

RK611/RK06

SUBSYSTEM VERIFICATION 2
CZR6ND0

AH-9142D-MC

MAR 1978

COPYRIGHT © 76-78

digital

FICHE 1 OF 2

MADE IN USA



RK611/RK06

SUBSYSTEM VERIFICATION 2
CZR6ND0

AH-9142D-MC

MAR 1978

COPYRIGHT © 76-78

digital

FICHE 2 OF 2

MADE IN USA

B01

00010000
CZR6ND.P11

PAGE 2
10-JAN-78 10:05

780223

PDP18E010001

CZR6NDO RK611/06 SS VERIF 2

MACY11 30A(1052) 10-

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-9140D-MC
PRODUCT NAME: CZR6NDO RK611/06 SS VERIF 2
DATE: FEB 1978
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: DAVE HOFFMAN

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

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

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

COPYRIGHT (C) 1976, 1978 BY DIGITAL EQUIPMENT CORPORATION

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

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	HARDWARE REQUIREMENTS
2.1	REQUIREMENTS FOR SUBSYSTEM TESTS
2.2	REQUIREMENTS FOR HEAD ALIGNMENT AID
3.0	PRELIMINARY PROGRAM REQUIREMENTS
4.0	GENERAL PROGRAM CONSIDERATIONS
4.1	SYSMAC
4.2	XXDP
4.3	CHAIN MODE
4.4	DUMP MODE
4.5	ACT/APT
4.6	AUTOMATIC MODE
4.7	DUMP MODE
4.8	APT ETABLE DEFINITIONS
4.9	DUAL-ACCESS
4.10	MEMORY MANAGEMENT
4.11	MEMORY PARITY CHECK
4.12	BAD SECTORS
4.13	EXECUTION TIME
5.0	PROGRAM LOADING
6.0	STARTING PROCEDURE
6.1	STARTING ADDRESSES
6.2	SWITCH REGISTER OPTIONS USED
7.0	OPERATOR ACTION
8.0	PROGRAM ACTION
8.1	DESCRIPTION OF OPERATING PARAMETERS
8.2	SELECTION OF OPERATING PARAMETERS
8.3	DRIVE SELECTION
8.4	TEST SELECTION
8.5	LIST TESTS, (L)
8.6	CHANGE TESTS, (C)
8.7	INPUT PARAMETERS AND RUN TESTS, (I)
8.8	CONTROL Z (IZ) FUNCTION
8.9	CONTROL C (IC) FUNCTION
8.10	PARAMETER LIST ALTERATION
8.11	TYPE LIST, (T)
8.12	OPEN LIST, (O)
8.13	SET INDIVIDUAL PARAMETER, (S)
8.14	RUN TESTS, (R)

DO1

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 4

SEQ 0003

96
97
98
99
100

8.2.3.5
8.2.3.6

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

TABLE OF CONTENTS (CONT'D)

8.2.4	SPECIAL PARAMETER SPECIFICATIONS
8.2.4.1	PT - DATA PATTERN SELECT WORD
8.2.4.2	CS - CONTROL SWITCH WORD
8.3	DATA PATTERNS
9.0	DESCRIPTION OF TESTS
9.1	TEST 1 - OFFSET-TO-FAILURE MEASUREMENT
9.2	NPR/MAIN MEMORY TESTS
9.2.1	TEST 2 - NPR/MEMORY WORD ADDRESSING TEST
9.2.2	TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST
9.2.3	TEST 4 - NPR/MEMORY DATA PATTERN TEST
9.3	TEST 5 - UNIBUS CONTENTION TEST
9.4	TEST 6 - MULTI-DRIVE INTERFERENCE TEST
10.0	ERROR REPORTING
10.1	COMMON ERRORS
10.2	ERROR HANDLING
10.3	ERROR PRINTOUT EXAMPLES
11.0	RK06-07 HEAD ALIGNMENT AID
11.1	HARDWARE REQUIREMENTS
11.2	OPERATIONAL MODES
11.2.1	MANUAL SELECT MODE
11.2.1.1	MANUAL SELECT ALIGNMENT
11.2.1.2	MANUAL SELECT VERIFY
11.2.1.3	MANUAL SELECT EXERCISE
11.2.2	AUTO SELECT MODE
11.2.2.1	AUTO SELECT ALIGNMENT
11.2.2.2	AUTO SELECT VERIFY
11.2.2.3	AUTO SELECT EXERCISE
11.3	ALIGNMENT AID ERROR MESSAGES
APPENDIX A	SAMPLE ADDRESS 200 DEFAULT RUN
APPENDIX B	SAMPLE ADDRESS 204 RUN
APPENDIX C	SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN
APPENDIX D	SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 2 OF THE RK611/RK06-RK07 SUBSYSTEM VERIFICATION TESTS, WHICH PROVIDE A FUNCTIONAL SHAKEDOWN OF THE ENTIRE SUBSYSTEM, INCLUDING THE UNIBUS INTERFACE AND ACCESS TO MAIN MEMORY. THE TESTING IN PART 2 EMPLOYS WORST-CASE SITUATIONS INVOLVING HEAD OFFSETTING, MEMORY ADDRESSING AND DATA TRANSFER, UNIBUS CYCLE CONTENTION, AND MULTIPLE DRIVE OPERATIONS. ADDITIONALLY, AN RK06-07 HEAD ALIGNMENT AID IS PROVIDED TO FACILITATE ON-LINE ALIGNMENT OF DRIVE HEADS.

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

RK611 REGISTER ADDRESS
RK06-07 VECTOR ADDRESS
RK06-07 PRIORITY LEVEL
DRIVE (S) TO BE TESTED
TEST (S) TO BE RUN
NUMBER OF TEST ITERATIONS
INITIAL DISK ADDRESS ON TRANSFERS
DATA PATTERNS USED
STALL TIME BETWEEN OPERATIONS

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

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

2.0 HARDWARE REQUIREMENTS

2.1 REQUIREMENTS FOR SUBSYSTEM TESTS

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

PDP-11/04, (05, 10 MFG. ONLY), 20, 34, 35, 40, 45, 50, 70 OR PDQ
16 K MEMORY
CONSOLE TELETYPE
RK06-07 UNIBUS CONTROLLER (RK611)
1 TO 8 RK06/07 DRIVES
1 TO 8 RK06/07 DISK CARTRIDGES (FORMATTED IN 20 OR 22 SECTOR FORMAT)

2.2 REQUIREMENTS FOR HEAD ALIGNMENT AID

205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260

ADDITIONAL HARDWARE IS REQUIRED BY THE RK06 HEAD ALIGNMENT AID:

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

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE SUBSYSTEM VERIFICATION PROGRAM, THE RK611 CONTROLLER DIAGNOSTIC (MAINDEC-11-CZR6A THRU CZR6E AND CZR6K) AND RK06 DRIVE DIAGNOSTIC (MAINDEC-11-CZR6H THRU CZR6J) SHOULD FIRST BE RUN, TO RESOLVE BASIC, SOLID HARDWARE FAULTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 SYSMAC

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

4.2 XXDP

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

4.2.1 CHAIN MODE

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

4.2.2 DUMP MODE

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

THIS IS NECESSARY.

4.3 ACT/APT

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

4.3.1 AUTOMATIC MODE

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

4.3.2 DUMP MODE

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

4.3.3 APT ETABLE DEFINITIONS

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

1. SOFTWARE ENVIRONMENT

=1 IF APT SCRIPT MODE
=0 IF STANDALONE MODE

2. ENVIRONMENT MODE BYTE

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

3. SWITCH 1 (SOFTWARE SWITCH REGISTER)

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

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

4. SWITCH 2 (USER SWITCH REGISTER)
NOT USED

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

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

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

8. INTERRUPT VECTOR 2
NOT USED

9. BUS PRIORITY 2
NOT USED

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

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

4.4 DUAL-ACCESS

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

4.5 MEMORY MANAGEMENT

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

4.6 MEMORY PARITY CHECK

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, ALL

373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428

TESTING IS DONE WITH PARITY CHECK ENABLED. PARITY ERRORS ARE
REPORTED, AND TESTING IS ABORTED, UNLESS THE BYTE LABELED MEMABT WAS
PREVIOUSLY LOADED WITH A NON-ZERO VALUE.

4.7 BAD SECTORS

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

4.8 EXECUTION TIME

EXECUTION TIME IS DEPENDENT UPON PARAMETERS INPUT BY THE OPERATOR
(SUCH AS STALL TIME OR ITERATION COUNT), AND TO A GREAT DEGREE, UPON
THE AMOUNT OF MEMORY. HOWEVER, THE "AVERAGE TIME" REQUIRED TO RUN A
QUICK VERIFICATION (FIRST PASS) IS 2 MINUTES (FOR A 64K SYSTEM). A
SUBSEQUENT PASS WITH PARAMETERS DEFAULTED REQUIRES 3 MINUTES PER DRIVE
(ON A 64-K SYSTEM).
NOTE: TIMES ARE APPROX DOUBLED FOR RK07 TESTING.

5.0 PROGRAM LOADING

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

6.0 STARTING PROCEDURE

6.1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS OF TESTS 1-6 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS, RK06/07 UNIBUS ADDR. AND INTERRUPT
VECTOR FOR TESTS 1-6
224 HEAD ALIGNMENT AID START ADDRESS

NOTE

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

6.2 SWITCH REGISTER OPTIONS USED

THIS PROGRAM IS DESIGNED TO ALLOW THE USE OF THE HARDWARE SWITCH REGISTER IF PRESENT, OR THE SYSMAC-SUPPORTED SOFTWARE SWITCH REGISTER (IF HARDWARE SWR IS NOT PRESENT OR IS SET TO ALL ONES). IN EITHER CASE, THE FOLLOWING OPTIONS ARE IMPLEMENTED WHEN THE APPROPRIATE BITS ARE SET TO 1:

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

NOTE

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

7.0 OPERATOR ACTION

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

8.0 PROGRAM ACTION

8.1 DESCRIPTION OF OPERATING PARAMETERS

AFTER THE PROGRAM IS STARTED, IT TYPES ITS IDENTIFICATION AS FOLLOWS:
"CZR6NDD RK11/RK06-RK07 SUBSYSTEM VERIFICATION:PART 2" FOLLOWED BY:
"LAST PHYS MEM ADRS=XXXXXXXX". THEN, EITHER THE TESTS BEGIN EXECUTION

WITH DEFAULT PARAMETERS, OR AN OPERATOR INTERACTIVE MODE IS ENTERED (SEE STARTING ADDRESSES, SECTION 6.1). IN THIS MODE OPERATING PARAMETERS MAY BE LISTED OR ALTERED BY VARIOUS MEANS. ALL PARAMETERS ARE TREATED AS OCTAL NUMBERS, AND THE PARAMETER LIST IS SHARED AMONG THE TESTS.

THE FOLLOWING IS THE LIST OF OPERATING PARAMETERS, (WHICH APPLY TO THE CURRENT SELECTION OF DRIVES AND TESTS). AFTER EACH, IS INDICATED THE VALID RANGE OF VALUES FOR THAT PARAMETER (IN OCTAL), AND ITS DEFAULT VALUE. ALL PARAMETERS ARE ENTERED AS OCTAL NUMBERS, AND THE PARAMETER MNEMONICS SHOWN ARE THOSE USED BY THE PROGRAM.

FC	FIRST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, FC=0-631, DEFAULT=0
LC	LAST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, LC=0-632, DEFAULT=632
FT	FIRST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, FT=0-2, DEFAULT=0
LT	LAST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, LT=0-2, DEFAULT=2
SO	FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. SO=0-23, DEFAULT=0.
S1	LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. S1=0-23, DEFAULT=23.
S2	FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE, S2=0-25, DEFAULT=0.
S3	LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE. S3=0-25, DEFAULT = 25.
PT	DATA PATTERN SELECT WORD (SEE SECTION 8.2.4.1 FOR DETAILS) PT=0-177777, DEFAULT=0
CS	CONTROL SWITCH WORD (SEE SECTION 8.2.4.2 FOR DETAILS) CS=0-000062, DEFAULT=0
ST	NUMBER OF UNIT STALL TIMES WITH WHICH TO STALL (DELAY) BETWEEN RK06 COMMANDS

8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)

485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539

8.2.1 DRIVE SELECTION

AFTER AN ADDRESS 204 START, THE PROGRAM INDICATES INPUT MODE BY TYPING "PARAMETER INPUT MODE". THEN, THE RK611 REGISTER ADDRESS, RK06/07 VECTOR ADDRESS AND RK06 PRIORITY ARE OPENED FOR POSSIBLE MODIFICATION AS FOLLOWS:

RK06-07 BUS ADR = 177440 NEW=(TYPE NEW VALUE HERE)
RK06-07 VEC ADR = 210 NEW=(TYPE NEW VALUE HERE)
RK06-07 PRIORITY= 5 NEW=(TYPE NEW VALUE HERE)

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

PARAMETER INPUT MODE

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

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

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

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

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

8.2.2 TEST SELECTION

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

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

THEN, THE FOLLOWING REQUEST IS MADE :

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

THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.

8.2.2.1 LIST TESTS, (L)

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

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

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

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

8.2.2.2 CHANGE TESTS, (C)

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

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

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

8.2.2.3 INPUT PARAMETERS AND RUN TESTS, (I)

IF (I) IS TYPED, THE PROGRAM ENTERS THE PARAMETER LIST ALTERATION

ROUTINE, DESCRIBED IN SECTION 8.2.3, IN WHICH OPERATING PARAMETERS MAY BE INPUT, AND TESTING BEGUN.

8.2.2.4 CONTROL Z (↑Z) FUNCTION

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

8.2.2.5 CONTROL C (↑C) FUNCTION

IF THE OPERATOR WISHES TO TERMINATE L, C, OR I MODE, AND RETURN TO REQUEST NEW DRIVES AND TESTS, CONTROL C (↑C) MAY BE TYPED.

8.2.3 PARAMETER LIST ALTERATION

THE PROGRAM NEXT TYPES THE FOLLOWING INSTRUCTIVE LINES:

T = TYPE LIST
O = OPEN LIST
S = SET INDIVIDUAL PARAM.
R = RUN TESTS
ENTER T O S OR R
* (CHARACTER GOES HERE)

THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.

8.2.3.1 TYPE LIST, (T)

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

FC = XXXXXX
LC = XXXXXX
ETC.

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

8.2.3.2 OPEN LIST, (O)

IF (O) IS TYPED THE PROGRAM THEN ASKS: "TO DEFAULT ALL PARAMETERS TYPE D<CR>, ELSE <CR>". IF JUST <CR> IS TYPED, THE PROGRAM SEQUENTIALLY OPENS EACH OPERATING PARAMETER IN THE LIST FOR ALTERATION

(IN THE ORDER SHOWN IN SECTION 8.1). EACH PARAMETER IS OPENED AND A NEW LINE IS TYPED, GIVING THE PARAMETER AND ITS CURRENT VALUE (IN OCTAL), FOLLOWED BY (*) ON THE SAME LINE. FOR EXAMPLE:

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

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

8.2.3.3 SET INDIVIDUAL PARAMETER, (S)

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

ENTER T,O,S, OR R

*S

> (ENTER PARAMETER AND VALUE HERE)

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

> FC = 600

> FS = 12

> IT = 1

> ETC.

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

8.2.3.4 RUN TESTS, (R)

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

8.2.3.5 CONTROL Z (↑Z) FUNCTION

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

8.2.3.6 CONTROL C (↑C) FUNCTION

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

8.2.4 SPECIAL PARAMETER SPECIFICATIONS

8.2.4.1 PT-DATA PATTERN SELECT WORD

THE OPERATOR SPECIFIES A WORD OF SIXTEEN BITS (6 OCT. DIGITS), TO SELECT UP TO ALL SIXTEEN OF THE PATTERNS, LISTED IN SECTION 9.5.1. THE NO. OF ANY BIT SET IN PT CORRESPONDS TO THE NO. OF A PATTERN CHOSEN. IF THE HIGH BIT OF PT (BIT 15) IS SET, THE USER WILL BE ASKED TO TYPE UP TO SIXTEEN DATA WORDS WHICH WILL BE LOADED INTO THE BUFFER FOR PATTERN 15. THE PROGRAM TYPES:

"SELECT USER DEFINED PATTERN 15"

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

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

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

8.2.4.2 CS- CONTROL SWITCH WORD

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

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

818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841

NOTE

OTHER BITS UNUSED

8.3 DATA PATTERNS

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

THE FOLLOWING IS THE LIST OF 16 SELECTABLE DATA PATTERNS (IN OCTAL), WITH NOTABLE FEATURES DESCRIBED:

F02

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 19

SEQ 0018

842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890

0	1	2	3
HIGH-LOW FREQUENCY MIX	HIGH FREQUENCY PHASE MIX	LOW FREQUENCY PHASE MIX	MAXIMUM PRECOMPENSATION PHASE MIX
177777	000000	052525	133333
177777	000000	052525	066666
177777	000000	052525	155555
052525	177777	125252	155555
052525	177777	125252	133333
052525	177777	125252	066666
177777	000000	052525	066666
177777	000000	052525	155555
052525	177777	125252	155555
052525	177777	125252	133333
177777	000000	052525	133333
052525	177777	125252	133333
177252	000000	052525	133333
177252	177777	125252	133333
172765	000000	052525	133333
172765	177777	125252	133333
4	5	6	7
ROTATING BOUNDARY PULSE PRECOMPENSATION	ROTATING CELL PULSE PRECOMPENSATION	ALL ZEROS	ALL ONES
121105	026455	000000	177777
150442	113226	000000	177777
064221	045513	000000	177777
132110	122645	000000	177777
055044	151322	000000	177777
026422	064551	000000	177777
013211	132264	000000	177777
105504	055132	000000	177777
042642	026455	000000	177777
021321	113226	000000	177777
110550	045513	000000	177777
044264	122645	000000	177777
022132	151322	000000	177777
011055	064551	000000	177777
104426	132264	000000	177777
042213	055132	000000	177777

G02

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 20

SEQ 0019

891				
892				
893				
894				
895				
896				
897				
898				
899				
900				
901				
902				
903				
904				
905				
906				
907				
908				
909				
910				
911				
912				
913				
914				
915				
916				
917				
918				
919				
920				
921				
922				
923				
924				
925				
926				
927				
928				
929				
930				
931				
932				
933				
934				
935				
936				
937				

9.0 DESCRIPTION OF TESTS

9.1 TEST 1 - OFFSET-TO-FAILURE MEASUREMENT

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

OFFSET-TO-FAILURE MEASUREMENTS:

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

ETC.

NOTE

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

9.2 NPR / MAIN MEMORY TESTS

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

NOTE

THERE IS NO PROGRAM RELOCATION (EXCEPT FOR THE XXDP OR ABSOLUTE LOADER, IF

PRESENT). HENCE, MEMORY IN WHICH THE PROGRAM RESIDES IS NOT TESTED. ALSO, THE UNIBUS I/O ADDRESSES ARE NOT TESTED. IF IT IS DESIRED THAT THE LOADER BE DESTROYED, SO THAT ADDITIONAL MEMORY MAY BE TESTED, LOCATION 170 (KILLDR:) MAY MANUALLY BE SET TO A NON-ZERO VALUE.

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

9.2.1 TEST 2 - NPR/MEMORY WORD ADDRESSING TEST

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

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

9.2.2 TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST

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

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

9.2.3 TEST 4 - NPR/MEMORY DATA PATTERN TEST

IN THIS TEST, ALL PHYSICAL MEMORY LOCATIONS ABOVE THE PROGRAM (NOT INCLUDING UPPER UNIBUS DEVICE ADDRESSES) ARE EXERCISED WITH UP TO 15 DATA PATTERNS, AS CHOSEN FROM THE FIRST 14 PATTERNS DESCRIBED IN SECTION 8.3, PLUS PATTERN 15 (USER DEFINED PATTERN). THE DATA PATTERNS DESIRED FOR THIS TEST ARE CHOSEN SEPARATELY, HOWEVER, AND THEY DEFAULT TO PATTERNS 8, 9, 10, AND 11.

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

9.3 TEST 5 - UNIBUS CONTENTION TEST

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

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

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

9.4 TEST 6 - MULTI-DRIVE INTERFERENCE TEST

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

THE TEST PROCEEDS AS FOLLOWS: IT IS FIRST DETERMINED WHICH DRIVE(S) (BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH. NEXT, A SEEK IS DONE TO CYLINDER FC (SCALED, IF NECESSARY) ON THE DRIVE UNDER TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH TO A DIFFERENT PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE FROM 10-67 MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN, A WRITE DATA IS BEGUN ON THE DRIVE UNDER TEST, AT THE CURRENT CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616 (DEC) WORDS IF 20 SECTOR FORMAT, OR 17,152 (DEC) WORDS IF 22 SECTOR FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA. THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN THAT TIME, ALL THE OTHER DRIVES SHOULD HAVE COMPLETED POSITIONING. NEXT, A WRITE CHECK IS DONE TO VERIFY THE DATA WRITTEN ON THE DRIVE UNDER TEST, AND A CHECK IS MADE TO INSURE THAT ALL OTHER DRIVES WHICH PERFORMED SEEKS WERE ABLE TO COMPLETE THEM SUCCESSFULLY, BY CHECKING ATTENTION BITS AND COMPARING DRIVE CYLINDER ADDRESSES TO EXPECTED VALUES.

