	4942*	4979*	4981*	4983*	5016*	5074*	5426*	5464*	5804*	5845*			
		5974*	6544	6892*	6940*	6986*	7022*	7025*	7290	7907	7911	7923	7933	7939			
		7945	7948	7954	7979	7986	8005	8052	8069	8091	8100						
P.CS1 =	000016	2732*	3834*	4895	4935*	6557	6675	6693	7056	7089	7311*	7363*	7401*	7402*			
		7404*	7405	7514*	7516*	7743*	7746*	7897	7951*	7952*	7954*	7958*	7959	7983*			
		7986*	7988*	7989*	7999*	8000	8008*	8009*	8011*	8012	8014*	8015	8020	8059*			
		8060*	8062*	8065*	8066	8082*	8083*	8085*	8088	8107*	8108	8117*	8118				
P.CS1H=	000007	2699*	4879*	4880*	4988*	4989*	5460*	5973*	7033*	7034*	7401	7642	7951	7988			
		8008	8059	8073	8082												
P.CS2 =	000020	2733*	4689	6558	6632	6637	6642	6644	6650	6652	6700	6750	6843	7317			
		7517*	7747*	7905*	7930	7995*	7996	8007	8026	8030	8056*	8057	8081				
P.CYLN=	000002	2695*	3908*	3913*	3951*	3954*	3957*	3960*	3970*	3973*	4085*	4090*	4102*	4113*			
		4117*	4175*	4178*	4246*	4249*	4474*	4476*	4984*	5251*	5465*	5618	5975*	6077			
		6136*	6546	7021*	7399	7935	7977	8079									
P.DCYL=	000030	2737*	6081	6128	6559	6812	7044	7078	7521*	7755*							

K16

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:4602-DEC-77 11:48 PAGE 207
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0205

RKPRI	003052	2841#	3580	3605*	3756	3772*	7152	7876						
RKPRY	007727	3269#	3761											
RKVADR	007704	3265#	3752											
RKVEC	003050	2840#	3578	3601*	3602*	3751	3754*							
RKWC =	000002	2443#	6977	7700	7748	7976*								
RLS =	000010	2520#	8056											
ROTBLO	023022	3965	4120	4252	5204#									
RWBUFF	053342	3917	4093*	4094	4163	4239	4986	4994	5467	5472	5562	5566	5643	5692
		5836	5842	6166	9748#									
R.ABNL	040154	7165	7275	7350	7367	7545	7565	7825#	8035					
R.CONT	040200	7227	7265	7329	7338	7409	7431	7454	7460	7493	7533	7557	7602	7654
		7677	7815	7833#	8049	8103	8112	8125						
R.NORM	040166	7303	7396	7505	7829#	7972	8045	8133						
SAVPRS	005454	2975#	6163	6170										
SAVREG=	104407	4462	4535	4577	4630	5150	5204	5233	5272	5424	5561	5589	5637	5682
		5720	5834	5858	5862	5886	5890	5905	6029	6076	6126	6161	6169	6264
		6327	6380	6490	6542	6604	6970	7011	7077	8417	8458	8995#		
SAVWRD	003160	2952#												
SCLR =	000040	2523#	8106											
SCNDRV	021316	4920#	5069											
SCOPE=	104411	8997#												
SCOPE1	044562	8802#	8997											
SCRACH	005444	2971#	3080*	3683*	3688	3828*	3829*	4929*	4930*	5385*	5387*	5388	5398*	5399*
		5400	5686*	5728	5734*	5737	5894*	5915	5921*	5924				
SCRLST	005556	2990#	3653	3661*	3667	3683	3795*	3798*	3799*	3817	3852*			
SECLST	006572	3131#	5244	5291										
SEEK =	000117	2470#	4084	4176	4471	7923	7933							
SELDIV=	000101	2463#	3833	4934	8062									
SELF	011235	3399#	4343											
SERNM	011747	3467#	5429	5471										
SETTUP	022042	3873	3896	3944	3999	4044	4078	4150	4227	4324	5011#			
SKI =	000002	2539#	6725											
SOFTBS	011775	3471#	6065											
SPACE1	012150	3497#	4356											
SPACE2	012147	3496#	4361	6046	6052	6419	6423	6426	6429	6449	6454	6469		
SPACE3	012146	3495#												
SPACE4	012145	3494#												
SPACE6	012143	3493#												
SPAR =	020000	2503#	6675	7658										
SPDLSS=	000020	2563#	6689											
SRHTBS	027700	6225#	6266											
SRPASS	010307	3313#	3779*	3780	4019*	4020								
SRTSPL=	000111	2467#	7945											
STACK =	001100	1336#	3527	5011*	5012	5054	5061	5126	5133	5803	8786			
STALL	024420	5535#	6309											
STALLS	005432	2966#	3588*	5544										
STDLSL	006552	3121#	5152											
STKLMT=	177774	1347#												
STRCMD	030220	6313	6327#											
STR204	010450	3330#	4012*	4013										
STR220	010525	3338#	4022*	4023	4409*	4410								
SUBCLR=	000177	2484#	4513	4875	5016	6940	7022	8100						
SUBSYS	005510	2984#	3606*	3734*	3736	3807	4008	4405						
SVAL =	100000	2573#												
SVPRMS	027500	4100	4511	6161#	6207									
SWR	001140	1526#	3525	3546*	3548	3554*	3561*	4844	5095	5537	5735	5762	5922	5949