10.0 ERROR REPORTING

10.1 COMMON ERRORS

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

1. UNIBUS PARITY ERROR
2. NON-EXISTANT MEMORY ERROR
3. NON-EXISTANT DRIVE ERROR
4. UNIT FIELD ERROR
5. SUBSYSTEM TIMEOUT
6. SERCON PARITY ERROR
7. DRIVE DETECTED PARITY ERROR
8. AC LOW
9. SPEED LOSS
10. ILLEGAL FUNCTION ERROR
11. PROGRAMMING ERROR
12. NON-EXECUTABLE FUNCTION ERROR
13. DRIVE TYPE ERROR
14. FORMAT ERROR
15. WRITE LOCK ERROR
16. DRIVE UNSAFE ERROR
17. SEEK INCOMPLETE ERROR
18. CYLINDER OVERFLOW ERROR
19. ILLEGAL CYLINDER ADDRESS ERROR
20. DRIVE OFF TRACK
21. DRIVE TIMING ERROR
22. DATA LATE ERROR

23. CONTROLLER TIMEOUT ERROR
24. OPERATION INCOMPLETE ERROR
25. HEADER VRC ERROR
26. DATA CHECK ERROR
27. WRITE CHECK ERROR
28. DATA MISCOMPARE
29. NO DRIVE RESPONSE - UFE AND NXD
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

10.2 ERROR HANDLING

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

10.3 ERROR PRINTOUT EXAMPLES

EXAMPLE 1:

** WRITE CHECK ERROR
TEST 2

PREVIOUS COMMAND:

DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
000000	000121	000016	000001	000000	175000

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

```
      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
```

EXAMPLE 2:

```
** DATA MISCOMPARE
TEST 2

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

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

PACK ADDRESS OF ERROR(S):
CYLNR  TRACK  SECTOR
000016 000001 000000

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

11.0 RK06 HEAD ALIGNMENT AID

THIS PROGRAM IS PROVIDED AS A SOFTWARE AID TO HEAD ALIGNMENT OF THE

1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327

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

11.1 HARDWARE REQUIREMENTS

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

11.2 OPERATIONAL MODES

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

" *** RK06-07 HEAD ALIGNMENT AID ***
TYPE H(CR) FOR INFORMATION- OTHERWISE, TYPE (CR) "

NEXT, THE PROGRAM REQUESTS THE DESIRED MODE OF OPERATION:

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

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

11.2.1 MANUAL SELECT MODE

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

THE PROGRAM FIRST ECHOS THE MODE:

1328
1329
1330
1331 "* MANUAL SELECT MODE *"
1332 AND TYPES "ENTER DRIVE NO. (0-7):".

1333
1334 THE OPERATOR TYPES THE DRIVE NUMBER, THE PROGRAM TYPES THE DRIVE
1335 SERIAL NUMBER, AND THEN ASKS "ALIGN, VERIFY, OR EXERCISE (A, V, OR E)
1336 ?". THE OPERATOR RESPONDS, AND THE PROGRAM ENTERS THE APPROPRIATE
1337 SUB-MODE.
1338

1339
1340
1341 11.2.1.1 MANUAL SELECT ALIGNMENT

1342 ALIGNMENT IN MANUAL MODE PROCEEDS AS FOLLOWS. THE PROGRAM TYPES:

1343 "* MANUAL SELECT ALIGNMENT *," FOLLOWED BY
1344 " ENTER HEAD NO. (0-2):"

1345
1346 THE OPERATOR TYPES THE DESIRED HEAD NUMBER, AND THE PROGRAM UNLOADS
1347 THE HEADS ON THE DRIVE, IN ANTICIPATION OF OPERATOR MOUNTING OF THE
1348 HEAD ALIGNMENT TOOL. THE PROGRAM TYPES "TYPE<R> WHEN READY:". AFTER
1349 THE TOOL HAS BEEN MOUNTED THE OPERATOR TYPES <R>, AND THE PROGRAM
1350 LOADS HEADS AND RESPONDS WITH:

1351 "HEADS POSITIONED AT CYLINDER 365(OCT)"
1352 AND "HEAD X SELECTED"

1353 THE POSITIONING TO CYLINDER 365 IS DONE IN SINGLE INCREMENT SEEKS, TO
1354 AVOID EXCESSIVE MOMENTUM ON THE POSITIONER, WHILE THE ALIGNMENT TOOL
1355 IS INSTALLED. THE OPERATOR MAY NOW PROCEED TO ALIGN THIS HEAD, AND
1356 THE PROGRAM LOOPS BACK TO ASK FOR ANOTHER HEAD NUMBER ON THIS DRIVE.
1357 IF THE OPERATOR TYPES CONTROL Z (↑Z) THE PROGRAM WILL LOOP BACK TO ASK
1358 FOR "ALIGN OR EXERCISE ?" (STILL IN THE SAME MODE). IF CONTROL C (↑C)
1359 IS TYPED, THE PROGRAM LOOPS FURTHER BACK TO ASK FOR A NEW DRIVE NUMBER
1360 (STILL IN THE SAME MODE). IF CONTROL R (↑R) IS TYPED, THE PROGRAM
1361 EXITS FROM THE CURRENT MODE, AND DOES A COMPLETE RESTART.
1362

1363
1364
1365 11.2.1.2 MANUAL SELECT VERIFY

1366 VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

1367 "* MANUAL SELECT VERIFY *" , FOLLOWED BY
1368 "ENTER HEAD NO. (0-2) :"

1369 THE VERIFY MODE IS IDENTICAL TO THE ALIGNMENT MODE, EXCEPT THAT
1370 UNLOADING AND LOADING OF HEADS IS NOT DONE BETWEEN HEAD SELECTIONS.
1371 THEREFORE, IN THIS MODE, THE OPERATOR MAY SELECT HEADS FOR QUICK
1372 ALIGNMENT VERIFICATION (USING THE ALIGNMENT METER), WITHOUT THE DELAYS
1373 OF UNLOADING AND LOADING HEADS. IN THIS MODE, THE OPERATOR WILL NOT
1374 BE ASKED TO "TYPE <R> WHEN READY".
1375
1376
1377
1378
1379
1380
1381

11.2.1.3 MANUAL SELECT EXERCISE

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

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

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

11.2.2 AUTO SELECT MODE

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

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

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

11.2.2.1 AUTO SELECT ALIGNMENT

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

" * AUTO SELECT ALIGNMENT * ".

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

THE PROGRAM RECOGNIZES THE DRIVE SELECTED, AND CHECKS IT TO BE SURE

1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493

THAT IT IS WRITE-LOCKED. THE PROGRAM THEN TYPES:

"DRIVE X SELECTED"

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

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

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

"HEADS POSITIONED AT CYLINDER 365(OCT)
HEAD 1 SELECTED".

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

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

NOTE

THE \uparrow C FUNCTION DOES NOT APPLY IN AUTO MODE.

11.2.2.2 AUTO SELECT VERIFY

VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

"* AUTO SELECT VERIFY *"

THE AUTO SELECT VERIFY MODE IS IDENTICAL TO THE AUTO ALIGNMENT MODE,

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

11.2.2.3 AUTO SELECT EXERCISE

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

AUTO SELECT EXERCISE MODE IDENTIFIES ITSELF AS FOLLOWS:

" * AUTO SELECT EXERCISES * "
FOLLO. -D BY "SHORT OR LONG (S OR L) ?".

THE OPERATOR TYPES HIS RESPONSE, AND THE SELECTED SEEK EXERCISES ARE PERFORMED ON EACH DRIVE. UPON COMPLETION OF THESE EXERCISES, THE PROGRAM RETURNS TO ASK FOR "ALIGN OR EXERCISE?" (STILL IN AUTO MODE). WHILE IN EXERCISE SUB-MODE THE CHARACTERS ↑Z AND ↑R PERFORM THE SAME FUNCTIONS AS DESCRIBED IN SECTION 11.2.1.1.

NOTE

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

11.3 ALIGNMENT AID ERROR MESSAGES

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

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

3. "PLEASE LOAD ALIGNMENT CARTRIDGE ON DRIVE X! PRESS 'CONT' ON

1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573

CPU WHEN READY." - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS NOT LOADED WITH AN ALIGNMENT CARTRIDGE.

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

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

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

APPENDIX A

SAMPLE ADDRESS 200 DEFAULT RUN

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

CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 2

LAST PHYS MEM ADR = 377776

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

DRIVE(S) = 0

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

OFFSET-TO-FAILURE MEASUREMENTS:

TRACK	CYLN	SECT	+OFST (UIN)	-OFST (UIN)
0	0	0	NONE	NONE
0	0	12	NONE	1175
0	631	0	NONE	NONE
0	631	12	NONE	NONE
1	0	0	1050	NONE
1	0	12	NONE	1200
1	631	0	NONE	NONE
1	631	12	NONE	NONE
2	0	0	NONE	NONE
2	0	12	NONE	NONE
2	631	0	NONE	NONE
2	631	12	NONE	NONE

END PASS # 1

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 34

SEQ 0033

APPENDIX B

SAMPLE ADDRESS 204 RUN

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

CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION:PART 2

LAST PHYS MEM ADR=377776

PARAMETER INPUT MODE

RK06-07 BUS ADR = 177440 NEW =
RK06-07 VEC ADR = 210 NEW =
RK06-07 PRIORITY = 5 NEW =

DRIVE(S) = 0

*

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

ENTER L,C, OR I

* C

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

*

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

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 35

SEQ 0034

1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740

5 2 * 0
6 100 * 1

ENTER L,C, OR I
* L

TEST	ITERATIONS
1	1
2	0
3	0
4	1
5	0
6	1

ENTER L,C, OR I
* I

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

ENTER T,O,S, OR R

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

ENTER T,O,S, OR R

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

MODIFY USER-DEFINED PATTERN 15:
WORD 00 = 072307 * 123456!
> tZ

ENTER T,O,S, OR R

* T
FC=0

1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796

LC=632
FT=0
LT=2
SO=0
SI=23
S2=4
S3=20
PT=100000
CS=0
ST=0

USER-DEFINED F ITERN 15:

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

ENTER T,O,S, OR R
* R

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

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

OFFSET-TO-FAILURE			MEASUREMENTS:	
TRACK	CYLN	SECT	+OFST (UIN)	-OFST (UIN)
0	0	4	NONE	NONE
00	0	20	NONE	NONE
00	631	4	NONE	NONE
00	631	20	NONE	NONE
1	0	4	NONE	NONE
1	0	20	NONE	NONE
1	631	4	NONE	NONE
1	631	20	NONE	NONE
2	0	4	NONE	NONE
2	0	20	NONE	NONE
2	631	4	NONE	NONE
2	631	20	NONE	NONE

K03

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 37

SEQ 0036

1797
1798
1799
1800
1801
1802
1803

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

APPENDIX C

SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING THE HEAD ALIGNMENT AID, PROVIDED IN SUBSYSTEM VERIFICATION PART 2 (ADDRESS 224 START). MANUAL MODE WAS REQUESTED, AND THE OPERATOR SELECTED DRIVE 0 FOR ALIGNMENT. IN THIS CASE, THE OPERATOR SELECTED HEADS 1, 0, AND 2, IN THAT ORDER, AND THEN REQUESTED RANDOM SEEK EXERCISES TO BE RUN ON THE DRIVE (AFTER REMOVAL OF THE HEAD ALIGNMENT TOOL). THE EXERCISES WOULD HAVE NORMALLY RUN FOR 5 MINUTES BUT THEY WERE PREMATURELY TERMINATED BY THE OPERATOR, BY TYPING (I2), IN THIS PARTICULAR RUN. FINALLY, VERIFY MODE WAS REQUESTED, AND HEAD 2 WAS SELECTED BY THE OPERATOR.

CZR6NDD - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 2

*** RK06-07 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A)?
M

* MANUAL SELECT MODE *

ENTER DRIVE NO. (0-7):

0
DRIVE SER. NO. 8

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

* MANUAL SELECT ALIGNMENT *

ENTER HEAD NO. (0-2):

1
TYPE <R> WHEN READY:

R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED

ENTER HEAD NO. (0-2):

1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 39

SEQ 0038

1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902

0
TYPE <R> WHEN READY:
R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED

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

ENTER HEAD NO. (0-2):
↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
E
TYPE <R> WHEN READY:
R

RANDOM SEEK EXERCISES IN PROGRESS ON DRIVE 0
↑Z

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

* MANUAL SELECT VERIFY *

ENTER HEAD NO. (0-2) :
2
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED

ENTER HEAD NO. (0-2) :
↑Z

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

1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958

APPENDIX D

SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

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

CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 2

*** RK06-07 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A)?
A

* AUTO SELECT MODE *

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

* AUTO SELECT ALIGNMENT *

DRIVE 0 SELECTED

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

B04

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 41

SEQ 0040

1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
↑Z

* AUTO SELECT MODE *

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

* AUTO SELECT EXERCISES *

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

* AUTO SELECT MODE *

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

* AUTO SELECT VERIFY *

DRIVE 0 SELECTED

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

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
0

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 42

SEQ 0041

000001

PART=1
;*** IF "PART" IS DEFINED = 0, PART 1 WILL BE ASSEMBLED. ***
;*** IF "PART" IS DEFINED NOT = 0, PART 2 WILL BE ASSEMBLED. ***
;*** REV 006 ***

167000

.NLIST MC,MD,CND
.LIST ME
ENABL ABS,AMA
\$SWR= 167000
;*****
SBTTL STARTING ADDRESSES
;
* 200 DEFAULT PARAMETERS FOR TESTS 1-6
* 204 SELECT PARAMETERS FOR TESTS 1-6
* 224 HEAD ALIGNMENT AID
;*****

000001

\$TN=1
*TITLE CZR6NDO RK611/06 SS VERIF 2
*COPYRIGHT (C) 1976,1977
*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.
*
SBTTL OPERATIONAL SWITCH SETTINGS
;
* SWITCH USE
*-----
* 15 HALT ON ERROR
* 14 LOOP ON TEST
* 13 INHIBIT ERROR TYPEOUTS
* 12 REPORT DESCRIPTION ONLY, ON ERRORS
* 11 INHIBIT ITERATIONS
* 10 BELL ON ERROR
* 9 LOOP ON ERROR
* 8 APPLY RANDOM STALL BETWEEN OPERATIONS
* 6 REPORT 1 ERROR PER TRANSFER IN TESTS 2-4
* 1 INHIBIT WRITES IN TEST 1
* 0 REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 2-4
SBTTL CONTROL SWITCH SETTINGS (PARAMETER CS)
;
* SWITCH USE
*-----
* 05 DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
* 04 TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
* 01 INHIBIT OFFSET REPORTS IN TEST 1
;
SBTTL BASIC DEFINITIONS
;
*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100

001100

```

2056      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
2057      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
2058
2059      ;*MISCELLANEOUS DEFINITIONS
2060      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
2061      000012      LF= 12      ;;CODE FOR LINE FEED
2062      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
2063      000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
2064      177776      PS= 177776  ;;PROCESSOR STATUS WORD
2065      .EQUIV PS,PSW
2066      177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
2067      177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
2068      177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER
2069      177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
2070
2071      ;*GENERAL PURPOSE REGISTER DEFINITIONS
2072      000000      R0= %0      ;;GENERAL REGISTER
2073      000001      R1= %1      ;;GENERAL REGISTER
2074      000002      R2= %2      ;;GENERAL REGISTER
2075      000003      R3= %3      ;;GENERAL REGISTER
2076      000004      R4= %4      ;;GENERAL REGISTER
2077      000005      R5= %5      ;;GENERAL REGISTER
2078      000006      R6= %6      ;;GENERAL REGISTER
2079      000007      R7= %7      ;;GENERAL REGISTER
2080      000006      SP= %6      ;;STACK POINTER
2081      000007      PC= %7      ;;PROGRAM COUNTER
2082
2083      ;*PRIORITY LEVEL DEFINITIONS
2084      000000      PR0= 0      ;;PRIORITY LEVEL 0
2085      000040      PR1= 40     ;;PRIORITY LEVEL 1
2086      000100      PR2= 100   ;;PRIORITY LEVEL 2
2087      000140      PR3= 140   ;;PRIORITY LEVEL 3
2088      000200      PR4= 200   ;;PRIORITY LEVEL 4
2089      000240      PR5= 240   ;;PRIORITY LEVEL 5
2090      000300      PR6= 300   ;;PRIORITY LEVEL 6
2091      000340      PR7= 340   ;;PRIORITY LEVEL 7
2092
2093      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
2094      100000      SW15= 100000
2095      040000      SW14= 40000
2096      020000      SW13= 20000
2097      010000      SW12= 10000
2098      004000      SW11= 4000
2099      002000      SW10= 2000
2100      001000      SW09= 1000
2101      000400      SW08= 400
2102      000200      SW07= 200
2103      000100      SW06= 100
2104      000040      SW05= 40
2105      000020      SW04= 20
2106      000010      SW03= 10
2107      000004      SW02= 4
2108      000002      SW01= 2
2109      000001      SW00= 1
2110      .EQUIV SW09,SW9
2111      .EQUIV SW08,SW8

```

```

2112      .EQUIV SW07,SW7
2113      .EQUIV SW06,SW6
2114      .EQUIV SW05,SW5
2115      .EQUIV SW04,SW4
2116      .EQUIV SW03,SW3
2117      .EQUIV SW02,SW2
2118      .EQUIV SW01,SW1
2119      .EQUIV SW00,SW0
2120
2121      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
2122      100000      BIT15= 100000
2123      040000      BIT14= 40000
2124      020000      BIT13= 20000
2125      010000      BIT12= 10000
2126      004000      BIT11= 4000
2127      002000      BIT10= 2000
2128      001000      BIT09= 1000
2129      000400      BIT08= 400
2130      000200      BIT07= 200
2131      000100      BIT06= 100
2132      000040      BIT05= 40
2133      000020      BIT04= 20
2134      000010      BIT03= 10
2135      000004      BIT02= 4
2136      000002      BIT01= 2
2137      000001      BIT00= 1
2138      .EQUIV BIT09,BIT9
2139      .EQUIV BIT08,BIT8
2140      .EQUIV BIT07,BIT7
2141      .EQUIV BIT06,BIT6
2142      .EQUIV BIT05,BIT5
2143      .EQUIV BIT04,BIT4
2144      .EQUIV BIT03,BIT3
2145      .EQUIV BIT02,BIT2
2146      .EQUIV BIT01,BIT1
2147      .EQUIV BIT00,BIT0
2148
2149      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
2150      000004      ERRVEC= 4      ; TIME OUT AND OTHER ERRORS
2151      000010      RESVEC= 10     ; RESERVED AND ILLEGAL INSTRUCTIONS
2152      000014      TBITVEC=14     ; "T" BIT
2153      000014      TRTVEC= 14     ; TRACE TRAP
2154      000014      BPTVEC= 14     ; BREAKPOINT TRAP (BPT)
2155      000020      IOTVEC= 20     ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
2156      000024      PWRVEC= 24     ; POWER FAIL
2157      000030      EMTVEC= 30     ; EMULATOR TRAP (EMT) **ERROR**
2158      000034      TRAPVEC=34     ; "TRAP" TRAP
2159      000060      TKVEC= 60      ; TTY KEYBOARD VECTOR
2160      000064      TPVEC= 64     ; TTY PRINTER VECTOR
2161      000240      PIRQVEC=240    ; PROGRAM INTERRUPT REQUEST VECTOR
2162      .SBTTL MEMORY MANAGEMENT DEFINITIONS
2163
2164      ;*KT11 VECTOR ADDRESS
2165
2166      000250      MMVEC= 250
2167

```

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 45
MEMORY MANAGEMENT DEFINITIONS

SEQ 0044

```

2168
2169
2170      177572      SRO=      177572
2171      177574      SR1=      177574
2172      177576      SR2=      177576
2173      172516      SR3=      172516
2174
2175      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
2176
2177      172300      KIPDR0= 172300
2178      172302      KIPDR1= 172302
2179      172304      KIPDR2= 172304
2180      172306      KIPDR3= 172306
2181      172310      KIPDR4= 172310
2182      172312      KIPDR5= 172312
2183      172314      KIPDR6= 172314
2184      172316      KIPDR7= 172316
2185
2186      ;*KERNEL "I" PAGE ADDRESS REGISTERS
2187
2188      172340      KIPAR0= 172340
2189      172342      KIPAR1= 172342
2190      172344      KIPAR2= 172344
2191      172346      KIPAR3= 172346
2192      172350      KIPAR4= 172350
2193      172352      KIPAR5= 172352
2194      172354      KIPAR6= 172354
2195      172356      KIPAR7= 172356
2196
2197      170200      MAPL00=170200
2198      170202      MAPH00=170202
2199      172100      MEMCSR=172100      ; MEMORY CSR REG START ADRS
2200      177740      LOERAD=177740      ; 11/70 MEM LO ERROR ADRS REG
2201      177742      HIERAD=177742      ; 11/70 MEM HI ERROR ADRS REG
2202      177744      MEMSYS=177744      ; 11/70 MEM SYSTEM REG
2203      .SBTTL TRAP CATCHER
2204
2205      000000      .=0
2206      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
2207      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
2208      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
2209      .=174
2210      000174      000000      DISREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
2211      000176      000000      SWREG:  .WORD 0      ;; SOFTWARE SWITCH REGISTER
2212
2213      000200      000137      013342      .SBTTL STARTING ADDRESS(ES)
2214      .=170      JMP      @#DFSTRT      ;; JUMP TO STARTING ADDRESS OF PROGRAM
2215      000170      000000      KILLDR: .WORD 0      ; IF NOT = 0, IT IS OK TO OVERLAY LOADER
2216      000204      000204      .=204
2217      000204      000137      013376      JMP      @#PSTART
2218      000224      000224      .=224
2219      000224      000137      024134      JMP      @#ASTART
2220      000230      000700      ..LOW:  .WORD 700
2221      000232      001100      ..HIGH: .WORD 1100
2222      .SBTTL ACT11 HOOKS
2223

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 46
ACT11 HOOKS

SEQ 0045

```

2224      ;*****
2225      ;HOOKS REQUIRED BY ACT11
2226      $SVPC=      ;SAVE PC
2227      =46
2228      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
2229      =52
2230      .WORD 140000      ;;2)SET LOC.52 TO 140000
2231      = $SVPC      ;; RESTORE PC
2232      =1000
2233      .SBTTL APT PARAMETER BLOCK
2234      ;*****
2235      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
2236      ;*****
2237      ;*****
2238      $X=      ;SAVE CURRENT LOCATION
2239      =24      ;SET POWER FAIL TO POINT TO START OF PROGRAM
2240      200      ;FOR APT START UP
2241      =44      ;POINT TO APT INDIRECT ADDRESS PNTR.
2242      $APTHDR ;POINT TO APT HEADER BLOCK
2243      = $X      ;RESET LOCATION COUNTER
2244      ;*****
2245      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
2246      ;INTERFACE SPEC.
2247
2248      $APTHD:
2249      $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
2250      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
2251      $TSTM: .WORD 1130      ;;RUN TIM OF LONGEST TEST
2252      $PASTM: .WORD 3410      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
2253      $UNITM: .WORD 3410      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
2254      .WORD $ETEND-$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)
2255      AVECT1=120210      ;RKVEC=210, RKPRI=5
2256      ABASE=177440      ;RKBAS ADR$
2257      ADEVM=000377      ;SET DEVICES 0-7 IN MAP

```

2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313

001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000
001162 000000
001164 000000
001166 000000
001170 000000
001172 000000
001174 000000
001176 000000
001200 000000
001202 000000
001204 000000
001206 000000
001210 000000
001212 000000
001214 000000
001216 000000
001220 000000
001222 000000

.SBTTL COMMON TAGS

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

SCMTAG: . =1100

.WORD 0
\$TSTNM: .BYTE 00
\$ERFLG: .BYTE 00
\$ICNT: .WORD 00
\$LPADR: .WORD 00
\$LPERR: .WORD 00
\$ERTTL: .WORD 00
\$ITEMB: .BYTE 00
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 00
\$GDADR: .WORD 00
\$BDADR: .WORD 00
\$GDDAT: .WORD 00
\$BDDAT: .WORD 00
\$AUTOB: .BYTE 00
\$INTAG: .BYTE 00
\$SWR: .WORD 0 DSWR
\$DISPLAY: .WORD 0 DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0
\$REG0: .WORD 0
\$REG1: .WORD 0
\$REG2: .WORD 00
\$REG3: .WORD 00
\$REG4: .WORD 00
\$REG5: .WORD 00
\$REG6: .WORD 00
\$REG7: .WORD 00
\$REG10: .WORD 00
\$REG11: .WORD 00
\$REG12: .WORD 00
\$REG13: .WORD 00
\$REG14: .WORD 00
\$REG15: .WORD 00
\$REG16: .WORD 00
\$REG17: .WORD 00
\$REG20: .WORD 0

; ; START OF COMMON TAGS

;; CONTAINS THE TEST NUMBER
;; CONTAINS ERROR FLAG
;; CONTAINS SUBTEST ITERATION COUNT
;; CONTAINS SCOPE LOOP ADDRESS
;; CONTAINS SCOPE RETURN FOR ERRORS
;; CONTAINS TOTAL ERRORS DETECTED
;; CONTAINS ITEM CONTROL BYTE
;; CONTAINS MAX. ERRORS PER TEST
;; CONTAINS PC OF LAST ERROR INSTRUCTION
;; CONTAINS ADDRESS OF 'GOOD' DATA
;; CONTAINS ADDRESS OF 'BAD' DATA
;; CONTAINS 'GOOD' DATA
;; CONTAINS 'BAD' DATA
;; RESERVED--NOT TO BE USED
;; AUTOMATIC MODE INDICATOR
;; INTERRUPT MODE INDICATOR
;; ADDRESS OF SWITCH REGISTER
;; ADDRESS OF DISPLAY REGISTER
;; TTY KBD STATUS
;; TTY KBD BUFFER
;; TTY PRINTER STATUS REG. ADDRESS
;; TTY PRINTER BUFFER REG. ADDRESS
;; CONTAINS NULL CHARACTER FOR FILLS
;; CONTAINS # OF FILLER CHARACTERS REQUIRED
;; INSERT FILL CHARS. AFTER A "LINE FEED"
;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
;; CONTAINS THE ADDRESS FROM WHICH (\$REG0) WAS OBTAINED
;; CONTAINS ((\$REGAD)+0)
;; CONTAINS ((\$REGAD)+2)
;; CONTAINS ((\$REGAD)+4)
;; CONTAINS ((\$REGAD)+6)
;; CONTAINS ((\$REGAD)+10)
;; CONTAINS ((\$REGAD)+12)
;; CONTAINS ((\$REGAD)+14)
;; CONTAINS ((\$REGAD)+16)
;; CONTAINS ((\$REGAD)+20)
;; CONTAINS ((\$REGAD)+22)
;; CONTAINS ((\$REGAD)+24)
;; CONTAINS ((\$REGAD)+26)
;; CONTAINS ((\$REGAD)+30)
;; CONTAINS ((\$REGAD)+32)
;; CONTAINS ((\$REGAD)+34)
;; CONTAINS ((\$REGAD)+36)
;; CONTAINS ((\$REGAD)+40)

2314	001224	000000	\$REG21: .WORD	0	:: CONTAINS ((SREGAD)+42)
2315	001226	000000	\$REG22: .WORD	0	:: CONTAINS ((SREGAD)+44)
2316	001230	000000	\$REG23: .WORD	0	:: CONTAINS ((SREGAD)+46)
2317	001232	000000	\$REG24: .WORD	0	:: CONTAINS ((SREGAD)+50)
2318	001234	000000	\$REG25: .WORD	0	:: CONTAINS ((SREGAD)+52)
2319	001236	000000	\$REG26: .WORD	0	:: CONTAINS ((SREGAD)+54)
2320	001240	000000	\$REG27: .WORD	0	:: CONTAINS ((SREGAD)+56)
2321	001242	000000	\$REG30: .WORD	0	:: CONTAINS ((SREGAD)+60)
2322	001244	000000	\$REG31: .WORD	0	:: CONTAINS ((SREGAD)+62)
2323	001246	000000	\$REG32: .WORD	0	:: CONTAINS ((SREGAD)+64)
2324	001250	000000	\$REG33: .WORD	0	:: CONTAINS ((SREGAD)+66)
2325	001252	000000	\$REG34: .WORD	0	:: CONTAINS ((SREGAD)+70)
2326	001254	000000	\$REG35: .WORD	0	:: CONTAINS ((SREGAD)+72)
2327	001256	000000	\$REG36: .WORD	0	:: CONTAINS ((SREGAD)+74)
2328	001260	000000	\$REG37: .WORD	0	:: CONTAINS ((SREGAD)+76)
2329	001262	000000	\$TMP0: .WORD	0	:: USER DEFINED
2330	001264	000000	\$TMP1: .WORD	0	:: USER DEFINED
2331	001266	000000	\$TMP2: .WORD	0	:: USER DEFINED
2332	001270	000000	\$TMP3: .WORD	0	:: USER DEFINED
2333	001272	000000	\$TMP4: .WORD	0	:: USER DEFINED
2334	001274	000000	\$TMP5: .WORD	0	:: USER DEFINED
2335	001276	000000	\$TMP6: .WORD	0	:: USER DEFINED
2336	001300	000000	\$TMP7: .WORD	0	:: USER DEFINED
2337	001302	000000	\$TMP10: .WORD	0	:: USER DEFINED
2338	001304	000000	\$TIMES: 0		:: MAX. NUMBER OF ITERATIONS
2339	001306	000000	\$ESCAPE: 0		:: ESCAPE ON ERROR ADDRESS
2340	001310	177607	\$BELL: .ASCIZ	<207><377><377>	:: CODE FOR BELL
2341	001314	077	\$QUES: .ASCII	/?	:: QUESTION MARK
2342	001315	015	\$CRLF: .ASCII	<15>	:: CARRIAGE RETURN
2343	001316	000012	\$LF: .ASCIZ	<12>	:: LINE FEED
2344			:*****		
2345			.SBTTL APT MAILBOX-ETABLE		
2346			:*****		
2347			.EVEN		
2348			\$MAIL: .WORD		
2349	001320	000000	\$MSGTY: .WORD	AMSGTY	:: APT MAILBOX
2350	001320	000000	\$FATAL: .WORD	AFATAL	:: MESSAGE TYPE CODE
2351	001322	000000	\$TESTN: .WORD	ATESTN	:: FATAL ERROR NUMBER
2352	001324	000000	\$PASS: .WORD	APASS	:: TEST NUMBER
2353	001326	000000	\$DEVCT: .WORD	ADEVCT	:: PASS COUNT
2354	001330	000000	\$UNIT: .WORD	AUNIT	:: DEVICE COUNT
2355	001332	000000	\$MSGAD: .WORD	AMSGAD	:: I/O UNIT NUMBER
2356	001334	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE ADDRESS
2357	001336	000000	\$ETABLE: .WORD	ACPUOP	:: MESSAGE LENGTH
2358	001340	000	\$ENV: .BYTE	AENV	:: APT ENVIRONMENT TABLE
2359	001340	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT BYTE
2360	001341	000	\$SWREG: .WORD	ASWREG	:: ENVIRONMENT MODE BITS
2361	001342	000000	\$USWR: .WORD	AUSWR	:: APT SWITCH REGISTER
2362	001344	000000	\$CPUOP: .WORD	ACPUOP	:: USER SWITCHES
2363	001346	000000			:: CPU TYPE, OPTIONS
2364					BIT 15-11=CPU TYPE
2365					11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
2366					11/70=06, PDQ=07, Q=10
2367					BIT 10=REAL TIME CLOCK
2368					BIT 9=FLOATING POINT PROCESSOR
2369					BIT 8=MEMORY MANAGEMENT

J04

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 49
APT MAILBOX-ETABLE

SEQ 0048

2370	001350	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
2371	001351	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE,BLK#1
2372			::*		MEM.TYPE,BYTE -- (HIGH BYTE)
2373			::*		900 NSEC CORE=001
2374			::*		300 NSEC BIPOLAR=002
2375			::*		500 NSEC MOS=003
2376	001352	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS,BLK#1
2377			::*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2378	001354	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
2379	001355	000	\$MTYP2: .BYTE	AMTYP2	::MEM. TYPE,BLK#2
2380	001356	000000	\$MADR2: .WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
2381	001360	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
2382	001361	000	\$MTYP3: .BYTE	AMTYP3	::MEM. TYPE,BLK#3
2383	001362	000000	\$MADR3: .WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
2384	001364	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
2385	001365	000	\$MTYP4: .BYTE	AMTYP4	::MEM. TYPE,BLK#4
2386	001366	000000	\$MADR4: .WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
2387	001370	120210	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
2388	001372	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2,BUS PRIORITY#2
2389	001374	177440	\$BASE: .WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
2390	001376	000377	\$DEV1: .WORD	ADEV1	::DEVICE MAP
2391	001400		\$ETEND:		
2392			.MEXIT		

K04

.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

2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407 001400
2408
2409
2410 001400 060375
2411 001402 063072
2412 001404 064610
2413 001406 064724
2414
2415
2416 001410 060414
2417 001412 063072
2418 001414 064610
2419 001416 064724
2420
2421
2422 001420 060432
2423 001422 063072
2424 001424 064610
2425 001426 064724
2426
2427
2428 001430 060450
2429 001432 063072
2430 001434 064610
2431 001436 064724
2432
2433
2434 001440 060467
2435 001442 063072
2436 001444 064610
2437 001446 064754
2438
2439
2440 001450 060506
2441 001452 063072
2442 001454 064610
2443 001456 065010
2444
2445
2446 001460 060525
2447 001462 063072
2448 001464 064610

CZR6N00 RK611/06 SS VERIF 2
CZR6N0.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 51
ERROR POINTER TABLE

SEQ 0050

2449	001466	065010	DF03	
2450				
2451			:ERROR 10	
2452	001470	060544	EM10	;AC LOW
2453	001472	063072	DH100	
2454	001474	064610	DT100	
2455	001476	064754	DF02	
2456				
2457			:ERROR 11	
2458	001500	060553	EM11	;SPEED LOSS
2459	001502	063072	DH100	
2460	001504	064610	DT100	
2461	001506	064754	DF02	
2462				
2463			:ERROR 12	
2464	001510	060566	EM12	;ILLEGAL FUNCTION
2465	001512	063072	DH100	
2466	001514	064610	DT100	
2467	001516	064754	DF02	
2468				
2469			:ERROR 13	
2470	001520	060600	EM13	;PROGRAMMING ERROR
2471	001522	063072	DH100	
2472	001524	064610	DT100	
2473	001526	064724	DF01	
2474				
2475			:ERROR 14	;NON-EXISTANT FUNCTION
2476	001530	060611	EM14	
2477	001532	063072	DH100	
2478	001534	064610	DT100	
2479	001536	064754	DF02	
2480				
2481			:ERROR 15	
2482	001540	060631	EM15	;DRIVE TYPE ERROR
2483	001542	063072	DH100	
2484	001544	064610	DT100	
2485	001546	064754	DF02	
2486				
2487			:ERROR 16	
2488	001550	060645	EM16	;FORMAT ERROR
2489	001552	063072	DH100	
2490	001554	064610	DT100	
2491	001556	064754	DF02	
2492				
2493			:ERROR 17	
2494	001560	060656	EM17	;WRITE LOCK ERROR
2495	001562	063072	DH100	
2496	001564	064610	DT100	
2497	001566	064754	DF02	
2498				
2499			:ERROR 20	
2500	001570	060673	EM20	;DRIVE UNSAFE
2501	001572	063072	DH100	
2502	001574	064610	DT100	
2503	001576	064754	DF02	
2504				

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 52
ERROR POINTER TABLE

M04

SEQ 0051

2505			:ERROR 21	
2506	001600	060706	EM21	;SEEK INCOMPLETE
2507	001602	063072	DH100	
2508	001604	064610	DT100	
2509	001606	064754	DF02	
2510				
2511			:ERROR 22	
2512	001610	060722	EM22	;CYLINDER OVERFLOW
2513	001612	063072	DH100	
2514	001614	064610	DT100	
2515	001616	064754	DF02	
2516				
2517			:ERROR 23	
2518	001620	060735	EM23	;ILLEGAL CYLINDER
2519	001622	063072	DH100	
2520	001624	064610	DT100	
2521	001626	064754	DF02	
2522				
2523			:ERROR 24	
2524	001630	060752	EM24	;DRIVE OFF TRACK
2525	001632	063072	DH100	
2526	001634	064610	DT100	
2527	001636	064754	DF02	
2528				
2529			:ERROR 25	
2530	001640	060770	EM25	;DRIVE TIMING ERROR
2531	001642	063072	DH100	
2532	001644	064610	DT100	
2533	001646	064754	DF02	
2534				
2535			:ERROR 26	
2536	001650	061007	EM26	;DATA LATE
2537	001652	063072	DH100	
2538	001654	064610	DT100	
2539	001656	064754	DF02	
2540				
2541			:ERROR 27	
2542	001660	061021	EM27	;CONTROLLER TIMEOUT
2543	001662	063072	DH100	
2544	001664	064610	DT100	
2545	001666	064754	DF02	
2546				
2547			:ERROR 30	
2548	001670	061037	EM30	;OPERATION INCOMPLETE
2549	001672	063072	DH100	
2550	001674	064610	DT100	
2551	001676	065064	DF05	
2552				
2553			:ERROR 31	
2554	001700	061060	EM31	;HEADER VRC ERROR
2555	001702	063072	DH100	
2556	001704	064610	DT100	
2557	001706	065064	DF05	
2558				
2559			:ERROR 32	
2560	001710	061077	EM32	;DATA CHECK ERROR

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 53
ERROR POINTER TABLE

SEQ 0052

2561	001712	063072	DH100	
2562	001714	064610	DT100	
2563	001716	065120	DF07	
2564				
2565			:ERROR 33	
2566	001720	061116	EM33	;WRITE CHECK ERROR
2567	001722	063072	DH100	
2568	001724	064610	DT100	
2569	001726	065154	DF10	
2570				
2571			:ERROR 34	
2572	001730	061132	EM34	;DATA MISCOMPARE(S)
2573	001732	063072	DH100	
2574	001734	064610	DT100	
2575	001736	065050	DF04	
2576				
2577			:ERROR 35	
2578	001740	061152	EM35	;NO DRIVE RESPONSE-UFE & NXD
2579	001742	063072	DH100	
2580	001744	064610	DT100	
2581	001746	064724	DF01	
2582				
2583			:ERROR 36	
2584	001750	061204	EM36	;DRIVE ERROR WILL NOT CLEAR
2585	001752	000000	0	
2586	001754	000000	0	
2587	001756	000000	0	
2588				
2589			:ERROR 37	
2590	001760	061233	EM37	;DRIVE STATUS CHANGE WILL NOT CLEAR
2591	001762	000000	0	
2592	001764	000000	0	
2593	001766	000000	0	
2594				
2595			:ERROR 40	
2596	001770	061274	EM40	;ATTENTION BUT NO STATUS CHANGE OR FAULT
2597	001772	063072	DH100	
2598	001774	064610	DT100	
2599	001776	064754	DF02	
2600				
2601			:ERROR 41	
2602	002000	061340	EM41	;ATTENTION BUT DRIVE NOT AVAILABLE
2603	002002	063072	DH100	
2604	002004	064610	DT100	
2605	002006	064754	DF02	
2606				
2607			:ERROR 42	
2608	002010	061370	EM42	;ATTENTION WHEN NOT EXPECTED
2609	002012	063072	DH100	
2610	002014	064610	DT100	
2611	002016	064754	DF02	
2612				
2613			:ERROR 43	
2614	002020	061420	EM43	;ERROR WHILE GATHERING DRIVE STATUS
2615	002022	063072	DH100	
2616	002024	064610	DT100	

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78
ERROR POINTER TABLE

12:02 PAGE 54

SEQ 0053

2617	002026	065220	DF12	
2618				
2619			:ERROR 44	
2620	002030	061631	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2621	002032	063072	DH100	
2622	002034	064610	DT100	
2623	002036	065220	DF12	
2624				
2625			:ERROR 45	
2626	002040	061667	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2627	002042	063072	DH100	
2628	002044	064610	DT100	
2629	002046	065220	DF12	
2630				
2631			:ERROR 46	
2632	002050	061715	EM65	;UNSOLICITED ATTENTION
2633	002052	063072	DH100	
2634	002054	064610	DT100	
2635	002056	065220	DF12	
2636				
2637			:ERROR 47	
2638	002060	061737	EM66	;UNEXPECTED DATA TYPE ERROR
2639	002062	063072	DH100	
2640	002064	064610	DT100	
2641	002066	065220	DF12	
2642				
2643			:ERROR 50	
2644	002070	061763	EM67	;ATTENTION DID NOT RESET WITH CLEAR
2645	002072	063072	DH100	
2646	002074	064610	DT100	
2647	002076	065220	DF12	
2648				
2649			:ERROR 51	
2650	002100	062022	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2651	002102	063072	DH100	
2652	002104	064610	DT100	
2653	002106	065220	DF12	
2654				
2655			:ERROR 52	
2656	002110	061451	EM52	;MULTIPLE DRIVE SELECT
2657	002112	063072	DH100	
2658	002114	064610	DT100	
2659	002116	065220	DF12	
2660				
2661			:ERROR 53	
2662	002120	061466	EM53	;ABREVIATED HCE ERROR
2663	002122	063072	DH100	
2664	002124	064610	DT100	
2665	002126	065250	DF13	
2666				
2667			:ERROR 54	
2668	002130	061037	EM30	;OPERATION INCOMPLETE ERROR
2669	002132	063072	DH100	
2670	002134	064610	DT100	
2671	002136	065300	DF14	
2672				

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 55
ERROR POINTER TABLE

SEQ 0054

2673			:ERROR 55	
2674	002140	061060	EM31	;ABREVIATED HVRC ERROR
2675	002142	063072	DH100	
2676	002144	064610	DT100	
2677	002146	065250	DF13	
2678				
2679			:ERROR 56	
2680	002150	061511	EM56	;2 TIMEOUT ERROR
2681	002152	063072	DH100	
2682	002154	064610	DT100	
2683	002156	065330	DF15	
2684				
2685			:ERROR 57	;2ND LEVEL IN SUBSYSTEM TIMEOUT
2686	002160	061511	EM56	
2687	002162	063072	DH100	
2688	002164	064610	DT100	
2689	002166	065374	DF16	
2690				
2691			:ERROR 60	
2692	002170	061530	EM60	;ERROR IN RECAL FOR RECOVERY
2693	002172	000000	0	
2694	002174	000000	0	
2695	002176	000000	0	
2696				
2697			:ERROR 61	
2698	002200	061562	EM61	;ABORT MESSAGE
2699	002202	000000	0	
2700	002204	000000	0	
2701	002206	000000	0	
2702				
2703			:ERROR 62	
2704	002210	061615	EM62	;CYLINDER MISCOMPARE
2705	002212	063072	DH100	
2706	002214	064610	DT100	
2707	002216	065440	DF17	
2708				
2709			:ERROR 63	;DATA ERROR WORDS
2710	002220	000000	0	
2711	002222	000000	0	
2712	002224	064702	DT602	
2713	002226	065550	DF25	
2714				
2715			:ERROR 64	
2716	002230	061631	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2717	002232	063072	DH100	
2718	002234	064610	DT100	
2719	002236	064754	DF02	
2720				
2721			:ERROR 65	
2722	002240	061667	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2723	002242	063072	DH100	
2724	002244	064610	DT100	
2725	002246	064754	DF02	
2726				
2727			:ERROR 66	
2728	002250	061715	EM65	;UNSOLICITED ATTENTION

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 56
ERROR POINTER TABLE

SEQ 0055

2729	002252	063072	DH100	
2730	002254	064610	DT100	
2731	002256	064754	DF02	
2732				
2733			:ERROR 67	
2734	002260	061737	EM66	;UNEXPECTED DATA TYPE ERROR
2735	002262	063072	DH100	
2736	002264	064610	DT100	
2737	002266	064754	DF02	
2738				
2739			:ERROR 70	
2740	002270	061763	EM67	;ATTENTION DID NOT RESET WITH CLEAR
2741	002272	063072	DH100	
2742	002274	064610	DT100	
2743	002276	064754	DF02	
2744				
2745			:ERROR 71	
2746	002300	062022	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR ATT
2747	002302	063072	DH100	
2748	002304	064610	DT100	
2749	002306	064754	DF02	
2750				
2751			:ERROR 72	
2752	002310	062067	EM71	;DATA LATE WHEN UNLOADING HEADER
2753	002312	063072	DH100	
2754	002314	064610	DT100	
2755	002316	064754	DF02	
2756				
2757			:ERROR 73	
2758	002320	062127	EM72	;CONTROLLER ERROR DURING DRIVER SERVICE
2759	002322	063072	DH100	
2760	002324	064610	DT100	
2761	002326	064754	DF02	
2762				
2763			:ERROR 74	
2764	002330	062162	EM73	;DRIVE DETECTED PARITY ERROR
2765	002332	063072	DH100	
2766	002334	064610	DT100	
2767	002336	064754	DF02	
2768				
2769			:ERROR 75	
2770	002340	062202	EM74	;UNDEFINED ERROR
2771	002342	063072	DH100	
2772	002344	064610	DT100	
2773	002346	064754	DF02	
2774				
2775			:ERROR 76	
2776	002350	062214	EM75	;MARKING SECTOR BAD MESSAGE
2777	002352	000000	0	
2778	002354	000000	0	
2779	002356	000000	0	
2780				
2781			:ERROR 77	
2782	002360	062244	EM76	;BAD DATA VERIFICATION WITH READ
2783	002362	064010	DH605	
2784	002364	064674	DT601	