SWREG	000176	6382	6409	6437	8692	8699	8711	8714	9804	8824	8840
SWRMSG	011245	1452#	3554	4844	4846	4849*					
SW0	= 000001	1400#									
SW00	= 000001	1390#	1400								
SW01	= 000002	1389#	1399								
SW02	= 000004	1388#	1398								
SW03	= 000010	1387#	1397								
SW04	= 000020	1386#	1396								
SW05	= 000040	1385#	1395								
SW06	= 000100	1384#	1394								
SW07	= 000200	1383#	1393								
SW08	= 000400	1382#	1392								
SW09	= 001000	1381#	1391								
SW1	= 000002	1399#									
SW10	= 002000	1380#									
SW11	= 004000	1379#									
SW12	= 010000	1378#									
SW13	= 020000	1377#	6382								
SW14	= 040000	1376#									
SW15	= 100000	1375#									
SW2	= 000004	1398#									
SW3	= 000010	1397#									
SW4	= 000020	1396#									
SW5	= 000040	1395#									
SW6	= 000100	1394#									
SW7	= 000200	1393#									
SW8	= 000400	1392#									
SW9	= 001000	1391#									
SYSDRV	010103	3289#	3627*	3628							
S.ACLO=	000100	2608#									
S.BRHM=	000100	2623#									
S.BRKE=	040000	2645#									
S.CART=	000400	2625#									
S.DCLO=	010000	2615#									
S.DIB =	002000	2641#									
S.DOOR=	000200	2624#									
S.DRA =	000040	2594#									
S.DROT=	020000	2616#									
S.DRY =	000200	2596#	4963								
S.OSC =	040000	2603#	7332	7480	7497	7560					
S.FLT =	000200	2609#	7475								
S.FORM=	001000	2598#									
S.FWD =	002000	2627#									
S.HDFL=	000200	2638#									
S.HDHM=	000040	2622#									
S.ICYL=	000040	2607#									
S.ILF =	000400	2610#									
S.LIMD=	020000	2644#									
S.LOAD=	010000	2629#									
S.MHD =	000400	2639#									
S.NMOV=	010000	2643#									
S.OFF =	002000	2599#									
S.PAR =	001000	2611#	7335	7812							
S.PIP =	020000	2602#	7343	7538							
S.PLO =	004000	2642#									

[illegible]