E05

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78
ERROR FOINTER TABLE

12:02 PAGE 57

SEQ 0056

2785	002366	065474	DF21	
2786				
2787			:ERROR 100	
2788	002370	062326	EM77	;RETRY SUCCESSFUL MESSAGE
2789	002372	000000	0	
2790	002374	064674	DT601	
2791	002376	065534	DF23	
2792				
2793			:ERROR 101	
2794	002400	062326	EM77	;ANOTHER RETRY SUCCESSFUL MESSAGE
2795	002402	000000	0	
2796	002404	064674	DT601	
2797	002406	065534	DF23	
2798				
2799			:ERROR 102	
2800	002410	062347	EM100	;RETRY UNSUCCESSFUL MESSAGE
2801	002412	000000	0	
2802	002414	064674	DT601	
2803	002416	065534	DF23	
2804				
2805			:ERROR 103	
2806	002420	062372	EM101	;NO VALID HEADERS IN TRACK JUST READ
2807	002422	063772	DM6042	
2808	002424	064674	DT601	
2809	002426	065544	DF24	
2810				
2811			:ERROR 104	
2812	002430	062450	EM102	;BSE ERROR ON SECTOR NOT LISTED AS BAD
2813	002432	063072	DM100	
2814	002434	064610	DT100	
2815	002436	064754	DF02	
2816				
2817			:ERROR 105	
2818	002440	062514	EM103	;TIMED-OUT ON READ HEADER
2819	002442	063072	DM100	
2820	002444	064610	DT100	
2821	002446	065010	DF03	
2822				
2823			:ERROR 106	
2824	002450	062542	EM104	;TIMED-OUT ON SEEK
2825	002452	063072	DM100	
2826	002454	064610	DT100	
2827	002456	065010	DF03	
2828				
2829			:ERROR 107	
2830	002460	062564	EM105	;DRIVE SIEZED BY OTHER PORT
2831	002462	063072	DM100	
2832	002464	064610	DT100	
2833	002466	064754	DF02	
2834				
2835			:ERROR 110	
2836	002470	062615	EM106	; "DATA MISCMPR WHILE BAI SET"
2837	002472	063072	DM100	
2838	002474	064610	DT100	
2839	002476	065564	DF27	
2840				

2841			.ERROR 111	
2842	002500	062650	EM107	; "NO NEM WHEN EXPECTED"
2843	002502	000000	0	
2844	002504	000000	0	
2845	002506	000000	0	
2846				
2847			.ERROR 112	
2848	002510	062713	EM110	; "INTRPT WHEN CNTRLR NOT READY"
2849	002512	063072	DH100	
2850	002514	064610	DT100	
2851	002516	064724	DF01	
2852				
2853			.ERROR 113	
2854	002520	062746	EM111	; "NO ATT'N ON SEEK"
2855	002522	063072	DH100	
2856	002524	064610	DT100	
2857	002526	064754	DF02	
2858				
2859			.ERROR 114	
2860	002530	062767	EM112	; DRIVE'S CYLINDER INCORRECT
2861	002532	064217	DH202	
2862	002534	064650	DT202	
2863	002536	065554	DF26	
2864				
2865			.ERROR 115	
2866	002540	000000	0	; TYPE ADRS OF DATA MISCOMPARE(S)
2867	002542	000000	0	
2868	002544	064674	DT601	
2869	002546	065200	DF11	
2870				
2871			.ERROR 116	
2872	002550	061132	EM34	; DATA MISCOMPARE (11/70)
2873	002552	063350	DH103	
2874	002554	064720	DT103	
2875	002556	065464	DF20	
2876				
2877			.ERROR 117	
2878	002560	000000	0	; PART OF DATA MISCOMPARE
2879	002562	000000	0	
2880	002564	064610	DT100	
2881	002566	065504	DF22	
2882				
2883			.ERROR 120	
2884	002570	063013	EM113	; ABORT- CAN'T READ BSF
2885	002572	063072	DH100	
2886	002574	064610	DT100	
2887	002576	064724	DF01	
2888				
2889			.ERROR 121	
2890	002600	063041	EM114	; KT11 FAILURE
2891	002602	063072	DH100	
2892	002604	064610	DT100	
2893	002606	065610	DF30	
2894				
2895			.ERROR 122	
2896	002610	063056	EM115	; MEM PARITY ERROR

G05

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78
ERROR POINTER TABLE

12:02 PAGE 59

SEQ 0058

2897 002612 063072
2898 002614 064610
2899 002616 065634
2900
2901 002620 060432
2902 002622 000000
2903 002624 000000
2904 002626 000000
2905
2906
2907
2908
2909
2910

DH100
DT100
DF31
:ERROR 123
EM3
0
0
0

;NED IN SIZING UNDER APT

H05

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 60
BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0059

.SBTTL BIT ASSIGNMENTS IN THE RK611 REGISTERS

RKCS1							
15	14	13	12	11	10	9	8
CERR	DI	DCPAR	CFMT	CTO	CDT	BA17	BA16
CCLR							
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	NDS	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							

2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964

2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015

```

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!

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 62
BIT ASSIGNMENTS IN THE RK611 REGISTERS

J05

SEQ 0061

3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054

```

: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 ! WRT ! ECCW ! PCD ! PCA ! MEWD ! MERD ! MCLK !
: GATE ! GATE ! READ ONLY ! READ/WRITE !
:-----
: 7 6 5 4 3 2 1 0
:-----
: MIND ! MSP ! DMD ! PAT ! MS3 ! MS2 ! MS1 ! MS0 !
: READ/WRITE
:-----

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 63
BIT ASSIGNMENTS IN THE RK611 REGISTERS

K05

SEQ 0062

3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092

```
;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 !
-----
```


CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 64
BIT ASSIGNMENTS IN THE RK611 REGISTERS

L05

SEQ 0063

3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130

LINE B MESSAGE 00							
15	14	13	12	11	10	9	8
PAR ! UNS ! DROT ! DCLO ! WLE ! SKI ! PERR ! ILF !							

7	6	5	4	3	2	1	0
FLT ! ACLO ! IVDA ! UN ! UN ! UN ! 0 ! 0 !							

LINE A MESSAGE 01							
15	14	13	12	11	10	9	8
PAR ! UNLDG ! RTZ ! LDG ! REV ! FWD ! SPEED ! CART !							
HDS ! HDS ! OK ! PRES !							

7	6	5	4	3	2	1	0
DOOR ! BRUSH ! HEADS ! TRK ! UN ! DRIVE SELECT CODE !							
LTCH ! HOME ! HOME ! FOLW ! 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 !							

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 65
BIT ASSIGNMENTS IN THE RK611 REGISTERS

M05

SEQ 0064

3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186

```
LINE A MESSAGE 10
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 10
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 11
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 11
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
RKDA= 6 ; DESIRED TRACK SECTOR REGISTER
RKCS2= 10 ; CONTROL AND STATUS REGISTER 2

N05

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 66
RK06 CONTROLLER REGISTER DEFINITION

SEQ 0065

3187	000012	RKDS= 12	; DRIVE STATUS REGISTER
3188	000014	RKER= 14	; ERROR REGISTER
3189	000016	RKASOF= 16	; ATTENTION SUMMARY AND OFFSET REGISTER
3190	000020	RKDC= 20	; DESIRED CYLINDER REGISTER
3191	000020	RKDCYL= 20	; DESIRED CYLINDER REGISTER
3192	000024	RKDB= 24	; DATA BUFFER
3193	000026	RKMR1= 26	; MAINTENANCE REGISTER 1
3194	000034	RKMR2= 34	; MAINTENANCE REGISTER 2
3195	000036	RKMR3= 36	; MAINTENANCE REGISTER 3
3196	000030	RKPOS= 30	; ECC POSITION INFORMATION
3197	000030	RKECPS= 30	; ECC POSITION INFORMATION
3198	000032	RKPAT= 32	; ECC PATTERN INFORMATION
3199	000032	RKECPT= 32	; ECC PATTERN INFORMATION
3200			
3201		.SBTTL DRIVE COMMANDS	
3202			
3203	000101	SELDRV= 101	; SELECT DRIVE
3204	000103	PACK= 103	; PACK ACKNOWLEDGE
3205	000105	CLEAR= 105	; DRIVE CLEAR
3206	000107	UNLOAD= 107	; UNLOAD
3207	000111	SRTSPL= 111	; START SPINDLE
3208	000113	RECAL= 113	; RECALIBRATE
3209	000115	OFFSET= 115	; OFFSET
3210	000117	SEEK= 117	; SEEK
3211	000121	RDDATA= 121	; READ DATA
3212	000123	WRDATA= 123	; WRITE DATA
3213	000125	RDHEAD= 125	; READ HEADER
3214	000127	WRHEAD= 127	; WRITE HEADER AND DATA
3215	000131	WRTCHK= 131	; WRITE CHECK
3216			
3217		; THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER	
3218		; TO SIMULATE A SPECIFIC DESIRED OPERATION	
3219			
3220	000140	RELEAS= 140	; RELEASE DRIVE
3221	000141	RDSTAT= 141	; GET ALL STATUS FROM DRIVE
3222	000164	RDALHD= 164	; READ ALL HEADERS
3223	000176	CONCLR= 176	; CONTROLLER CLEAR (BIT 15 OF CS1)
3224	000177	SUBCLR= 177	; SUBSYSTEM CLEAR (BIT 5 OF CS2)
3225	000300	INTR= 300	; GENERATE INTERRUPT TO CPU
3226			
3227		; DRIVER ISSUED SERVICE COMMANDS	
3228			
3229	000001	DR.SEL= 001	; DRIVE SELECT
3230	000005	DR.CLR= 005	; DRIVE CLEAR
3231			
3232		.SBTTL CONTROL AND STATUS REGISTER 1 BITS	
3233			
3234	000001	GO= BIT0	; GO BIT
3235	000100	IE= BIT6	; INTERRUPT ENABLE
3236	000200	RDY= BIT7	; CONTROLLER READY
3237	000400	BA16= BIT8	; BUS ADDRESS BIT 16
3238	001000	BA17= BIT9	; BUS ADDRESS BIT 17
3239	002000	CDT= BIT10	; CONTROLLER DRIVE TYPE (0=RK06, 1=RK07)
3240	004000	CTO= BIT11	; CONTROLLER TIMED OUT WAITING FOR
3241			; DRIVE RESPONSE
3242	010000	CFMT= BIT12	; CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

3243	020000	SPAR=	BIT13	;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
3244	040000	DI=	BIT14	;DRIVE INTERRUPT
3245	100000	CERR=	BIT15	;CONTROLLER ERROR
3246	100000	CCLR=	BIT15	;CONTROLLER CLEAR
3247				
3248				
3249				
3250				
3251	000001	B.BA16=	BIT0	;BUS ADDRESS BIT 16
3252	000002	B.BA17=	BIT1	;BUS ADDRESS BIT 17
3253	000004	B.CDT=	BIT2	;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
3254	000020	B.CFMT=	BIT4	;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
3255				
3256				
3257				
3258	000007	.SBTTL	CONTROL AND STATUS REGISTER 2 BITS	
3259	000010	DRVMSK=	7	;MASK FOR DRIVE SELECTION CODE
3260	000010	DESL=	BIT3	;DESELECT OR RELEASE DRIVE IN BITS 0-2
3261	000020	RLS=	BIT3	;DESELECT OR RELEASE DRIVE IN BITS 0-2
3262	000040	BAI=	BIT4	;BUS ADDRESS INCREMENT INHIBIT
3263	000040	CLR=	BIT5	;CLEAR CONTROLLER AND ALL DRIVES
3264	000100	SCLR=	BIT5	;CLEAR CONTROLLER AND ALL DRIVES
3265	000200	IR=	BIT6	;INPUT READY
3266	000400	OR=	BIT7	;OUTPUT READY
3267	001000	UFE=	BIT8	;UNIT FIELD ERROR
3268	002000	MDS=	BIT9	;MULTIPLE DRIVE SELECT
3269	004000	PGE=	BIT10	;PROGRAMMING ERROR
3270	010000	NEM=	BIT11	;NON-EXISTENT MEMORY
3271	020000	NED=	BIT12	;NON-EXISTENT DRIVE
3272	040000	UPE=	BIT13	;UNIBUS PARITY ERROR
3273	100000	WCE=	BIT14	;WRITE CHECK ERROR
3274		DLT=	BIT15	;DATA LATE ERROR
3275				
3276				
3277	000001	.SBTTL	ERROR REGISTER BIT DEFINITION	
3278		ILC=	BIT0	;ILLEGAL FUNCTION CODE
3279	000002	*ILF=	BIT0	;ILLEGAL FUNCTION CODE
3280	000004	SKI=	BIT1	;SEEK INCOMPLETE
3281	000004	ILF=	BIT2	;ILLEGAL DRIVE FUNCTION
3282	000010	NXF=	BIT2	;ILLEGAL DRIVE FUNCTION
3283	000020	DRPAR=	BIT3	;DRIVE DETECTED DRIVE BUS PARITY ERROR
3284	000040	FMTE=	BIT4	;FORMAT ERROR
3285	000100	DTYPE=	BIT5	;DRIVE TYPE ERROR
3286	000200	ECH=	BIT6	;ECC HARD
3287	000400	BSE=	BIT7	;BAD SECTOR ERROR
3288	000400	HCRC=	BIT8	;HEADER CRC ERROR
3289	001000	HVRC=	BIT8	;HEADER VRC ERROR
3290	002000	COE=	BIT9	;CYLINDER ADDRESS OVERFLOW ERROR
3291	004000	IDAE=	BIT10	;INVALID DISK ADDRESS ERROR
3292	010000	WLE=	BIT11	;WRITE LOCK ERROR
3293	020000	DTE=	BIT12	;DRIVE TIMING ERROR
3294	040000	OPI=	BIT13	;OPERATION (SEARCH) INCOMPLETE
3295	100000	UNS=	BIT14	;DRIVE UNSAFE
3296		DCK=	BIT15	;DATA CHECK
3297				
3298		.SBTTL	STATUS REGISTER BIT DEFINITION	

3299	000001	DRA=	BIT0	;DRIVE AVAILABLE (CONTROLLER IS SET IF
3300				; THIS BIT IS RESET)
3301	000004	OFST=	BIT2	;DRIVE OFFSET
3302	000010	ACLO=	BIT3	;AC LOW
3303	000020	SPDLSS=	BIT4	;SPEED LOSS
3304	000020	DCLO=	BIT4	;DC LOW
3305	000040	DROT=	BIT5	;DRIVE OFF TRACK
3306	000100	VV=	BIT6	;VOLUME VALID
3307	000200	DRY=	BIT7	;DRIVE READY
3308	000200	DRDY=	BIT7	;DRIVE READY
3309	000400	DDT=	BIT8	;DRIVE TYPE (0=RK06, 1=RK07)
3310	004000	WRL=	BIT11	;WRITE LOCK
3311	020000	PIP=	BIT13	;POSITIONING IN PROGRESS
3312	040000	DSC=	BIT14	;DRIVE STATUS CHANGE
3313	100000	SVAL=	BIT15	;STATUS VALID
3314				
3315		.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION		
3316				
3317	000017	MESMSK=	17	;MESSAGE MASK
3318				
3319	000020	PAT=	BIT4	;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
3320	000040	DMD=	BIT5	;DIAGNOSTIC MODE
3321	000100	MSP=	BIT6	;MAINTENANCE SECTOR PULSE
3322	000200	MIND=	BIT7	;MAINTENANCE INDEX
3323	000400	MCLK=	BIT8	;MAINTENANCE CLOCK
3324	001000	MERD=	BIT9	;MAINTENANCE ENCODED READ DATA
3325	002000	MEWD=	BIT10	;MAINTENANCE ENCODED WRITE DATA
3326	004000	PCA=	BIT11	;PRECOMPENSATION ADVANCE
3327	010000	PCD=	BIT12	;PRECOMPENSATION DELAY
3328	020000	ECCW=	BIT13	;ECC WORD IS BEING READ OR WRITTEN
3329	040000	WRTGAT=	BIT14	;WRITE GATE
3330	100000	RDGATE=	BIT15	;READ GATE
3331				
3332		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A		
3333				
3334	000040	S.DRA=	BIT5	;DRIVE AVAILIABLE
3335	000100	S.VV=	BIT6	;VOLUME VALID
3336	000200	S.DRY=	BIT7	;DRIVE READY
3337	000400	S.TYPE=	BIT8	;DRIVE TYPE
3338	001000	S.FORM=	BIT9	;DRIVE FORMAT
3339	002000	S.OFF=	BIT10	;OFFSET
3340	004000	S.WRL=	BIT11	;WRITE LOCK
3341	010000	S.SPIN=	BIT12	;SPINDLE ON
3342	020000	S.PIP=	BIT13	;POSITIONING IN PROGRESS
3343	040000	S.DSC=	BIT14	;DRIVE STATUS CHANGE
3344				
3345		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B		
3346				
3347	000040	S.ICYL=	BIT5	;ILLEGAL CYLINDER ADDRESS
3348	000100	S.ACLO=	BIT6	;AC LOW
3349	000200	S.FLT=	BIT7	;DRIVE FAULT
3350	000400	S.ILF=	BIT8	;ILLEGAL FUNCTION
3351	001000	S.PAR=	BIT9	;DRIVE DETECTED DRIVE BUS PARITY ERROR
3352	002000	S.SKI=	BIT10	;SEEK INCOMPLETE
3353	004000	S.WLE=	BIT11	;WRITE LOCK ERROR
3354	010000	S.SPLS=	BIT12	;SPEED LOSS

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 69
DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

SEQ 0068

3355	010000	S.DCLO= BIT12	;DC LOW
3356	020000	S.DROT= BIT13	;DRIVE OFF TRACK
3357	040000	S.UNS= BIT14	;DRIVE UNSAFE
3358			
3359		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3360			
3361	000020	S.XDOK= BIT4	;TRANSDUCER OK
3362	000040	S.HOHM= BIT5	;HEADS HOME
3363	000100	S.BRHM= BIT6	;BRUSHES HOME
3364	000200	S.DOOR= BIT7	;DOOR INTERLOCKED
3365	000400	S.CART= BIT8	;CARTRAGE INTERLOCK
3366	001000	S.SPOK= BIT9	;SPEED OK
3367	002000	S.FWD= BIT10	;FORWARD
3368	004000	S.REV= BIT11	;REVERSE
3369	010000	S.LOAD= BIT12	;HEADS LOADING
3370	020000	S.RTZ= BIT13	;RETURN TO ZERO
3371	040000	S.UNLD= BIT14	;HEADS UNLOADING
3372			
3373		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3374			
3375	000020	S.SECT= BIT4	;SECTOR ERROR
3376	000040	S.WCLK= BIT5	;WRITE CLOCK AND NO WRITE GATE
3377	000100	S.WGAT= BIT6	;WRITE GATE AND NO TRANSISTIONS
3378	000200	S.HDFL= BIT7	;HEAD FAULT
3379	000400	S.MHD= BIT8	;MULTIPLE HEAD SELECT
3380	001000	S.XERR= BIT9	;INDEX ERROR
3381	002000	S.DIB= BIT10	;DIBIT ERROR
3382	004000	S.PLO= BIT11	;PLO ERROR
3383	010000	S.NMOV= BIT12	;SEEK AND NO MOTION
3384	020000	S.LIMD= BIT13	;LIMIT DETECT ON SEEK
3385	040000	S.BRKE= BIT14	;SERVO-BRAKE
3386			
3387		.SBTTL	COMMON MASKS
3388			
3389	000007	M.DRV= 7	;DRIVE CODE
3390	100000	M.PAR= BIT15	;PARITY
3391	000003	M.ID= 3	;BYTE ID
3392	017760	M.CDIF= 17760	;CYLINDER DIFFERENCE/OFFSET
3393	017760	M.CADD= 17760	;CYLINDER ADDRESS
3394	077770	M.SER= 77770	;DRIVE SERIAL NUMBER
3395	000760	M.SECT= 760	;SECTOR COUNT
3396	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
*****

```

.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

```

3453      000100      CMDT0= BIT6      ; NO TERMINATION TO COMMAND FOR AT
3454      ; LEAST 1 SECOND
3455      000200      W.WCK= BIT7      ; WRITE FOR WRITE WRITE CHECK
3456      000400      NOCHK= BIT8      ; NO CHECK, DO NOT SET INTERRUPT ENABLE
3457      001000      PBSVAL= BIT9      ; PARAMETER STATUS WORDS VALID
3458      ; (SET WHEN ERROR TERMINATION OR
3459      ; READ STATUS COMMAND)
3460      002000      DRPDRV= BIT10     ; DROP DRIVE FROM TEST SEQUENCE
3461      004000      NODSC= BIT11     ; ATTENTION SET BUT DCS AND FAULT RESET
3462      010000      DRVSZD= BIT12     ; DRIVE SEIZED BY OTHER PORT
3463      020000      E.UNLD= BIT13     ; DRIVE UNLOADED DUE TO ERROR
3464      040000      Q.INIT= BIT14     ; PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
3465      100000      DTBAII= BIT15     ; INHIBIT BUS ADDRESS INCREMENT
3466
3467      .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
3468
3469      ; THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETUF ,
3470      ; FROM THE DRIVER TO THE CALLING PROGRAM
3471
3472      000016      P.CS1= 16      ; COMMAND AND STATUS REGISTER 1
3473      000020      P.CS2= 20      ; COMMAND AND STATUS REGISTER 2
3474      000022      P.WCR= 22      ; WORD COUNT REGISTER
3475      000024      P.BAR= 24      ; BUS ADDRESS REGISTER
3476      000026      P.OTS= 26      ; DESIRED TRACK SECTOR REGISTER
3477      000030      P.DCYL= 30      ; DESIRED CYLINDER REGISTER
3478      000032      P.ASOF= 32      ; ATTENTION SUMMARY/OFFSET REGISTER
3479      000034      P.ER= 34      ; ERROR REGISTER
3480      000036      P.OS= 36      ; STATUS REGISTER
3481      000040      P.A00= 40      ; MESSAGE A STATUS BYTE 00
3482      000042      P.B00= 42      ; MESSAGE B STATUS BYTE 00
3483      000044      P.A01= 44      ; MESSAGE A STATUS BYTE 01
3484      000046      P.B01= 46      ; MESSAGE B STATUS BYTE 01
3485      000050      P.A10= 50      ; MESSAGE A STATUS BYTE 10
3486      000052      P.B10= 52      ; MESSAGE B STATUS BYTE 10
3487      000054      P.A11= 54      ; MESSAGE A STATUS BYTE 11
3488      000056      P.B11= 56      ; MESSAGE B STATUS BYTE 11
3489      000060      P.EPOS= 60      ; ECC POSITION INFORMATION
3490      000062      P.EPAT= 62      ; ECC PATTERN INFORMATION
3491
3492      .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
3493
3494      002630      000      PARMO: .BYTE 0      ; DRIVE NUMBER
3495      002631      000      .BYTE 0      ; COMMAND
3496      002632      000000   .WORD 0      ; CYLINDER ADDRESS
3497      002634      000      .BYTE 0      ; SECTOR ADDRESS
3498      002635      000      .BYTE 0      ; TRACK ADDRESS
3499      002636      000      .BYTE 0      ; OFFSET VALUE
3500      002637      000      .BYTE 0      ; BUS ADDRESS (BITS 16 AND 17)
3501      002640      000000   .WORD 0      ; BUS ADDRESS (BITS 0 - 15)
3502      002642      000000   .WORD 0      ; WORD COUNT (2'S COMPLEMENT)
3503      002644      000000   .WORD 0      ; PROGRAM DRIVE STATUS INFORMATION
3504      002646      000000   .WORD 0      ; COMMAND AND STATUS REGISTER 1
3505      002650      000000   .WORD 0      ; COMMAND AND STATUS REGISTER 2
3506      002652      000000   .WORD 0      ; WORD COUNT REGISTER
3507      002654      000000   .WORD 0      ; BUS ADDRESS REGISTER
3508      002656      000000   .WORD 0      ; DESIRED TRACK AND SECTOR REGISTER