WP16HT = 040000

4812	4822	4823	4847	4953	4954	4967	4968	4975	4976	4993	4998	4999
5042	5043	5044	5051	5052	5114	5115	5116	5123	5124	5355	5361	5367
5373	5384	5388	5392	5400	5405	5407	5408	5410	5428	5429	5448	5470
5471	5475	5481	5482	5489	5490	5815	5816	6032	6033	6034	6035	6041
6042	6046	6052	6056	6065	6396	6397	6398	6402	6403	6408	6411	6412
6413	6414	6419	6422	6423	6424	6425	6426	6429	6432	6433	6435	6449
6454	6458	6464	6469	6470	6472	8278	8525	8590	8665	8694	8702	8787
8987#												
6380#	8701											
2922#	3823	3906	3949	3968	4880	4923	4989	5495*	5505*			
6045	6051	6055	6418	6428	6431	6451	8988#					
8990#												
4809	6405	8989#										
3346#	5043	5115										
3273#	3720											
2826#	7300*	7301	7325	7370*	7371	7450*	7451	7457	7488	7522	7530	7554
7705*	7967*	7968										
2823#	7519	7701*										
2818#	7245*	7297*	7298	7311	7322*	7355*	7357	7363	7392*	7414*	7416	7446
7471*	7485*	7515*	7516	7527*	7551*	7597*	7598	7638	7642*	7643*	7645	7649
7658	7698*	7745*	7746	7786*	7793*	7800*	7807*	7961*	7962	7964		
2820#	7223*	7224	7239	7242	7254	7283	7383*	7384	7420	7434	7517	7639*
7640*	7641	7651	7660	7699*								
2824#	7520	7702*										
2835#												
2825#	7521	7706*										
2829#	7248*	7249	7524	7703*								
2828#	7257*	7260	7423*	7426	7523	7662	7704*					
2830#	7707*											
2831#	7332	7343	7479	7496	7538	7560	7708*	7789	7796	7803	7810	
2832#	7335	7395	7474	7709*	7790	7797	7804	7811	7812			
2834#	7711*											
2833#	7710*											
2822#	7518	7700*										
2710#	6867	7306										
2526#	6644	6650	7239	7317	7660	8030						
2466#	7911											
2554#	6720											
2531#	6632	7254	7420	7660								
2945#												
2566#												
2946#												
2532#	6843	7254	7420	7660								
2932#	5985*	5987	6002*	6004	6018*							
3243#	6846	6923										
2976#	5638	5685	5717	5864	5892	5902	6091*	6131	6164*	6165*		
2944#		</										

02R6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 211
02R6Q8.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- USER SYMBOLS

WRTLOK	011347	3417#	4976											
WRTSEC	025770	5246#	5834#											
W.DRV	003124	2915#	7154#	7269#	7295#	7347#	7362#	7375#	7389#	7437#	7504#	7513#	7542#	7564#
		7835#	7884#	7947#	7950#	7971#	8023#	8041#						
J.MILI	003070	2868#	7149	8115										
W.MIN	003076	2884#	7947											
W.MTIM	003066	2867#	7147#	7149#										
W.SEC	003072	2881#	7884											
W.TIME	003110	2890#	7150	7156#	7268#	7296#	7346#	7361#	7374#	7390#	7436#	7503#	7512#	7541#
		7563#	7834#	7883#	7970#	8022#	8040#							
W.WCK =	000200	2715#	7894	7955	7981	7987								
W.WTCH	034572	6317	7142#	7960	7966	8013								
W.BSEC	003074	2883#	7347	7542	7950									
\$APTHO	001000	1483	1489#											
\$ASTAT=	*****	8888	8903											
\$ATYC	045002	8859	8861#											
\$ATY1	044756	8857#												
\$ATY3	044764	8263	8858#											
\$ATY4	044774	8707	8860#											
\$AUTOB	001134	1523#												
\$BASE	001404	1634#	3596											
\$BDADR	001122	1518#												
\$BDDAT	001122	1520#												
\$BELL	001320	1585#	8694	8726										
\$CHARC	042632	8280#	8290#	8297	8306#	8311#								
\$CKSWR=	*****	8994												
\$CNTAG	001100	1506#	3522	3523	3531	3537	3538							
\$CM1 =	000040	1538#	1539#	1540#	1541#	1542#	1543#	1544#	1545#	1546#	1547#	1548#	1549#	1550#
		1551#	1552#	1553#	1554#	1555#	1556#	1557#	1558#	1559#	1560#	1561#	1562#	1563#
		1564#	1565#	1566#	1567#	1568#	1569#	1570#						
\$CM2 =	000100	1538#	1539#	1540#	1541#	1542#	1543#	1544#	1545#	1546#	1547#	1548#	1549#	1550#
		1551#	1552#	1553#	1554#	1555#	1556#	1557#	1558#	1559#	1560#	1561#	1562#	1563#
		1564#	1565#	1566#	1567#	1568#	1569#	1570#						
\$CM3 =	000040	1536#	1538											
\$CM4 =	000016	1570#	1571#	1572#	1573#	1574#	1575#	1576#	1577#	1578#	1579#	1580#	1581#	1582#
		1583#	1584#											
\$CNTLG	044423	8764#												
\$CNTLU	044416	8763#												
\$CPUOP	001356	1608#												
\$CRLF	001325	1587#	3617	3773	4370	4373	4392	4397	4762	5405	5410	5448	5475	5481
		6035	6042	6056	6402	6412								