```


3509	002660	000000	.WORD	0	; DESIRED CYLINDER REGISTER
3510	002662	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
3511	002664	000000	.WORD	0	; ERROR REGISTER
3512	002666	000000	.WORD	0	; STATUS REGISTER
3513	002670	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
3514	002672	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
3515	002674	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
3516	002676	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
3517	002700	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
3518	002702	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
3519	002704	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
3520	002706	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
3521	002710	000000	.WORD	0	; ECC POSITION INFORMATION
3522	002712	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL PARAMETER BLOCK 1 FOR DRIVE

3526	002714	000	PARM1: .BYTE	0	; DRIVE NUMBER
3527	002715	000	.BYTE	0	; COMMAND
3528	002716	000000	.WORD	0	; CYLINDER ADDRESS
3529	002720	000	.BYTE	0	; SECTOR ADDRESS
3530	002721	000	.BYTE	0	; TRACK ADDRESS
3531	002722	000	.BYTE	0	; OFFSET VALUE
3532	002723	000	.BYTE	0	; BUS ADDRESS (BITS 16 AND 17)
3533	002724	000000	.WORD	0	; BUS ADDRESS (BITS 0 - 15)
3534	002726	000000	.WORD	0	; WORD COUNT (2'S COMPLEMENT)
3535	002730	000000	.WORD	0	; PROGRAM DRIVE STATUS INFORMATION
3536	002732	000000	.WORD	0	; COMMAND AND STATUS REGISTER 1
3537	002734	000000	.WORD	0	; COMMAND AND STATUS REGISTER 2
3538	002736	000000	.WORD	0	; WORD COUNT REGISTER
3539	002740	000000	.WORD	0	; BUS ADDRESS REGISTER
3540	002742	000000	.WORD	0	; DESIRED TRACK AND SECTOR REGISTER
3541	002744	000000	.WORD	0	; DESIRED CYLINDER REGISTER
3542	002746	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
3543	002750	000000	.WORD	0	; ERROR REGISTER
3544	002752	000000	.WORD	0	; STATUS REGISTER
3545	002754	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
3546	002756	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
3547	002760	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
3548	002762	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
3549	002764	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
3550	002766	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
3551	002770	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
3552	002772	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
3553	002774	000000	.WORD	0	; ECC POSITION INFORMATION
3554	002776	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL TEMPORARY CONTROLLER REGISTER STORAGE

3558	003000	000000	T.CS1: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 1
3559					
3560	003002	000000	T.CS2: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 2
3561					
3562	003004	000000	T.WCR: .WORD	0	; TEMPORARY STORAGE FOR WORD COUNT REGISTER
3563	003006	000000	T.BA: .WORD	0	; TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
3564	003010	000000	T.DA: .WORD	0	; TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

3565	003012	000000	T.DC: .WORD	0	TEMPORARY STORAGE FOR DRIVE CYLINDER
3566	003014	000000	T.ASOF: .WORD	0	TEMPORARY STORAGE FOR ATTENTION SUMMARY
3567					AND OFFSET
3568	003016	000000	T.ER: .WORD	0	TEMPORARY STORAGE FOR ERROR REGISTER
3569	003020	000000	T.DS: .WORD	0	TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3570	003022	000000	T.MR1: .WORD	0	TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3571	003024	000000	T.MR2: .WORD	0	TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3572	003026	000000	T.MR3: .WORD	0	TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3573	003030	000000	T.POS: .WORD	0	TEMPORARY STORAGE FOR ECC POSITION
3574	003032	000000	T.PAT: .WORD	0	TEMPORARY STORAGE FOR ECC PATTERN
3575	003034	000000	T.DB: .WORD	0	TEMPORARY STORAGE FOR DATA BUFFER REGISTER
3576					
3577			.SB*TL DRIVER PARAMERTERS		
3578					
3579	003036	177440	RKBAS: .WORD	177440	ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3580	003040	000210	RKVEC: .WORD	210	ADDRESS OF R611 VECTOR
3581	003042	000240	RKPRI: .WORD	PRS	RK611 INTERRUPT PRIORITY
3582	003044	042732	A.NORM: ERRFRE		ADDRESS OF NORMAL RETURN FROM DRIVER
3583	003046	044200	A.ABNL: ERRHDL		ADDRESS OF ABNORMAL RETURN FROM DRIVER
3584	003050	043474	A.CONT: CONERR		ADDRESS OF CONTROLLER ERROR RETURN
3585	003052	000000	E.CONT: .WORD	0	CONTROLLER ERROR STATUS
3586					THIS LOCATION IS CLEARED WHEN EVERY COMMAND
3587					IS INITIATED. IF A CONTROLLER ERROR
3588					OCCURS THE FOLLOWING BIT ASSIGNMENT IS
3589					USED:
3590					
3591		000001	E.CCLR= BIT0		CLEAR CONTROLLER DID NOT CLEAR ERROR
3592		000002	E.NOAT= BIT1		NO ATTENTION IN ATTENTION SUMMARY REG
3593		000004	E.UATT= BIT2		UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3594		000010	E.UDAT= BIT3		UNEXPECTED DATA TYPE ERROR
3595		000020	E.CLAT= BIT4		ATTENTION DID NOT RESET WITH CLEAR
3596		000040	E.SCLR= BIT5		SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
3597					ATTENTION
3598		000100	E.ILLD= BIT6		ILLEGAL DRIVER COMMAND
3599		000400	E.DLT= BIT8		DATA LATE WHEN UNLOADING HEADER
3600		001000	E.CERR= BIT9		CONTROLLER ERROR DURING DRIVER SERVICING
3601		002000	E.OPAR= BIT10		DRIVE DETECTED PARITY ERROR
3602		040000	E.CMT0= BIT14		CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3603		100000	E.MDS= BIT15		MULTIPLE DRIVE SELECT
3604					
3605	003054	000000	O.WAIT: .WORD	0	PARAMETER BLOCK OF THE DRIVE
3606					WAITING FOR COMMAND COMPLETION
3607	003056	000400	W.MTIM: .WORD	400	LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
3608	003060	000400	W.MILI: .WORD	400	16 MILLISECOND TIME FOR PROGRAM
3609					
3610					
3611					
3612					
3613					
3614					
3615					
3616					
3617					
3618					
3619					
3620					

CPU	VALUE
---	-----
11/05	100
11/10	
11/20	
11/34	
11/40	
11/45	400
11/50	
11/70	

3621	003062	000300			W.SEC: .WORD	300	; SECOND COUNT COUNT FOR ALL COMMANDS
3622							; EXCEPT START SPINDLE
3623	003064	003000			W.8SEC: .WORD	3000	; 8 SECOND FOR DRIVE CYCLE DOWN
3624	003066	030000			W.MIN: .WORD	30000	; MINUTE TIME FOR START SPINDLE
3625	003070	000000			HDR.AD: .WORD	0	; ADDRESS USED FOR READ ALL HEADERS
3626	003072	000000			HDR.CT: .WORD	0	; NUMBER OF HEADERS LEFT TO READ FOR READ
3627							; ALL HEADERS
3628	003074	000			I.ISRL: .BYTE	0	; INTERRUPT OR RELEASED COMMAND ISSUED
3629	003075	002	004	010	H.HEAD: .BYTE	2,4,10	; HEAD DECODES
3630	003100	000			W.TIME: .BYTE	0	; DRIVES BEING WATCH-DOG TIMED
3631							
3632					.SBTTL	INTERRUPT MASKS	
3633							
3634	003101	000			INTMSK: .BYTE	0	; INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
3635							
3636						INTERRUPT MASK TABLE	
3637							
3638	003102	001			I.DRV: .BYTE	1	; INTERRUPT MASK FOR DRIVE 0
3639	003103	002			.BYTE	2	; INTERRUPT MASK FOR DRIVE 1
3640	003104	004			.BYTE	4	; INTERRUPT MASK FOR DRIVE 2
3641	003105	010			.BYTE	10	; INTERRUPT MASK FOR DRIVE 3
3642	003106	020			.BYTE	20	; INTERRUPT MASK FOR DRIVE 4
3643	003107	040			.BYTE	40	; INTERRUPT MASK FOR DRIVE 5
3644	003110	100			.BYTE	100	; INTERRUPT MASK FOR DRIVE 6
3645	003111	200			.BYTE	200	; INTERRUPT MASK FOR DRIVE 7
3646							
3647					.SBTTL	PARAMETER BLOCK TABLE	
3648							
3649	003112	002630			PBLKT: PARM0		; ADDRESS OF PARAMETER BLOCK GIVEN WITH
3650							; DRIVE CALL. MUST BE LOADED INTO PBLKT
3651							
3652							
3653					.SBTTL	TIME FOR WATCH-DOG TIMER	
3654							
3655	003114	000000			W.DRV: .WORD	0	; TIME FOR INSTRUCTION IN PARAMETER BLOCK
3656					.SBTTL	PROGRAM SPECIFIC RESERVED LOCATIONS	
3657	003116	000			MDFLAG: .BYTE	0	; FLAG TO INDIC. DEFLT OR PARAM MODE
3658	003117	000			XXDPCH: .BYTE	0	; XXDP CHAIN MODE FLAG
3659	003120	000			TSTING: .BYTE	0	; CURRENTLY RUNNING TESTS IF = 1
3660	003121	000			DERCNT: .BYTE	0	; DATA ERROR COUNT
3661	003122	000			OPCOMP: .BYTE	0	; OPERATION COMPLETE FLAG
3662	003123	000			DONE: .BYTE	0	; DONE SWITCH
3663	003124	000			TYPFMT: .BYTE	0	; DRIVE TYPE & FORMAT CONTROL
3664	003125	000			FORMAT: .BYTE	0	; DRIVE FORMAT IN BIT 4 OF BYTE
3665	003126	000			ERRCNT: .BYTE	0	; ERROR COUNT
3666	003127	004			ERRLMT: .BYTE	4	; ERROR LIMIT
3667	003130	000			DRVERS: .BYTE	0	; ERROR COUNT FOR CURRENT DRIVE
3668	003131	000			OPCONT: .BYTE	0	; OPERATION CONTROL SWITCHES
3669	003132	000			PCLKF: .BYTE	0	; IF BYTE=1, KW11-P CLOCK IS PRESENT
3670	003133	000			XOVLAD: .BYTE	0	; FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
3671	003134	000			XDPSVD: .BYTE	0	; FLAG = 1 IF XXDP IS SAVED
3672	003135	000			DULACS: .BYTE	0	; FLAG=1 IF DUAL ACCESS TEST
3673	003136	000			ORNAFG: .BYTE	0	; =1 INDICATES DRIVE SIEZED BY OTHER PORT
3674	003137	000			REISSU: .BYTE	0	; DUAL-ACC FLAG TO RE-ISSUE COMMAND
3675	003140	000			WCEFLG: .BYTE	0	; WRITE CHECK ERROR FLAG
3676	003141	000			DLTFLG: .BYTE	0	; DATA LATE ERROR FLAG

J06

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 75
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0074

3677	003142	000	NORTRY: .BYTE	0	;"NO-RETRY" FLAG
3678	003143	000	UBMPRS: .BYTE	0	;UNIBUS MAP PRESENT IF = 1
3679	003144	000	MEMABT: .BYTE	0	;SET BYTE = 1 FOR NO ABORT
3680					; ON MEMORY PARITY ERRORS
3681	003145	000	HLPOVL: .BYTE	0	;HELP FILE OVERLaid INDICATOR
3682	003146	000	NOTYPE: .BYTE	0	;INDICATES PROGRAM JUST LOADED IF 0
3683		000001	WHDSW=BIT0		;WRITE HEADER & DATA SWITCH
3684		000002	VFHDSW=BIT1		;VERIFY HEADERS SWITCH
3685		000004	WCDASW=BIT2		;WRITE CHECK DATA SWITCH
3686		000010	RCDASW=BIT3		;READ CHECK DATA SWITCH
3687		000020	OREQSW=BIT4		;OFFSET REQUIRED SWITCH
3688		003150			
3689	003150	000000	AUTOFG: .EVEN	0	;SPECIAL FLAG
3690					
3691	003152	177546	LKS: .WORD	177546	;KW11-L CLOCK STATUS REGISTER
3692	003154	172540	PKS: .WORD	172540	;KW11-P CONTROL AND STATUS REGISTER
3693	003156	172542	PKSB: .WORD	172542	;KW11-P COUNT SET BUFFER REGISTER
3694	003160	172544	PKRB: .WORD	172544	;KW11-P COUNTER REGISTER
3695	003162	000100	LCVEC: .WORD	100	;KW11-L VECTOR STORAGE
3696	003164	000104	PCVEC: .WORD	104	;KW11-P VECTOR STORAGE
3697		000114	MEMVEC=114		;MEMORY PARITY TRAP VECTOR

3698	003166	000000			TCONLO: .WORD	0	;LO BITS OF CALIBRATION TIME CONSTANT
3699	003170	000000			TCONHI: .WORD	0	;HI BITS OF CALIB TIME CONST
3700	003172	000000			SUMLO1: .WORD	0	;LO BITS OF FORWARD TIME SUM
3701	003174	000000			SUMHI1: .WORD	0	;HI BITS OF FORWARD TIME SUM
3702	003176	000000			SAVPAR: .WORD	0	;SAVE WORD FOR PAR CONSTANT
3703	003200	000000			SAVWRD: .WORD	0	;SCRATCH WORD
3704	003202	000000			CYL: .WORD	0	;STARTING CYL FOR MULTI-DRV TEST
3705	003204	000010			CYLLST: .BLKW	↑08	;LIST OF PSEUDO-RAND CYL ADDRESSES
3706	003224	000400			BSSOFT: .BLKW	↑0256	;RECORD OF BAD SECTORS FROM SOFTWARE
3707	004224	000400			BSFACT: .BLKW	↑0256	;RECORD OF BAD SECTORS FROM FACTORY
3708	005224	000000	000000	000000	PRVCMO: .WORD	0,0,0,0,0,0	;PREVIOUS COMMAND STORAGE
3709	005232	000000	000000	000000			
3710	005240	000000	000000	000000	COMSTR: .WORD	0,0,0,0,0,0	;CURRENT COMMAND STORAGE
3711	005246	000000	000000	000000			
3712	005254	000000			PRMPLO: .WORD	0	;PREV. U.B. MAP REG 0
3713	005256	000000			PRMPHO: .WORD	0	
3714	005260	000000			CRMPLO: .WORD	0	;CURRENT U.B. MAP REG 0
3715	005262	000000			CRMPHO: .WORD	0	
3716	005264	000102			BUFFO: .BLKW	↑066	;OUTPUT BUFFER 1
3717	005470	000000			LOWOCT: .WORD	0	;LOW 16 BITS OF CONVERTED BINARY NUMBER
3718	005472	000000			HIGOCT: .WORD	0	;HIGH BITS OF CONVERTED BINARY NO.
3719	005474	000000			RECODE: .WORD	0	;RECOVERY CODE WORD
3720	005476	000000			ERRCOM: .WORD	0	;ERROR COMMAND
3721	005500	000000			DRIVE: .WORD	0	;NO. OF DRIVE IN USE
3722	005502	000000			STALLS: .WORD	0	;CURRENT NO. OF UNIT STALLS TO APPLY
3723	005504	000000			CYLNRD: .WORD	0	;CURRENT CYLINDER NUMBER
3724	005506	000000			FS: .WORD	0	;FIRST SECTOR LIMIT
3725	005510	000000			LS: .WORD	0	;LAST SECTOR LIMIT
3726	005512	000000			NCYL1: .WORD	0	;NEXT CYL SCRATCH WORD
3727	005514	000000			NCYL2: .WORD	0	;NEXT CYL SCRATCH WORD
3728	005516	000000			OFINUS: .WORD	0	;OFFSET IN USE
3729	005520	000000			TRACK: .WORD	0	;TRACK IN USE
3730	005522	000000			INTCHR: .WORD	0	;TTY INTERRUPT INPUT WORD
3731	005524	000000			SELECT: .WORD	0	;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3732	005526	000000			ONLINE: .WORD	0	;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3733	005530	000000			NEWON: .WORD	0	;NEW LOOK AT ONLINE DRIVES
3734	005532	000000			HDALI: .WORD	0	;TAG USED FOR INTERNAL USE
3735	005534	000000			SCRACH: .WORD	0	;ALL-PURPOSE SCRATCH WORD
3736	005536	000000			PATRN: .WORD	0	;DATA PATTERN LIST WORD
3737	005540	000000			XXDPAD: .WORD	0	;STARTING ADDRESS OF XXDP LOADER
3738	005542	000002			XDPSAV: .BLKW	2	;XXDP PHYSICAL SAVE ADDRESS
3739	005546	000000			PLOFST: .WORD	0	;(+) OFFSET VALUE
3740	005550	000000			NGOFST: .WORD	0	;(-) OFFSET VALUE
3741	005552	000005			SAVPRS: .BLKW	5	;SAVE INITIAL PARAMS FOR XFER
3742	005564	000000			WDSXFR: .WORD	0	;NO. OF WORDS ACTUALLY XFERRED
3743	005566	000000			FINCYL: .WORD	0	;FINAL CYLINDER ADRS
3744	005570	000			FINTRK: .BYTE	0	;FINAL TRACK ADRS
3745	005571	000			FINSEC: .BYTE	0	;FINAL SECTOR ADRS
3746	005572	000000			LASTWC: .WORD	0	;ACTUAL FINAL WC
3747	005574	157776			MAHILM: .WORD	157776	;MA UPPER LIMIT
3748	005576	000077				77	
3749	005600	065660			MA: .WORD	RWBUF	;LO BITS OF PHYS MEM ADRS
3750	005602	000000				0	;HI BITS OF PHYS MEM ADRS
3751	005604	000002			PMA: .BLKW	2	;PARTIAL TRANSFER MEMORY START ADRS
3752							
3753							

3754
3755
3756 005610
3757 005610 001
3758 005611 001
3759 005612 001
3760 005613 001
3761 005614 001
3762 005615 001
3763 005616 001
3764 005617 001
3765
3766 005620
3767 005620 000
3768 005621 000
3769 005622 000
3770 005623 000
3771 005624 000
3772 005625 000
3773 005626 000
3774 005627 000
3775
3776
3777
3778
3779
3780
3781 005630
3782 005630 000002
3783 005632 000002
3784 005634 000002
3785 005636 000002
3786 005640 000002
3787 005642 000100
3788
3789
3790
3791
3792 005644
3793 005644 000002
3794 005646 000002
3795 005650 000002
3796 005652 000002
3797 005654 000002
3798 005656 000100
3799
3800
3801 000006
3802
3803
3804
3805 005660
3806 005660 000000
3807 005662 000000
3808 005664 000000
3809 005666 000002

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

.BYTE	1	;DRIVE	0
.BYTE	1	;DRIVE	1
.BYTE	1	;DRIVE	2
.BYTE	1	;DRIVE	3
.BYTE	1	;DRIVE	4
.BYTE	1	;DRIVE	5
.BYTE	1	;DRIVE	6
.BYTE	1	;DRIVE	7

;LIST OF DRIVE TYPES, 0=RK06, NON 0=RK07
DTYLIST:

.BYTE	0	;DRV	0
.BYTE	00	;DRV	1
.BYTE	00	;DRV	2
.BYTE	00	;DRV	3
.BYTE	00	;DRV	4
.BYTE	00	;DRV	5
.BYTE	00	;DRV	6
.BYTE	0	;DRV	7

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

.WORD	2	;TEST	1
.WORD	2	;TEST	2
.WORD	2	;TEST	3
.WORD	2	;TEST	4
.WORD	2	;TEST	5
.WORD	100	;TEST	6

;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
DFLTST:

.WORD	2	;TEST	1
.WORD	2	;TEST	2
.WORD	2	;TEST	3
.WORD	2	;TEST	4
.WORD	2	;TEST	5
.WORD	100	;TEST	6

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

;OPERATING PARAMETER LIST
PRMLST:

FC:	.WORD	0	;FIRST CYLINDER
LC:	.WORD	0	;LAST CYLINDER
FT:	.WORD	0	;FIRST TRACK
LT:	.WORD	2	;LAST TRACK

M06

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 78
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0077

3810 005670 000000
3811 005672 000023
3812 005674 000000
3813 005676 000025
3814 005700 000000
3815 005702 000000
3816 005704 000000

SO: .WORD 0 ;FIRST SECTOR IF 20(DEC) SECTOR FMT
S1: .WORD 23 ;LAST SECTOR IF 20(DEC) SECTOR FMT
S2: .WORD 0 ;FIRST SECTOR IF 22(DEC) SECTOR FMT
S3: .WORD 25 ;LAST SECTOR IF 22(DEC) SECTOR FMT
PT: .WORD 0 ;DATA PATTERN SELECT WORD
CS: .WORD 0 ;CONTROL SWITCH WORD
ST: .WORD 0 ;NUMBER OF UNIT STALLS

;DEFAULT OPERATING PARAMETER VALUES

PRDFLT:

.WORD 0 ;FC DEFAULT
.WORD 0 ;LC DEFAULT
.WORD 0 ;FT DEFAULT
.WORD 2 ;LT DEFAULT
.WORD 0 ;SO DEFAULT
.WORD 23 ;S1 DEFAULT
.WORD 0 ;S2 DEFAULT
.WORD 25 ;S3 DEFAULT
.WORD 0 ;PT DEFAULT
.WORD 0 ;CS DEFAULT
.WORD 0 ;ST DEFAULT

;OPERATING PARAMETER VALUE LOW AND HIGH LIMITS

PRMLIM:

.WORD 0 ;FC LIMITS
.WORD 0 ;LC LIMITS
.WORD 0 ;FT LIMITS
.WORD 2 ;LT LIMITS
.WORD 0 ;SO LIMITS
.WORD 23 ;S1 LIMITS
.WORD 23 ;S2 LIMITS
.WORD 25 ;S3 LIMITS
.WORD 0 ;PT LIMITS
.WORD 177777 ;CS LIMITS
.WORD 0 ;ST LIMITS
.WORD 000062
.WORD 177777

;ASCII PARAMETER MNEMONICS

PRMNEM:

.ASCII /FC/
.ASCII /LC/

3817
3818
3819
3820 005706
3821 005706 000000
3822 005710 000000
3823 005712 000000
3824 005714 000002
3825 005716 000000
3826 005720 000023
3827 005722 000000
3828 005724 000025
3829 005726 000000
3830 005730 000000
3831 005732 000000
3832
3833
3834
3835
3836 005734
3837 005734 000000
3838 005736 000000
3839 005740 000000
3840 005742 000000
3841 005744 000000
3842 005746 000002
3843 005750 000000
3844 005752 000002
3845 005754 000000
3846 005756 000023
3847 005760 000000
3848 005762 000023
3849 005764 000000
3850 005766 000025
3851 005770 000000
3852 005772 000025
3853 005774 000000
3854 005776 177777
3855 006000 000000
3856 006002 000062
3857 006004 000000
3858 006006 177777
3859
3860
3861
3862
3863 006010
3864 006010 041506
3865 006012 041514

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 79
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0078

3866	006014	052106	.ASCII	/FT/
3867	006016	052114	.ASCII	/LT/
3868	006020	030123	.ASCII	/SQ/
3869	006022	030523	.ASCII	/S1/
3870	006024	031123	.ASCII	/S2/
3871	006026	031523	.ASCII	/S3/
3872	006030	052120	.ASCII	/PT/
3873	006032	051503	.ASCII	/CS/
3874	006034	052123	.ASCII	/ST/

3875				
3876				
3877				
3878				
3879				
3880	006036			
3881	006036	177777		
3882	006040	177777		
3883	006042	177777		
3884	006044	052525		
3885	006046	052525		
3886	006050	052525		
3887	006052	177777		
3888	006054	177777		
3889	006056	052525		
3890	006060	052525		
3891	006062	177777		
3892	006064	052525		
3893	006066	177252		
3894	006070	177252		
3895	006072	172765		
3896	006074	172765		

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

177777
177777
177777
052525
052525
052525
177777
177777
052525
052525
177777
052525
177252
177252
172765
172765

3897				
3898				
3899				
3900				
3901				
3902	006076			
3903	006076	000000		
3904	006100	000000		
3905	006102	000000		
3906	006104	177777		
3907	006106	177777		
3908	006110	177777		
3909	006112	000000		
3910	006114	000000		
3911	006116	177777		
3912	006120	177777		
3913	006122	000000		
3914	006124	177777		
3915	006126	000000		
3916	006130	177777		
3917	006132	000000		
3918	006134	177777		

:DATA PATTERN 01
: HI FREQ. PHASE MIX
PAT01:

000000
000000
000000
177777
177777
177777
000000
000000
177777
177777
000000
177777
000000
177777
000000
177777

3919
3920
3921

3922
3923
3924 006136
3925 006136 052525
3926 006140 052525
3927 006142 052525
3928 006144 125252
3929 006146 125252
3930 006150 125252
3931 006152 052525
3932 006154 052525
3933 006156 125252
3934 006160 125252
3935 006162 052525
3936 006164 125252
3937 006166 052525
3938 006170 125252
3939 006172 052525
3940 006174 125252

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

052525
052525
052525
052525
125252
125252
125252
052525
052525
125252
125252
052525
125252
052525
125252
052525
125252
052525
125252

3941
3942
3943
3944
3945
3946 006176
3947 006176 133333
3948 006200 066666
3949 006202 155555
3950 006204 155555
3951 006206 133333
3952 006210 066666
3953 006212 066666
3954 006214 155555
3955 006216 155555
3956 006220 133333
3957 006222 133333
3958 006224 133333
3959 006226 133333
3960 006230 133333
3961 006232 133333
3962 006234 133333

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

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

3963
3964
3965
3966
3967
3968 006236
3969 006236 121105
3970 006240 150442
3971 006242 064221
3972 006244 132110
3973 006246 055044
3974 006250 026422
3975 006252 013211
3976 006254 105504
3977 006256 042642

;DATA PATTERN 04
; ROTATING BOUNDARY PULSE PRECOMP.
PAT04:

121105
150442
064221
132110
055044
026422
013211
105504
042642

3978 006260 021321
3979 006262 110550
3980 006264 044264
3981 006266 022132
3982 006270 011055
3983 006272 104426
3984 006274 042213
3985
3986
3987
3988
3989
3990 006276
3991 006276 026455
3992 006300 113226
3993 006302 045513
3994 006304 122645
3995 006306 151322
3996 006310 064551
3997 006312 132264
3998 006314 055132
3999 006316 026455
4000 006320 113226
4001 006322 045513
4002 006324 122645
4003 006326 151322
4004 006330 064551
4005 006332 132264
4006 006334 055132
4007
4008
4009
4010
4011 006336
4012 006336 000000
4013 006340 000000
4014 006342 000000
4015 006344 000000
4016 006346 000000
4017 006350 000000
4018 006352 000000
4019 006354 000000
4020 006356 000000
4021 006360 000000
4022 006362 000000
4023 006364 000000
4024 006366 000000
4025 006370 000000
4026 006372 000000
4027 006374 000000
4028
4029
4030
4031
4032
4033 006376

021321
110550
044264
022132
011055
104426
042213

; DATA PATTERN 05
; ROTATING CELL PULSE PRECOMP.
PAT05:

026455
113226
045513
122645
151322
064551
132264
055132
026455
113226
045513
122645
151322
064551
132264
055132

; DATA PATTERN 06
; ALL ZEROS
PAT06:

000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000
000000

; DATA PATTERN 07
; ALL ONES
PAT07:

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 82
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0081

4034	006376	177777	177777
4035	006400	177777	177777
4036	006402	177777	177777
4037	006404	177777	177777
4038	006406	177777	177777
4039	006410	177777	177777
4040	006412	177777	177777
4041	006414	177777	177777
4042	006416	177777	177777
4043	006420	177777	177777
4044	006422	177777	177777
4045	006424	177777	177777
4046	006426	177777	177777
4047	006430	177777	177777
4048	006432	177777	177777
4049	006434	177777	177777
4050			
4051			
4052			
4053			
4054	006436		
4055	006436	000001	000001
4056	006440	000002	000002
4057	006442	000004	000004
4058	006444	000010	000010
4059	006446	000020	000020
4060	006450	000040	000040
4061	006452	000100	000100
4062	006454	000200	000200
4063	006456	000400	000400
4064	006460	001000	001000
4065	006462	002000	002000
4066	006464	004000	004000
4067	006466	010000	010000
4068	006470	020000	020000
4069	006472	040000	040000
4070	006474	100000	100000
4071			
4072			
4073			
4074			
4075			
4076	006476		
4077	006476	177776	177776
4078	006500	177775	177775
4079	006502	177773	177773
4080	006504	177767	177767
4081	006506	177757	177757
4082	006510	177737	177737
4083	006512	177677	177677
4084	006514	177577	177577
4085	006516	177377	177377
4086	006520	176777	176777
4087	006522	175777	175777
4088	006524	173777	173777
4089	006526	167777	167777

DATA PATTERN 08
SHIFTED 1 IN FIELD OF ZEROS
PAT08:

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

DATA PATTERN 09
SHIFTED 0 IN FIELD OF ONES
PAT09:

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

CZR6N00 RK611/06 SS VERIF 2
CZR6N0.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 83
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0082

4090	006530	157777	157777
4091	006532	137777	137777
4092	006534	077777	077777

4093
4094
4095
4096

;DATA PATTERN 10
;ALTERNATING 0-1
PAT10:

4098	006536		
4099	006536	052525	052525
4100	006540	052525	052525
4101	006542	052525	052525
4102	006544	052525	052525
4103	006546	052525	052525
4104	006550	052525	052525
4105	006552	052525	052525
4106	006554	052525	052525
4107	006556	052525	052525
4108	006560	052525	052525
4109	006562	052525	052525
4110	006564	052525	052525
4111	006566	052525	052525
4112	006570	052525	052525
4113	006572	052525	052525
4114	006574	052525	052525

4115
4116
4117
4118

;DATA PATTERN 11
;ALTERNATING 1-0
PAT11:

4120	006576		
4121	006576	125252	125252
4122	006600	125252	125252
4123	006602	125252	125252
4124	006604	125252	125252
4125	006606	125252	125252
4126	006610	125252	125252
4127	006612	125252	125252
4128	006614	125252	125252
4129	006616	125252	125252
4130	006620	125252	125252
4131	006622	125252	125252
4132	006624	125252	125252
4133	006626	125252	125252
4134	006630	125252	125252
4135	006632	125252	125252
4136	006634	125252	125252

4137
4138
4139
4140

;DATA PATTERN 12
;SHIFTING ZEROS AND ONES
PAT12:

4142	006636		
4143	006636	000001	000001
4144	006640	000003	000003
4145	006642	000007	000007

4146	006644	000017	000017
4147	006646	000037	000037
4148	006650	000077	000077
4149	006652	000177	000177
4150	006654	000377	000377
4151	006656	000777	000777
4152	006660	001777	001777
4153	006662	003777	003777
4154	006664	007777	007777
4155	006666	017777	017777
4156	006670	037777	037777
4157	006672	077777	077777
4158	006674	177777	177777

```

;DATA PATTERN 13
;          COMPOSITE ROTATING
PAT13:

```

4163	006676		
4164	006676	072307	072307
4165	006700	135143	135143
4166	006702	156461	156461
4167	006704	167230	167230
4168	006706	073514	073514
4169	006710	035646	035646
4170	006712	016723	016723
4171	006714	107351	107351
4172	006716	143564	143564
4173	006720	061672	061672
4174	006722	030735	030735
4175	006724	114356	114356
4176	006726	046167	046167
4177	006730	123073	123073
4178	006732	151453	151453
4179	006734	164616	164616

```

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

```

ADDRESS	DATA	DESCRIPTION
4186	006736	
4187	006736	000000
4188	006740	000000
4189	006742	000000
4190	006744	000000
4191	006746	000000
4192	006750	000000
4193	006752	000000
4194	006754	000000
4195	006756	000000
4196	006760	000000
4197	006762	000000
4198	006764	000000
4199	006766	000000
4200	006770	000000
4201	006772	000000

4202 006774 000000

.WORD

4203

4204

4205

4206 006776

.USER-DEFINED DATA PATTERN 15
PAT15:

4207 006776 072307

072307

;WORD 00

4208 007000 135143

135143

;WORD 01

4209 007002 156461

156461

;WORD 02

4210 007004 167230

167230

;WORD 03

4211 007006 073514

073514

;WORD 04

4212 007010 035646

035646

;WORD 05

4213 007012 016723

016723

;WORD 06

4214 007014 107351

107351

;WORD 07

4215 007016 143564

143564

;WORD 10

4216 007020 061672

061672

;WORD 11

4217 007022 030735

030735

;WORD 12

4218 007024 114356

114356

;WORD 13

4219 007026 046167

046167

;WORD 14

4220 007030 123073

123073

;WORD 15

4221 007032 151453

151453

;WORD 16

4222 007034 164616

164616

;WORD 17

4223

4224

4225

4226 057467

MINLO1=1024375

;LO BITS OF SPEC'D MIN ROT. LATENCY

4227 000000

MINHI1=0

;HI BITS OF SPEC'D MIN ROT. LATENCY

4228 062031

MAXLO1=1025625

;LO BITS OF SPEC'D MAX ROT. LATENCY

4229 000000

MAXHI1=0

;HI BITS OF SPEC'D MAX ROT. LATENCY

4230 017500

MAXLO2=108000

;LO BITS OF SPEC'D MAX 1 CYL SEEK TIME

4231 000000

MAXHI2=0

;HI BITS OF SPEC'D MAX 1 CYL SEEK TIME

4232 112160

MAXLO3=1038000

;LO BITS OF SPEC'D MAX AVG SEEK TIME

4233 000000

MAXHI3=0

;HI BITS OF SPEC'D MAX AVG SEEK TIME

4234 022370

MAXLO4=109464

;LO BITS OF SPEC'D MAX 410 CYL SEEK TIME

4235 000001

MAXHI4=1

;HI BITS OF SPEC'D MAX 410 CYL SEEK TIME

4236 002734

RNDST=101500

;SHORT RANDOM SEEKS = 1500(10)

4237 016514

RNDLNG=107500

;LONG RANDOM SEEKS = 7500(10)

4238 000002

LSTTRK=2

;LAST TRACK = 2

4239 000365

ALNCYL=365

;ALIGNMENT CYLINDER =245(10)

4240 000002

BSERR=BIT1

;BSE ERROR

4241 000004

HVRERR=BIT2

;HVRC ERROR

4242 000010

OPIERR=BIT3

;OPI ERROR

4243 000020

DCKERR=BIT4

;DATA CHECK ERROR

4244 000040

ECCNC=BIT5

;ECC NON-CORRECTABLE

4245 000100

WCERR=BIT6

;WRITE CHECK ERROR

4246 000200

ABORT=BIT7

;ABORT

4247 000400

LEV2ER=BIT8

;LEVEL TWO ERROR

4248 001000

BADSEC=BIT9

;BAD SECTOR FLAG

4249 002000

TWOLOS=BIT10

;TWO TIME OUTS

4250 004000

RCLREQ=BIT11

;RECALIBRATE REQUIRED

4251 010000

DRNAV1=BIT12

;DRIVE NOT RELOD BY OTHER PORT

4252 100000

ANYDER=BIT15

;ANY ERROR DETECTED FLAG

4253

4254 007036 005015 055103 033122 DZR6N: .ASCIIZ <15><12>/CZR6ND0/

4255 007044 042116 000060

4256 007050 026440 051040 033113 SUBVER: .ASCIIZ & - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 3

4257 007056 030461 051057 030113

4258	007064	026466	045522	033460
4259	007072	051440	041125	054523
4260	007100	052123	046505	053040
4261	007106	051105	043111	041511
4262	007114	052101	047511	020116
4263	007122	020072	040520	052122
4264	007130	000040		
4265	007132	006461	000012	
4266	007136	006462	000012	
4267	007142	005015	046012	051501
4268	007150	020124	044120	051531
4269	007156	046440	046505	040440
4270	007164	051104	036440	000040
4271	007172	053117	051105	040514
4272	007200	020131	047514	042101
4273	007206	051105	037440	024040
4274	007214	020131	051117	047040
4275	007222	020051	020052	000
4276	007227	122	046505	053117
4277	007234	020105	054130	050104
4278	007242	050040	041501	020113
4279	007250	051106	046517	042040
4280	007256	044522	042526	020054
4281	007264	047111	052123	046101
4282	007272	020114	041523	040522
4283	007300	041524	020110	005015
4284	007306	377		
4285	007307	120	041501	026113
4286	007314	040440	042116	044040
4287	007322	052111	041440	047117
4288	007330	044524	052516	020105
4289	007336	047524	050040	047522
4290	007344	042503	042105	000056
4291	007352	005015	040520	040522
4292	007360	020115	047111	052520
4293	007366	020124	047515	042504
4294	007374	005015	000012	
4295	007400	045522	033060	030055
4296	007406	020067	052502	020123
4297	007414	042101	020122	020075
4298	007422	000		
4299	007423	122	030113	026466
4300	007430	033460	053040	041505
4301	007436	040440	051104	036440
4302	007444	000040		
4303	007446	045522	033060	030055
4304	007454	020067	051120	047511
4305	007462	020122	000075	
4306	007466	053523	020122	020075
4307	007474	000		
4308	007475	040	047040	053505
4309	007502	036440	000040	
4310	007506	005015	051104	024126
4311	007514	024523	036440	000040
4312	007522	051104	053111	020105
4313	007530	020040	000	

PART1: .ASCIZ /1/<15><12>
 PART2: .ASCIZ /2/<15><12>
 LSTMEM: .ASCIZ <15><12><12>/LAST PHYS MEM ADR = /

OVLODR: .ASCIZ /OVERLAY LOADER ? (Y OR N) * /

XXDPME: .ASCII /REMOVE XXDP PACK FROM DRIVE, INSTALL SCRATCH /<15><12><377>

.ASCIZ /PACK, AND HIT CONTINUE TO PROCEED./

PRMINP: .ASCIZ <15><12>/PARAM INPUT MODE/<15><12><12>

RKBADR: .ASCIZ /RK06-07 BUS ADR = /

RKVADR: .ASCIZ /RK06-07 VEC ADR = /

RKPRTY: .ASCIZ /RK06-07 PRIOR = /

SWRMSG: .ASCIZ /SWR = /

NEWMSG: .ASCIZ / NEW = /

DRVSEQ: .ASCIZ <15><12>/DRV(S) = /

BADDRV: .ASCIZ /DRIVE /

4314	007533	116	047117	042455	NXDRIV: .ASCIZ /NON-EXIST/<15><12>
4315	007540	044530	052123	005015	
4316	007546	000			
4317	007547	116	052117	051040	NTREDY: .ASCIZ /NOT RDY/<15><12>
4318	007554	054504	005015	000	
4319	007561	127	052122	046055	WRTLOK: .ASCIZ /WRT-LOCK/<15><12>
4320	007566	041517	006513	000012	
4321	007574	047514	042101	042105	ALNPAK: .ASCIZ /LOADED WITH ALIGN PACK/<15><12>
4322	007602	053440	052111	020110	
4323	007610	046101	043511	020116	
4324	007616	040520	045503	005015	
4325	007624	000			
4326	007625	015	005012	025052	NODRTS: .ASCII <15><12><12>/** NO DRVS TO TEST/<15><12>
4327	007632	047040	020117	051104	
4328	007640	051526	052040	020117	
4329	007646	042524	052123	005015	
4330	007654	051120	051505	020123	CNTRDY: .ASCIZ /PRESS "CONT" WHEN RDY/<15><12>
4331	007662	041442	047117	021124	
4332	007670	053440	042510	020116	
4333	007676	042122	006531	000012	
4334	007704	040510	052114	051040	HLTRQD: .ASCIZ /HALT REQ/<15><12>
4335	007712	050505	005015	000	
4336	007717	015	005012	047524	ALDRVS: .ASCIZ <15><12><12>/TO TEST ALL DRVS TYPE "A" <CR>, ELSE <CR>/<15><12>/** /
4337	007724	052040	051505	020124	
4338	007732	046101	020114	051104	
4339	007740	051526	052040	050131	
4340	007746	020105	040442	020042	
4341	007754	041474	037122	020054	
4342	007762	046105	042523	036040	
4343	007770	051103	006476	025012	
4344	007776	000040			
4345	010000	005015	020114	020075	TSTMDS: .ASCII <15><12>/L = LIST TESTS/<15><12>
4346	010006	044514	052123	052040	
4347	010014	051505	051524	005015	
4348	010022	020103	020075	044103	.ASCII /C = CHANGE TESTS/<15><12>
4349	010030	047101	042507	052040	
4350	010036	051505	051524	005015	
4351	010044	020111	020075	047111	.ASCIZ /I = INPUT PARAMS, RUN TESTS/<15><12>
4352	010052	052520	020124	040520	
4353	010060	040522	051515	020054	
4354	010066	052522	020116	042524	
4355	010074	052123	006523	000012	
4356	010102	005015	047105	042524	ENTLCI: .ASCIZ <15><12>/ENTER L,C, OR I/<15><12>/** /
4357	010110	020122	026114	026103	
4358	010116	047440	020122	006511	
4359	010124	025012	000040		
4360	010130	047524	042040	043105	DFTEST: .ASCIZ /TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12>/** /
4361	010136	052501	052114	052040	
4362	010144	051505	051524	052040	
4363	010152	050131	020105	020104	
4364	010160	041474	037122	020054	
4365	010166	046105	042523	036040	
4366	010174	051103	006476	025012	
4367	010202	000040			
4368	010204	005015	042524	052123	TLSTHD: .ASCIZ <15><12>/TEST ITERATIONS/<15><12>
4369	010212	020040	020040	044440	

4370	010220	042524	040522	044524	
4371	010226	047117	006523	000012	
4372	010234	005015	020124	020075	EXPLAN: .ASCII <15><12>/T = TYPE PARAM LIST/<15><12>
4373	010242	054524	042520	050040	
4374	010250	051101	046501	046040	
4375	010256	051511	006524	012	
4376	010263	117	036440	047440	.ASCII /O = OPEN PARAM LIST/<15><12>
4377	010270	042520	020116	040520	
4378	010276	040522	020115	044514	
4379	010304	052123	005015		
4380	010310	020123	020075	042523	.ASCII /S = SET INDIVIDUAL PARAM/<15><12>
4381	010316	020124	047111	044504	
4382	010324	044526	052504	046101	
4383	010332	050040	051101	046501	
4384	010340	005015			
4385	010342	020122	020075	052522	.ASCIIZ /R = RUN TESTS/<15><12>
4386	010350	020116	042524	052123	
4387	010356	006523	000012		
4388	010362	005015	047105	042524	PARMDE: .ASCIIZ <15><12>/ENTER T,O,S, OR R/<15><12>
4389	010370	020122	026124	026117	
4390	010376	026123	047440	020122	
4391	010404	006522	000012		
4392	010410	005015	020052	052504	DUACES: .ASCIIZ <15><12>/* DUAL-ACCESS DATA TEST */<15><12>
4393	010416	046101	040455	041503	
4394	010424	051505	020123	040504	
4395	010432	040524	052040	051505	
4396	010440	020124	006452	000012	
4397	010446	040515	020130	047527	MAWRDC: .ASCIIZ /MAX WORD COUNT = /
4398	010454	042122	041440	052517	
4399	010462	052116	036440	000040	
4400	010470	025052	020040	030123	SECNL1: .ASCII /** S0>S1/
4401	010476	051476	061		
4402	010501	040	047516	020124	NOTALD: .ASCIIZ / NOT ALLOWED/<15><12>
4403	010506	046101	047514	042527	
4404	010514	006504	000012		
4405	010520	025052	020040	031123	SECNL2: .ASCIIZ /** S2>S3/
4406	010526	051476	000063		
4407	010532	025052	020040	052106	TRKNLW: .ASCIIZ /** FT>LT/
4408	010540	046076	000124		
4409	010544	025052	020040	041527	WC2BIG: .ASCIIZ /** WC OR MA TOO LARGE/<15><12>
4410	010552	047440	020122	040515	
4411	010560	052040	047517	046040	
4412	010566	051101	042507	005015	
4413	010574	000			
4414	010575	124	020117	042504	DFQUES: .ASCIIZ /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR>/<15><12>/** /
4415	010602	040506	046125	020124	
4416	010610	046101	020114	040520	
4417	010616	040522	051515	020054	
4418	010624	054524	042520	042040	
4419	010632	036040	051103	026076	
4420	010640	042440	051514	020105	
4421	010646	041474	037122	005015	
4422	010654	020052	000		
4423	010657	015	052412	042523	PFIFTN: .ASCIIZ <15><12>/USER-DEF PATT 15 :/<15><12>
4424	010664	026522	042504	020106	
4425	010672	040520	052124	030440	

K07

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 89
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0088