[illegible]

CZR6QBD RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 213
CZR6QB.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- USER SYMBOLS

\$OVER	044742	8825	8843	8851*												
\$PASS	001336	1598*	3558*	4433*	4434*	4450										
\$PASTM	001006	1493*														
\$PWACT	044536	8780*	8781*	8790*												
\$PWARN	044452	3535	8775*	8783												
\$PWARP	044464	8775	8780*													
\$QUES	001324	1586*	3638	3715	3730	3776	4794	4823	5052	5124	5408	5490	8314	8726		
\$RAND	042252	5539	8210*													
\$RDOCHR	044276	8742*	8994													
\$RDEEC=	*****	8995														
\$RDLIN=	*****	8995														
\$RDOCT=	*****	8995														
\$RDSZ =	000000	8763*														
\$REGAD	001160	1536*														
\$REGO	001162	1538*	6545	9469												
\$REG1	001164	1539*	9469													
\$REG10	001202	1546*	5716*	5717*	5718	5749*	5752	5901*	5902*	5903	5936*	5939	6770*	6800*		
		6839*	9473	9481												
\$REG11	001204	1547*	5750*	5937*	6771*	6801*	6840*	9473	9481							
\$REG12	001206	1548*	5751*	5938*	6772*	6802*	6833*	6835*	9473							
\$REG13	001210	1549*	5753*	5755*	5757*	5758*	5940*	5942*	5944*		9473	9481				
\$REG14	001212	1550*	5752*	5754*	5756*	5939*	5941*	5943*	9476	9481						
\$REG15	001214	1551*	5759*	5760*	5946*	5947*	9476	9481								
\$REG16	001216	1552*	9477	9481												
\$REG17	001220	1553*	9477													
\$REG2	001166	1540*	9469													
\$REG20	001222	1554*	9477													
\$REG21	001224	1555*	9477													
\$REG22	001226	1556*	9477													
\$REG23	001230	1557*	9477													
\$REG24	001232	1558*	9477													
\$REG25	001234	1559*	9477													
\$REG26	001236	1560*	6416	6576												
\$REG27	001240	1561*														
\$REG3	001170	1541*	9469													
\$REG30	001242	1562*														
\$REG31	001244	1563*														
\$REG32	001246	1564*														
\$REG33	001250	1565*														
\$REG34	001252	1566*														
\$REG35	001254	1567*														
\$REG36	001256	1568*	6555*	9472												

[illegible]

CZR6QBO RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 215
CZR6QB.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0213

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046)
CZR6Q8.P11 02-DEC-77 10:46

CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

CZR6Q80 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 217
CZR6Q8.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- MACRO NAMES

CKEXIT	2167#	6310													
COMMEN	1443#														
ENDCOM	1443#														
ERROR	1337#														
	6511	3837	4938	4957	5730	5748	5761	5917	5935	5948	5993	6366	6496	6500	6507
	6667	6515	6519	6523	6526	6529	6532	6533	6634	6639	6646	6648	6654	6656	6665
	6727	6669	6673	6677	6682	6687	6691	6695	6698	6702	6706	6710	6714	6718	6722
	6854	6732	6737	6742	6747	6752	6762	6779	6782	6792	6805	6807	6824	6841	6850
	1443#	6859	6864	6869	6903	6905	6936								
ESCAPE	1443#														
GETPRI	1443#														
GETSWR	1443#														
MSG	3859#	3861	3882#	3884	3933#	3935	3978#	3980	4029#	4031	4052#	4054	4129#	4131	4198#
	4200	4302#	4304												
MULT	1443#														
NEWTST	1443#	3859	3882	3933	3978	4029	4052	4129	4198	4302					
NEXTAD	2163#														
NXTAD	2159#														
PARMBK	1#	2751	2783												
PARMDF	1#	2657													
POP	1443#	8229	8660	8900	8901	8949									
PUSH	1443#	8210	8619	8861	8863	8884	8929								
P.DRVR	1#	2815													
REGDEF	1#	2439													
REPORT	1443#														
R.DRIV	1#	7100													
R.QDRV	1#														
SCOPE	1338#	3871	3894	3942	3997	4042	4076	4148	4225	4322	4431				
SETPRI	1443#														
SETTRA	8979#	8988	8989	8990	8991	8994	8995	8996	8997						
SETUP	1443#	3521													
SKIP	1443#														
SLASH	1443#														
SPACE	1443#														
STARS	1295	1302	1443#	1465	1476	1478	1485	1501	1589	1592	3611	3615	3859	3870	3882
	3893	3933	3941	3978	3996	4029	4041	4052	4075	4129	4147	4198	4224	4302	4321
	4425	4453	4461	4530	4534	4571	4576	4623	4629	4683	4688	4710	4715	4730	4732
	4767	4775	4784	4791	4802	4807	4837	4843	4859	4867	4885	4890	4905	4918	5006
	5010	5025	5032	5103	5107	5142	5149	5198	5203	5224	5232	5259	5271	5321	5342
	5419	5423	5454	5458	5527	5533	5557	5560	5579	5588	5626	5635	5673	5681	5779
	5788	5828	583												

CZR6QB0 RK6 DR CPT PROG MACY11 30(1046) 02-DEC-77 11:48 PAGE 218
 CZR6QB.P11 02-DEC-77 10:46 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0215

SDCBN	1#														
SOCTBN	1#	8143													
SSCMRE	1499#	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566
	1567	1568	1569												
SSCMTM	1499#	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
SSESCA	1443#														
SSNEWT	1443#	3859	3882	3933	3978	4029	4052	4129	4198	4302					
SSSET	8979#	8988	8989	8990	8991	8994	8995	8996	8997						
SSSETM	3558#														
SSSKIP	1443#														
.EQUAT	1292#	1333													
.HEADE	1292#	1307													
.SETUP	1292#	3506													
.SWRHI	1292#	1320													
.SWRLO	1292#	1331#													
.SACT1	1292#	1463													
.SAPTB	1590#														
.SAPTH	1292#	1474													
.SAPTY	1292#	8854													
.SCATC	1292#	1444													
.SCMTA	1292#	1499													
.SDB20	1292#	8445													
.SDB20	1292#	8406													
.SEOP	1292#	4423													
.SERRO	1292#	8674													
.SPOWE	1292#														
.SRAND	1292#	8199													
.SREAD	1292#	8726													
.SSAVE	1292#	8911													
.SSCOP	1292#	8811													
.SSIZE	1292#														
.SSUPR	1292#	8507													
.STRAP	1292#	8956													
.STYPD	1292#	8607													
.STYPE	1292#	8235													
.STYPO	1292#	8530													

. ABS. 055163 000

ERRORS DETECTED: 0

RM03:CZR6QB, RM03:CZR6QB.SEQ/SOL/CRF/NL: TOC/DOC=RM03:DRIV10.P11/EQ:QNEWSW, RM03:CZR6QB.P11

RUN-TIME: 36 28 2 SECONDS

RUN-TIME RATIO: 617/67=9.1

CORE USED: 47K (93 PAGES)

DOCUMENT PAGES: 215