4426	010700	020065	006472	000012	
4427	010706	005015	047515	020104	SELP15: .ASCIZ <15><12>/MOD PATT 15 :/<15><12>
4428	010714	040520	052124	030440	
4429	010722	020065	006472	000012	
4430	010730	047524	046440	042117	MDFY15: .ASCIZ /TO MOD PATT 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
4431	010736	050040	052101	020124	
4432	010744	032461	020054	054524	
4433	010752	042520	046440	036040	
4434	010760	051103	026076	042440	
4435	010766	051514	020105	041474	
4436	010774	037122	005015	020052	
4437	011002	000			
4438	011003	015	042412	052116	ENTPAS: .ASCIZ <15><12>/ENTER NO. OF PASSES (1-77777) :/<15><12>/* /
4439	011010	051105	047040	027117	
4440	011016	047440	020106	040520	
4441	011024	051523	051505	024040	
4442	011032	026461	033467	033467	
4443	011040	024467	035040	005015	
4444	011046	020052	000		
4445	011051	015	025012	020052	DROPDR: .ASCIZ <15><12>/** DROPPING DRV - 20 ERRORS/<15><12>
4446	011056	042040	047522	050120	
4447	011064	047111	020107	051104	
4448	011072	020126	020055	030062	
4449	011100	042440	051122	051117	
4450	011106	006523	000012		
4451	011112	005015	052012	051505	TSTDRN: .ASCII <15><12><12>/TESTING DRV /
4452	011120	044524	043516	042040	
4453	011126	053122	040		
4454	011131	040	005015	000	DRVNO: .ASCIZ /<15><12>
4455	011135	104	053122	000040	DRIV: .ASCIZ /DRV /
4456	011142	040503	052122	020056	CART: .ASCIZ /CART. /
4457	011150	000			
4458	011151	123	051105	020056	SERNM: .ASCIZ /SER. NO. /
4459	011156	047515	020056	000040	
4460	011164	005015	040506	052103	FACTBS: .ASCIZ <15><12>/FACTORY /
4461	011172	051117	020131	000	
4462	011177	012	047523	052106	SOFTBS: .ASCII <12>/SOFTWARE /
4463	011204	040527	042522	040	
4464	011211	102	042101	051440	BDSECT: .ASCIZ /BAD SECTORS :/<15><12>
4465	011216	041505	047524	051522	
4466	011224	035040	005015	000	
4467	011231	040	047040	047117	NOFALS: .ASCIZ / NONE/
4468	011236	000105			
4469					
4470					
4471					MESSAGES USED IN OFFSET-TO-FAILURE TEST
4472	011240	005015	047412	043106	OFSHED: .ASCII <15><12><12>/OFFSET-TO-FAILURE MEAS :/<15><12><12>
4473	011246	042523	026524	047524	
4474	011254	043055	044501	052514	
4475	011262	042522	046440	040505	
4476	011270	020123	006472	005012	
4477	011276	051124	041501	020113	.ASCII /TRACK CYLN SECT +OFST -OFST/<15><12>
4478	011304	041440	046131	020116	
4479	011312	051440	041505	020124	
4480	011320	025440	043117	052123	
4481	011326	020040	047455	051506	

4482	011334	006524	012		
4483	011337	011	020011	020040	.ASCIZ <011><011>/ (UIN) (UIN)/<15><12>
4484	011344	052450	047111	020051	
4485	011352	024040	044525	024516	
4486	011360	005015	000		
4487					
4488					
4489	011363	000			UNLOD: .BYTE 0 ;DRIVE UNLOADED INDICATOR
4490	011364	000			VERIFY: .BYTE 0 ;VERIFY MODE FLAG
4491	011365	015	005012	025052	IDENT: .ASCIZ <15><12><12>/*** RK06-07 HEAD ALIGN AID ***/<15><12>
4492	011372	020052	045522	033060	
4493	011400	030055	020067	042510	
4494	011406	042101	040440	044514	
4495	011414	047107	040440	042111	
4496	011422	025040	025052	005015	
4497	011430	000			
4498	011431	106	051117	044040	FORHLP: .ASCIZ /FOR HELP TYPE H, ELSE <CR>/<15><12>
4499	011436	046105	020120	054524	
4500	011444	042520	044040	020054	
4501	011452	046105	042523	036040	
4502	011460	051103	006476	000012	
4503	011466	046412	047101	040525	TYPMOD: .ASCIZ <12>/MANUAL OR AUTO MODE (M OR A)?/<15><12>
4504	011474	020114	051117	040440	
4505	011502	052125	020117	047515	
4506	011510	042504	024040	020115	
4507	011516	051117	040440	037451	
4508	011524	005015	000		
4509	011527	012	025012	046440	TYPMAN: .ASCIZ <12><12>/* MANUAL SELECT MODE */<15><12>
4510	011534	047101	040525	020114	
4511	011542	042523	042514	052103	
4512	011550	046440	042117	020105	
4513	011556	006452	000012		
4514	011562	042412	052116	051105	ENTDRV: .ASCIZ <12>/ENTER DRV NO. (0-7):/<15><12>
4515	011570	042040	053122	047040	
4516	011576	027117	024040	026460	
4517	011604	024467	006472	000012	
4518	011612	025052	020040	051104	NOTRDY: .ASCII /** DRV /
4519	011620	020126			
4520	011622	020040	047516	020124	DRNRDY: .ASCII / NOT RDY ! START DRV IF NECESSARY./<15><12>
4521	011630	042122	020131	020041	
4522	011636	051440	040524	052122	
4523	011644	042040	053122	044440	
4524	011652	020106	042516	042503	
4525	011660	051523	051101	027131	
4526	011666	005015			
4527	011670	020040	020040	044127	.ASCIZ / WHEN DRV IS RDY, PRESS "CONT" ON CPU./<15><12>
4528	011676	047105	042040	053122	
4529	011704	044440	020123	042122	
4530	011712	026131	050040	042522	
4531	011720	051523	021040	047503	
4532	011726	052116	020042	047117	
4533	011734	041440	052520	006456	
4534	011742	000012			
4535	011744	025052	020040	051104	NONEXD: .ASCII /** DRV /
4536	011752	020126			
4537	011754	020040	047516	026516	DRVNED: .ASCIZ / NON-EXIST OR OFF-LINE ! PRESS "CONT" TO RESTART./<15><12>

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 91
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0090

4538 011762 054105 051511 020124
4539 011770 051117 047440 043106
4540 011776 046055 047111 020105
4541 012004 020041 051120 051505
4542 012012 020123 041442 047117
4543 012020 021124 052040 020117
4544 012026 042522 052123 051101
4545 012034 027124 005015 000
4546 012041 052 020052 042040
4547 012046 053127 040
4548 012051 040 047040 052117
4549 012056 053440 052122 046055
4550 012064 041517 020113 020041
4551 012072 046120 040505 042523
4552 012100 051440 052105 053440
4553 012106 052122 046055 041517
4554 012114 051440 020127 006456
4555 012122 012
4556 012123 040 020040 052040
4557 012130 042510 020116 051120
4558 012136 05505 020123 041442
4559 012144 047117 021124 047440
4560 012152 020116 050103 027125
4561 012160 005015 000
4562 012163 052 020052 046440
4563 012170 046125 020124 051104
4564 012176 020126 052501 047524
4565 012204 051455 046105 041505
4566 012212 042524 020104 020041
4567 012220 051120 051505 020123
4568 012226 041447 047117 023524
4569 012234 052040 020117 042522
4570 012242 052123 051101 027124
4571 012250 005015 000
4572 012253 012 046101 043511
4573 012260 026116 042526 044522
4574 012266 054506 020054 051117
4575 012274 042440 042530 041522
4576 012302 051511 020105 040450
4577 012310 053054 020054 051117
4578 012316 042440 020051 006477
4579 012324 000012
4580 012326 005012 020052 040522
4581 012334 042116 046517 051440
4582 012342 042505 020113 054105
4583 012350 051105 044503 042523
4584 012356 020123 047111 050040
4585 012364 047522 051107 051505
4586 012372 020123
4587 012374 047117 042040 044522
4588 012402 042526 040
4589 012405 040 025040 005015
4590 012412 000
4591 012413 012 025012 046440
4592 012420 047101 040525 020114
4593 012426 042523 042514 052103

NOTLOK: .ASCII /** DRV /

MODRLK: .ASCII / NOT WRT-LOCK ! PLEASE SET WRT-LOC SW ./'15'<12>

.ASCIZ / THEN PRESS "CONT" ON CPU./'15'<12>

MULSEL: .ASCIZ /** MULT DRV AUTO-SELECTED ! PRESS 'CONT' TO RESTART./'15'<12>

ENTMOD: .ASCIZ <12>/ALIGN,VERIFY, OR EXERCISE (A,V, OR E) ?/'15'<12>

MANEXR: .ASCII <12><12>/* RANDOM SEEK EXERCISES IN PROGRESS /

.ASCII /ON DRIVE /

DRIEXR: .ASCIZ / */'15'<12>

MANALN: .ASCIZ <12><12>/* MANUAL SELECT ALIGNMENT */'15'<12>

NO7

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND0.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 92
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0091

4594	012434	040440	044514	047107	
4595	012442	042515	052116	025040	
4596	012450	005015	000		
4597	012453	012	025012	046440	MANVRF: .ASCIZ <12><12> /* MANUAL SELECT VERIFY */<15><12>
4598	012460	047101	040525	020114	
4599	012466	042523	042514	052103	
4600	012474	053040	051105	043111	
4601	012502	020131	006452	000012	
4602	012510	042510	042101	020123	HEDPOS: .ASCIZ /HEADS AT CYL 365 (OCT) (760 RK07)/<15><12>
4603	012516	052101	041440	046131	
4604	012524	031440	032466	024040	
4605	012532	041517	024524	024040	
4606	012540	033067	020060	045522	
4607	012546	033460	006451	000012	
4608	012554	047105	042524	020122	DTYMSG: .ASCIZ /ENTER DRIVE TYPE /
4609	012562	051104	053111	020105	
4610	012570	054524	042520	000040	
4611	012576	020040	020040	035040	SOR7: .ASCIZ / : <6> OR <7> /
4612	012604	036040	037066	047440	
4613	012612	020122	033474	000076	
4614	012620	042412	052116	051105	ENTHED: .ASCIZ <12> /ENTER HEAD NO. (0-2):/<15><12>
4615	012626	044040	040505	020104	
4616	012634	047516	020056	030050	
4617	012642	031055	035051	005015	
4618	012650	000			
4619	012651	124	050131	020105	RWNRDY: .ASCIZ /TYPE <R> WHEN RDY :/<15><12>
4620	012656	051074	020076	044127	
4621	012664	047105	051040	054504	
4622	012672	035040	005015	000	
4623	012677	110	040505	020104	HEDSEL: .ASCII /HEAD /
4624	012704	020040	042523	042514	HEADNO: .ASCIZ / SELECTED/<15><12>
4625	012712	052103	042105	005015	
4626	012720	000			
4627	012721	012	025012	040440	TYPAUT: .ASCIZ <12><12> /* AUTO SELECT MODE */<15><12>
4628	012726	052125	020117	042523	
4629	012734	042514	052103	046440	
4630	012742	042117	020105	006452	
4631	012750	000012			
4632	012752	005012	020052	052501	AUTALN: .ASCIZ <12><12> /* AUTO SELECT ALIGNMENT */<15><12>
4633	012760	047524	051440	046105	
4634	012766	041505	020124	046101	
4635	012774	043511	046516	047105	
4636	013002	020124	006452	000012	
4637	013010	005012	020052	052501	AUTVRF: .ASCIZ <12><12> /* AUTO SELECT VERIFY */<15><12>
4638	013016	047524	051440	046105	
4639	013024	041505	020124	042526	
4640	013032	044522	054506	025040	
4641	013040	005015	000		
4642	013043	012	051104	020126	DRISEL: .ASCII <12> /DRV /
4643	013050	020040	042523	042514	DRVSEL: .ASCIZ / SELECTED/<15><12><12>
4644	013056	052103	042105	005015	
4645	013064	000012			
4646	013066	005012	020052	052501	AUTEXR: .ASCII <12><12> /* AUTO SELECT EXERCISES */<15><12>
4647	013074	047524	051440	046105	
4648	013102	041505	020124	054105	
4649	013110	051105	044503	042523	

```

4650 013116 020123 006452 012
4651 013123 012 044123 051117 .ASCIZ <12>/SHORT OR LONG (S OR L) ?/<15><12>
4652 013130 020124 051117 046040
4653 013136 047117 020107 051450
4654 013144 047440 020122 024514
4655 013152 037440 005015 000
4656 013157 105 042530 041522 AUTOEX: .ASCII /EXERCISING DRV /
4657 013164 051511 047111 020107
4658 013172 051104 020126
4659 013176 006440 000012
4660
4661 013202 025052 020040 040503 BAD632: .ASCIZ /** CANNOT READ BAD SECTOR TRACK!/<15><12>
4662 013210 047116 052117 051040
4663 013216 040505 020104 040502
4664 013224 020104 042523 052103
4665 013232 051117 052040 040522
4666 013240 045503 006441 000012
4667 013246 041536 005015 000 CNTRLC: .ASCIZ /↑C/<15><12>
4668 013253 136 006532 000012 CNTRLZ: .ASCIZ /↑Z/<15><12>
4669 013260 051136 005015 000 CNTRLR: .ASCIZ /↑R/<15><12>
4670 013265 136 006525 000012 CNTRLU: .ASCIZ /↑U/<15><12>
4671 013272 043536 005015 000 CNTRLG: .ASCIZ /↑G/<15><12>
4672 013277 015 005012 000 CR2LF: .ASCIZ <15><12><12>
4673 013303 134 000 BKSLSH: .ASCIZ /\
4674 013305 054 000 COMMA: .ASCIZ /,/
4675 013307 040 040 SPACE6: .ASCII //
4676 013311 040 SPACE4: .ASCII //
4677 013312 040 SPACE3: .ASCII //
4678 013313 040 SPACE2: .ASCII //
4679 013314 000040 SPACE1: .ASCIZ //
4680 .EVEN
4681 013316 020040 000075 PRMBUF: .ASCIZ / = /
4682 013322 047527 042122 000040 WORDSP: .ASCIZ /WORD /
4683 013330 036440 000040 EQUALS: .ASCIZ / = /
4684 013334 020052 000 PROMPT: .ASCIZ /* /
4685 013337 076 000040 PRMPSP: .ASCIZ /> /
4686
4687 .EVEN
4688
4689
4690
4691
4692 013342 105037 003116 DFSTRT: CLRB MDFLAG ;SET FLAG FOR DEFAULT MODE
4693 013346 105037 003135 CLRB DULACS ;CLEAR DUAL-ACCESS FLAG
4694 013352 105037 003126 CLRB ERRCNT ;CLEAR ERROR COUNT FOR RESTARTS
4695 013356 022737 014434 000042 CMP #DRVTST,2#42 ;SEE IF EOP RETURN ADRS = DRVTST
4696 013364 001003 4$ BNE 4$ ;BR IF NOT DRVTST
4697 013366 012737 017320 000042 MOV #NEWPAS,2#42 ;SET RETURN ADRS = NEWPAS
4698 013374 000405 4$: BR CMSTRT ;PROCEED
4699
4700
4701 013376 112737 000001 003116 PSTART: MOV #1,MDFLAG ;SET FLAG FOR PARAMETER MODE
4702 013404 105037 003135 CLRB DULACS ;CLEAR DUAL-ACCESS TEST FLAG
4703
4704 013410 012737 000340 177776 CMSTRT: MOV #PR7,2#PS ;BLOCK ALL INTERRUPTS
4705 .SBTTL INITIALIZE THE COMMON TAGS

```

```

4706      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
4707      MOV    $CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
4708      CLR    (R6)+          ;;CLEAR MEMORY LOCATION
4709      CMP    $SWR,R6      ;;DONE?
4710      BNE    -6             ;;LOOP BACK IF NO
4711      MOV    $STACK,SP      ;;SETUP THE STACK POINTER
4712      ;;INITIALIZE A FEW VECTORS
4713      MOV    $SCOPE,$IOTVEC  ;;IOT VECTOR FOR SCOPE ROUTINE
4714      MOV    $340,$IOTVEC+2  ;;LEVEL 7
4715      MOV    $ERROR,$EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
4716      MOV    $340,$EMTVEC+2  ;;LEVEL 7
4717      MOV    $TRAP,$TRAPVEC  ;;TRAP VECTOR FOR TRAP CALLS
4718      MOV    $340,$TRAPVEC+2  ;;LEVEL 7
4719      MOV    $SPWRDN,$PWAVEC  ;;POWER FAILURE VECTOR
4720      MOV    $340,$PWAVEC+2  ;;LEVEL 7
4721      MOV    $ENDCT,$EOPCT    ;;SETUP END-OF-PROGRAM COUNTER
4722      CLR    $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
4723      CLR    $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
4724      MOVB   $1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
4725      MOV    $,$SLPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
4726      MOV    $,$SLPERR      ;;SETUP THE ERROR LOOP ADDRESS
4727      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
4728      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
4729      MOV    $ERRVEC,($SP)    ;;SAVE ERROR VECTOR
4730      MOV    $64,$ERRVEC     ;;SET UP ERROR VECTOR
4731      MOV    $OSWR,$SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
4732      MOV    $DDISP,$DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
4733      CMP    $-1,$SWR        ;;TRY TO REFERENCE HARDWARE SWR
4734      BNE    $66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
4735      ;;AND THE HARDWARE SWR IS NOT = -1
4736      BR     $65$           ;;BRANCH IF NO TIMEOUT
4737      MOV    $65,$($SP)      ;;SET UP FOR TRAP RETURN
4738      RTI
4739      MOV    $SWREG,$SWR      ;;POINT TO SOFTWARE SWR
4740      MOV    $DISPREG,$DISPLAY
4741      MOV    ($SP)+,$ERRVEC  ;;RESTORE ERROR VECTOR
4742
4743      CLR    $PASS           ;;CLEAR PASS COUNT
4744      BITB   $APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
4745      BEQ    $67$           ;;YES,USE NON-APT SWITCH
4746      MOV    $SSWREG,$SWR    ;;NO,USE APT SWITCH REGISTER
4747
4748      67$: CLR    AUTOFG      ;;CLEAR THE SPECIAL FLAG
4749      RESET  ;;CLEAR THE UNIBUS
4750      MOV    $6,$ERRVEC      ;;SET TIME-OUT VECTOR
4751      CLR    $ERRVEC+2
4752      MOV    $MPERHD,$MEMVEC  ;;SET MEM PARITY TRAP VECTOR
4753      MOV    $PR7,$MEMVEC+2  ;;SET TRAP PRIORITY = 7
4754      JSR    PC,$ENBCSR      ;;ENABLE MEMORY PARITY CHECK
4755      MOVB   $100,$ERMAX     ;;SET MAX ERROR CNT TO 100 FOR $SCOPE
4756      CLRB   $DRVER$        ;;CLEAR ERROR COUNT FOR CURRENT DRIVE
4757      MOVB   $1,$HLPVOL      ;;SET HLP FILE OVLD INDICTR
4758      TYPE   ,DZREN          ;;TYPE PROGRAM I.D. FOR PART 2
4759      TYPE   ,SUBVER
4760      TYPE   ,PART2
4761      TSTB   NOTYPE          ;;SEE IF OPERATOR NOTE SHOULD BE TYPED

```

4762	013766	001004		BNE	39\$: BR IF NOT
4763	013770	104401	065660	TYPE			: TYPE OPERATOR NOTE
4764	013774	105237	003146	INCB			: SET FLAG FOR NEXT TIME
4765	014000	105737	003135	39\$: TSTB	DULACS		: SEE IF DUAL-ACCESS DATA TEST
4766	014004	001402		BEQ	40\$: BR IF NOT
4767	014006	104401	010410	TYPE			: TYPE ** DUAL-ACCESS DATA TEST **
4768	014012	012737	030436	000060 40\$: MOV	#KACHOL, #TKVEC		: LOAD VECTOR FOR TTY KBD
4769	014020	012737	000200	000062 MOV	#PR4, #TKVEC+2		: SET KBD PRIORITY = 4
4770	014026	013701	003040	MOV	RKVEC, R1		: ADDR. OF RK06 VECTOR STORAGE
4771	014032	012721	047304	MOV	#I, INTR, (R1)+		: SET IT TO RK06 HANDLER
4772	014036	013711	003042	MOV	RKPRI, (R1)		: SET RK06 PRIORITY
4773	014042	012737	031246	000250 MOV	#KTERHD, #MMVEC		: VECT FOR KT11 FAILURE
4774	014050	012737	000340	000252 MOV	#PR7, #MMVEC+2		: SET PRIOR. = 7 FOR HNDLER
4775	014056	105037	003120	CLRB	TSTING		: CLEAR "RUNNING TESTS" FLAG
4776	014062	012737	000000	177776 MOV	#PRO, #PS		: ALLOW ALL INTERRUPTS AGAIN
4777	014070	012701	005224	MOV	#PRVCMO, R1		: ZERO OUT PREVIOUS COMMAND
4778	014074	012700	000006	MOV	#6, RO		: SET COUNTER
4779	014100	005021		42\$: CLR	(R1)+		: ZERO A WORD
4780	014102	005300		DEC	RO		
4781	014104	001375		BNE	42\$: BR IF NOT DONE YET
4782	014106	005037	005502	CLR	STALLS		: DON'T ALLOW STALLS YET
4783	014112	023727	000042	017320 CMP	#42, #NEWPAS		: SEE IF CHAIN MODE
4784	014120	101002		BHI	44\$: BR IF YES
4785	014122	004737	031126	JSR	PC, GTSWRG		: OPEN SOFTWARE SWR FOR MODIFICATION
4786	014126	012737	176543	054732 44\$: MOV	#176543, \$HINUM		: INIT. PSEUDO-RANDOM NOS.
4787	014134	012737	123456	054734 MOV	#123456, \$LONUM		
4788	014142	004737	030630	JSR	PC, SIZMEM		: SIZE MEMORY, FIX MA LIMIT IN PRMLIM
4789	014146	105737	003134	TSTB	XDPSVD		: SEE IF XXDP, PREVIOUSLY SAVED
4790	014152	001404		BEQ	9\$: BR IF NOT
4791	014154	004737	032050	JSR	PC, GETXDP		: RESTORE SAVED XXDP
4792	014160	105037	003134	CLRB	XDPSVD		: CLEAR THE FLAG
4793	014164	105737	003116	9\$: TSTB	MDFLAG		: SEE IF DEFAULT MODE
4794	014170	001031		BNE	10\$: BR IF NOT DEFAULT MODE
4795	014172	013700	001370	MOV	\$VECT1, RO		: GET APT RK06 VECTOR AND PRTY
4796	014176	013737	001374	003036 MOV	\$BASE, RKBAS		: GET APT RK06 BASE ADDRESS
4797	014204	132737	000200	001341 BITB	#BIT7, \$ENVM		: SEE IF NO SIZING
4798	014212	001005		BNE	18\$: BR IF NO SIZING
4799	014214	012700	120210	MOV	\$AVECT1, RO		: GET DEFAULT VECTOR AND PRIORITY
4800	014220	012737	177440	003036 MOV	\$ABASE, RKBAS		: GET DEFAULT BASE ADDRESS
4801	014226	110037	003040	18\$: MOV	RO, RKVEC		: STORE VECTOR
4802	014232	105037	003041	CLRB	RKVEC+1		: CLEAR HI BYTE
4803	014236	000300		SWAB	RO		: GET PRTY INTO BITS 5-7
4804	014240	042700	177437	BIC	#177437, RO		: CLEAR OTHER BITS
4805	014244	010037	003042	MOV	RO, RKPRI		: STORE RK06 PRIORITY
4806	014250	000137	015120	JMP	ALLDRV		: GO CHECK ALL DRIVES
4807							
4808							
4809							
4810							
4811	014254	104401	007352	10\$: TYPE	, PRMINP		: BEGIN PARAMETER INPUT MODE
4812							: TYPE "PARAMETER INPUT MODE"
4813							
4814	014260	013746	003036	MOV	RKBAS, -(SP)		: PUT OLD VALUE ON STACK
4815	014264	104401	007400	TYPE	RKBAOR		: TYPE "RK06 BUS ADR = "
4816	014270	004737	031016	JSR	PC, GETPRM		: TYPE OLD, GET NEW RKBAS VALUE
4817	014274	012637	003036	MOV	(SP)+, RKBAS		: STORE NEW VALUE

4818				;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION	
4819	014300	013746	003040	MOV RKVEC, -(SP)	;PUT OLD VALUE ON STACK
4820	014304	104401	007423	TYPE RKVADR	;TYPE "RK06 VEC ADR = "
4821	014310	004737	031016	JSR PC, GETPRM	;TYPE OLD, GET NEW RKVEC VALUE
4822	014314	012637	003040	MOV (SP)+, RKVEC	;STORE NEW VALUE
4823				;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION	
4824	014320	013746	003042	11\$: MOV RKPRI, -(SP)	;GET OLD VALUE OF PRIORITY
4825	014324	006316		ASL (SP)	;GET IT INTO BITS 0-2
4826	014326	006316		ASL (SP)	
4827	014330	006316		ASL (SP)	
4828	014332	000316		SWAB (SP)	
4829	014334	104401	007446	TYPE RKPRTY	;TYPE "RK06 PRIORITY = "
4830	014340	004737	031016	JSR PC, GETPRM	;TYPE OLD, GET NEW RKPRI VALUE
4831	014344	012600		MOV (SP)+, RO	
4832	014346	020027	000004	CMP RO, #4	;SEE IF AT LEAST LEVEL 4
4833	014352	002414		BLT 12\$;BR IF NEW VALUE TOO SMALL
4834	014354	020027	000007	CMP RO, #7	;SEE IF LEVEL 7 OR LESS
4835	014360	003011		BGT 12\$;BR IF NEW VALUE TOO LARGE
4836	014362	000300		SWAB RO	;GET PRIORITY INTO BITS 5-7
4837	014364	006270		ASR RO	
4838	014366	006200		ASR RO	
4839	014370	006200		ASR RO	
4840	014372	010037	003042	MOV RO, RKPRI	;STORE NEW VALUE
4841	014376	104401	001315	TYPE \$CRLF	
4842	014402	000405		BR 16\$	
4843	014404	104401	005264	12\$: TYPE ,BUFFO	;ECHO BAD INPUT
4844	014410	104401	001314	TYPE ,SQUES	
4845	014414	000741		BR 11\$;GO ASK AGAIN
4846					

F08

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 97
INITIALIZE THE COMMON TAGS

SEQ 0096

4847 014416 105037 003126

16\$: CLRB ERRCNT

;CLEAR ERROR CNT FOR RESTARTS

G08

CZR6ND0 RK611/06 SS VERIF 2 MACY11 30A(1052) 10-JAN-78 12:02 PAGE 98
CZR6ND.P11 10-JAN-78 10:05 INITIALIZE THE COMMON TAGS

SEQ 0097

4848 014422 012737 014434 000042 MOV #DRVTST,2#42 ;SET UP DUMP MODE RETURN FROM SEOP
4849 ;UPON COMPLETION OF REQUESTED PASSES
4850 014430 000137 015176 JMP CHKLST ;GO CHECK STATUS OF MARKED DRIVES
4851
4852
4853

```

4854 ;*****
4855 ;SBTTL TO - DESIRED DRIVE INPUT ROUTINE
4856 ;*THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
4857 ;*WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
4858 ;*CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
4859 ;*DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
4860 ;*AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
4861 ;*LIST (DRVLST).
4862 ;*****
4863
4864 014434 DRVTST:
4865 014434 012706 001100 MOV #STACK,SP ;RESET THE STACK
4866 014440 005037 005502 CLR STALLS ;INHIBIT STALL BETWEEN OPERATIONS
4867 014444 004737 031342 JSR PC,ENBCSR ;ENABLE MEMORY PARITY CHECK
4868 014450 104401 007717 TYPE ALDRVS ;ASK IF ALL DRIVES DESIRED
4869 014454 004737 034034 JSR PC,RDCHRS ;READ RESPONSE
4870 014460 014434 DRVTST ;(1C) RETURN ADDRESS
4871 014462 014434 DRVTST ;(1Z) RETURN ADDRESS
4872 014464 014434 DRVTST ;(1U) OR ERROR RETURN ADDRESS
4873 014466 005700 TST R0 ;SEE IF NULL INPUT
4874 014470 001413 BEQ TELDRV ;BR IF NULL INPUT
4875 014472 022737 000101 005264 CMP #A,BUFF0 ;SEE IF ALL DRIVES REQUESTED
4876 014500 001002 BNE 4$ ;BR IF NOT ALL DRIVES
4877 014502 000137 015120 JMP ALLDRV ;CHECK ALL DRIVES
4878 014506 104401 005264 4$: TYPE ,BUFF0 ;ECHO BAD INPUT
4879 014512 104401 001314 TYPE ,SQUES ;TYPE '?' AND <CR>,<LF>
4880 014516 000746 BR DRVTST ;GO ASK AGAIN
4881 ;TYPE CURRENT DRIVE LIST
4882 014520 104401 007506 TELDRV: TYPE ,DRVSEQ ;TYPE "DRIVE(S) = "
4883 014524 005001 CLR R1 ;INITIALIZE DRIVE NUMBER
4884 014526 005000 CLR R0 ;INIT. COUNT OF LISTED DRIVES
4885 014530 012703 000001 MOV #BIT0,R3 ;INIT BIT POINTER
4886 014534 105761 005610 4$: TSTB DRVLST(R1) ;SEE IF THIS DRIVE IS LISTED
4887 014540 001414 BEQ 8$ ;BR IF DRIVE NOT LISTED
4888 014542 005700 TST R0 ;SEE IF FIRST TIME HERE
4889 014544 001402 BEQ 6$ ;BR IF FIRST TIME HERE
4890 014546 104401 013305 TYPE ,COMMA ;TYPE A COMMA
4891 014552 010137 005534 6$: MOV R1,SCRACH ;USE SCRACH FOR BUFFER
4892 014556 052737 000060 005534 BIS #0,SCRACH ;CONVERT DRIVE NO. TO ASCII
4893 014564 104401 005534 TYPE ,SCRACH ;TYPE DRIVE NO.
4894 014570 005200 INC R0 ;INCREMENT NO. OF LISTED DRIVES
4895 014572 005201 8$: INC R1 ;INCREMENT DRIVE NO.
4896 014574 006303 ASL R3 ;SHIFT BIT POINTER
4897 014576 022701 000010 CMP #10,R1 ;SEE IF DONE YET
4898 014602 001354 BNE 4$ ;BR IF NOT DONE
4899 014604 005700 TST R0 ;SEE IF ANY DRIVES WERE LISTED
4900 014606 001016 BNE 26$ ;BR IF YES
4901 014610 104401 011231 TYPE ,NOFALS ;TYPE " NONE"
4902 014614 104401 007625 TYPE ,NODRTS ;TYPE "** NO DRIVES TO TEST"
4903 ;"PRESS 'CONT' WHEN RDY"
4904 014620 012703 000401 MOV #401,R3 ;PATTERN TO MARK DRIVES
4905 014624 012701 005610 MOV #DRVLST,R1 ;DRIVE LIST ADDRESS
4906 014630 010321 MOV R3,(R1)+ ;MARK ALL DRIVES IN LIST
4907 014632 010321 MOV R3,(R1)+
4908 014634 010321 MOV R3,(R1)+
4909 014636 010311 MOV R3,(R1)

```

4910	014640	000137	046150			JMP	HLTPRG	;HALT PROGRAM
4911	014644	104401	001315			26\$: TYPE	\$CRLF	;TYPE <CR>,<LF>
4912	014650	122737	000013	000041		CMPB	#13,#41	;DID DRIVE LOAD XXDP??
4913	014656	001003				BNE	28\$;NO, SO SKIP NEXT
4914	014660	104401	007227			TYPE	,XXDPME	;INFORM USER TO REMOVE XXDP PACK
4915	014664	000000				HALT		;AND FORCE OPERATOR INTERVENTION
4916	014666	105037	000041			28\$: CLR8	#41	;OPERATOR HIT CONT. CLR THE LD PTR
4917	014672	105737	003116			TSTB	MDFLAG	;SEE IF DEFAULT MODE
4918	014676	001012				BNE	19\$;BR IF NOT DEFAULT MODE
4919	014700	000137	015536			JMP	LODFLS	;GO LOAD DEFLT ITER. COUNTS
4920	014704	001007				BNE	19\$;BR IF NOT
4921	014706	105737	005610			TSTB	DRVLST	;SEE IF DRIVE 0 LISTED TO TEST
4922	014712	001404				BEQ	19\$;BR IF NOT
4923	014714	004737	032310			JSR	PC,INITSS	;INIT THE SUB-SYS
4924	014720	000137	015176			JMP	CHKLST	;GO CHECK STATUS OF LISTED DRIVES
4925	014724	104401	013334			19\$: TYPE	,PROMPT	;TYPE ASTERISK AND SPACE
4926						;READ AND CHECK	NEW DRIVE NUMBERS	
4927	014730	004737	034034			JSR	PC,ROCHRS	;READ IN INPUT STRING
4928	014734	014434				DRVTST		;(<C> RETURN ADDRESS
4929	014736	014434				DRVTST		;(<Z> RETURN ADDRESS
4930	014740	014520				TELDV		;(<U> OR ERROR RETURN ADDRESS
4931	014742	005700				TST	RO	;SEE IF NULL INPUT
4932	014744	001007				BNE	9\$;BR IF NEW DRIVES WILL BE SELECTED
4933	014746	105737	003135			TSTB	DULACS	;SEE IF DUAL-ACCESS FLAG SET
4934	014752	001002				BNE	22\$;BR IF YES
4935	014754	000137	015250			JMP	ASKTST	;JUMP TO THE TEST INPUT ROUTINE
4936	014760	000137	016014			22\$: JMP	INPUTP	;GO ASK FOR INPUT PARAMETERS
4937	014764	022700	000017			9\$: CMP	#15,RO	;SEE IF TOO MANY CHARACTERS TYPED
4938	014770	002005				BGE	12\$;BR IF NOT TOO MANY
4939	014772	104401	005264			10\$: TYPE	,BUFFO	;ECHO BAD INPUT
4940	014776	104401	001314			TYPE	\$QUES	;TYPE <?> AND <CR>,<LF>
4941	015002	000646				BR	TELDV	;GO TYPE DRIVES AND ASK AGAIN
4942	015004	005001				12\$: CLR	R1	;CLEAR CHAR. POINTER
4943	015006	126127	005264	000060		14\$: CMPB	BUFFO(R1),#0	;SEE IF DRIVE NO. < 0
4944	015014	002766				BLT	10\$;BR TO ECHO BAD INPUT
4945	015016	126127	005264	000067		CMPB	BUFFO(R1),#7	;SEE IF > 7
4946	015024	003362				BGT	10\$;BR TO ECHO BAD INPUT
4947	015026	005201				INC	R1	;INCREMENT CHAR POINTER
4948	015030	020100				CMP	R1,RO	;SEE IF MORE CHARS TO CHECK
4949	015032	001406				BEQ	16\$;BR IF ALL CHARS CHECKED
4950	015034	126127	005264	000054		CMPB	BUFFO(R1),#'	;SEE IF THIS IS COMMA
4951	015042	001353				BNE	10\$;BR IF NOT COMMA
4952	015044	005201				INC	R1	;INCREMENT CHAR POINTER
4953	015046	000757				BR	14\$;BR TO CONTINUE CHECKING CHARS
4954						;GET NEW DRIVE NUMBERS INTO LIST		
4955	015050	012701	005610			16\$: MOV	#DRVLST,R1	;GET DRIVE LIST ADDRESS
4956	015054	005021				CLR	(R1)+	;CLEAR OUT THE DRIVE LIST
4957	015056	005021				CLR	(R1)+	
4958	015060	005021				CLR	(R1)+	
4959	015062	005011				CLR	(R1)	
4960	015064	005000				CLR	RO	;CLEAR BUFFER POINTER
4961	015066	116001	005264			18\$: MOV8	BUFFO(RO),R1	;GET A DRIVE NUMBER
4962	015072	042701	000060			BIC	#0,R1	;STRIP ASCII BITS
4963	015076	112761	000001	005610		MOV8	#1,DRVLST(R1)	;MARK THIS DRIVE IN DRIVE LIST
4964	015104	062700	000002			ADD	#2,RO	;POINT TO NEXT DRIVE NUMBER
4965	015110	105760	005263			TSTB	BUFFO-1(RO)	;SEE IF NULL CHAR YET

```

4966 015114 J01364      BNE 18$      ;BR IF NOT DONE YET
4967 015116 000427      BR  CHKLST    ;GO CHECK NEW DRIVES FOR VALID STATUS
4968                                     ;COME HERE IF ALL DRIVES REQUESTED
4969 015120 005000      ALLDRV: CLR RO      ;CLEAR DRIVE NUMBER
4970 015122 013703 001376  MOV $DEVN,R3 ;GET APT DEVICE MAP
4971 015126 132737 000200 001341  BITB #BIT7,$ENVN ;SEE IF NO SIZING
4972 015134 001002      BNE 1$      ;BR IF NO SIZING
4973 015136 012703 000377      MOV #377,R3 ;SET UP FOR SIZING
4974 015142 012701 000001      MOV #BIT0,R1 ;SET BIT POINTER
4975 015146 105060 005610 2$: CLRB DRVLST(RO) ;INITIALIZE DRIVE ENTRY
4976 015152 030103      BIT R1,R3 ;SEE IF THIS DRIVE IS REQUESTED
4977 015154 001403      BEQ 4$      ;BR IF NOT
4978 015156 112760 000001 005610 4$: MOVB #1,DRVLST(RO) ;MARK THIS DRIVE IN DRIVE LIST
4979 015164 006301      ASL R1 ;SHIFT BIT POINTER
4980 015166 005200      INC RO ;INCREMENT DRIVE NUMBER
4981 015170 022700 000010      CMP #10,RO ;SEE IF DONE MARKING ALL DRIVES
4982 015174 001364      BNE 2$      ;BR IF NOT DONE YET
4983                                     ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
4984 015176 005000      CHKLST: CLR RO      ;CLEAR DRIVE NUMBER
4985 015200 105760 005610 2$: TSTB DRVLST(RO) ;SEE IF THIS DRIVE IS MARKED IN LIST
4986 015204 001410      BEQ 4$      ;BR IF NOT MARKED
4987 015206 010C37 005500      MOV RO,DRIVE ;SET DRIVE NUMBER FOR SCNDRV
4988 015212 004737 032436      JSR PC,SCNDRV ;CHECK STATUS OF THIS DRIVE
4989                                     ;IF NOT VALID, TYPE MESSAGE
4990                                     ;AND REMOVE IT FROM DRIVE LIST
4991 015216 015242      6$      ;ERROR RETURN ADDRESS FOR SCNDRV
4992 015220 113760 003124 005620 4$: MOVB TYPFMT,DTYLIST(RO) ;SET DRV TYP, 0=RK06, 4=RK07
4993 015226 005200      INC RO ;RETURN HERE IF DRIVE IS USEABLE
4994 015230 022700 000010      CMP #10,RO ;SEE IF DONE CHECKING LIST
4995 015234 001361      BNE 2$      ;BR IF NOT DONE YET
4996 015236 000137 014520      JMP TELDRV ;GO BACK TO LIST SELECTED DRIVES
4997 015242 105060 005610 6$: CLRB DRVLST(RO) ;REMOVE INVALID DRIVE FROM LIST
4998 015246 000767      BR 4$      ;CONTINUE CHECKING THE LIST
4999
5000
5001 ;*****
5002 ;SBTTL TO - DESIRED TEST INPUT ROUTINE
5003 ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
5004 ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
5005 ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
5006 ;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
5007 ;*THAT THE TEST IS NOT TO BE RUN.
5008 ;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
5009 ;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
5010 ;*TYPED, (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
5011 ;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
5012 ;*OF OPERATING PARAMETERS, AND RUN TESTS.
5013 ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
5014 ;* (↑Z) CAUSES RETURN TO ASKTMD.
5015 ;*****
5016
5017 ; DETERMINE L,C, OR I MODE
5018 015250 104401 010000 ASKTST: TYPE ,TSTMDS ;ASK FOR TEST INPUT MODE
5019 015254 104401 010102 ASKTMD: TYPE ,ENTLCI ;TYPE "ENTER L,C, OR I"
5020 015260 004737 034034 JSR PC,RDCHRS ;READ RESPONSE
5021 015264 014434      DRVTST ;(↑C) RETURN ADDRESS

```

```

5022 015266 015254 ASKTM0 ;(1Z) RETURN ADDRESS
5023 015270 015254 ASKTM0 ;(1U) OR ERROR RETURN ADDRESS
5024 015272 005700 TST RO ;SEE IF ANY INPUT
5025 015274 001005 BNE 4$ ;BR IF ANY INPUT
5026 015276 104401 005264 2$: TYPE ,BUFF0 ;ECHO BAD INPUT
5027 015302 104401 001314 TYPE $QUES ;TYPE <?> AND <CR>,<LF>
5028 015306 000762 BR ASKTM0 ;GO ASK AGAIN
5029 015310 022737 000114 005264 4$: CMP #'L,BUFF0 ;SEE IF (L) TYPED
5030 015316 001002 BNE 6$ ;BR IF NOT (L)
5031 015320 000137 015354 JMP LSTTST ;JUMP TO LIST TESTS
5032 015324 022737 000103 005264 6$: CMP #'C,BUFF0 ;SEE IF (C) TYPED
5033 015332 001002 BNE 8$ ;BR IF NOT (C)
5034 015334 000137 015472 JMP CHGTST ;JUMP TO CHANGE TESTS
5035 015340 022737 000111 005264 8$: CMP #'I,BUFF0 ;SEE IF (I) TYPED
5036 015346 001353 BNE 2$ ;BR IF NOT (I), TO ECHO BAD INPUT
5037 015350 000137 016014 JMP INPUTP ;GO INPUT PARAMS AND RUN TESTS
5038
5039 ;LIST CURRENT TESTS AND ITERATION COUNTS
5040 015354 104401 010204 LSTTST: TYPE TLSTHD ;TYPE HEADING "TEST ITERATIONS"
5041 015360 005001 CLR R1 ;INITIALIZE TEST INDEX
5042 015362 004737 030576 2$: JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5043 015366 004737 034346 JSR PC,TYPTST ;TYPE CURRENT TEST AND ITERATION NUMBER
5044 015372 104401 001315 TYPE $CRLF ;TYPE <CR>,<LF>
5045 015376 005737 005522 TST INTCHR ;SEE IF ANY INPUT
5046 015402 001416 BEQ 8$ ;BR IF NO INPUT
5047 015404 122737 000003 005522 CMPB #003,INTCHR ;SEE IF (1C) TYPED
5048 015412 001002 BNE 4$ ;BR IF NOT (1C)
5049 015414 000137 014434 JMP DRVTST ;JUMP TO ASK FOR DRIVES AGAIN
5050 015420 122737 000032 005522 4$: CMPB #032,INTCHR ;SEE IF (1Z) TYPED
5051 015426 001002 BNE 6$ ;BR IF NOT (1Z)
5052 015430 000137 015254 JMP ASKTM0 ;JUMP TO ASK FOR NEW TEST INPUT MODE
5053 015434 004737 030616 6$: JSR PC,ECOBAD ;ECHO BAD INPUT
5054 015440 062701 000002 8$: ADD #2,R1 ;INCREMENT TEST INDEX
5055 015444 010102 MOV R1,R2 ;GET COPY OF INDEX
5056 015446 062702 005630 ADD #TSTLST,R2 ;GET POSITION IN LIST
5057 015452 022702 005644 CMP #DFLTST,R2 ;SEE IF DONE WITH LIST
5058 015456 001341 BNE 2$ ;BR IF NOT DONE YET
5059 015460 042777 000100 163456 BIC #BIT6,$STKS ;DISABLE KBD INTERRUPT
5060 015466 000137 015254 JMP ASKTM0 ;GO ASK FOR NEW TEST INPUT MODE
5061
5062 ;CHANGE CURRENT TESTS AND ITERATION COUNTS
5063 015472 104401 010130 CHGTST: TYPE DFTST ;ASK IF TESTS SHOULD BE DEFAULTED
5064 015476 004737 034034 JSR PC,ROCHRS ;READ RESPONSE
5065 015502 014434 DRVTST ;(1C) RETURN ADDRESS
5066 015504 015254 ASKTM0 ;(1Z) RETURN ADDRESS
5067 015506 015472 CHGTST ;(1U) OR ERROR RETURN ADDRESS
5068 015510 005700 TST RO ;SEE IF NULL INPUT
5069 015512 001426 BEQ NULINP ;BR IF NULL INPUT
5070 015514 022737 000104 005264 CMP #'D,BUFF0 ;SEE IF (D) TYPED
5071 015522 001405 BEQ LODFLS ;BR IF DEFAULTS REQUESTED
5072 015524 104401 005264 TYPE ,BUFF0 ;ECHO BAD INPUT
5073 015530 104401 001314 TYPE $QUES ;TYPE <?> AND <CR>,<LF>
5074 015534 000756 BR CHGTST ;GO ASK AGAIN
5075 ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
5076 015536 012700 005644 LODFLS: MOV #DFLTST,R0 ;LOAD DEFAULT LIST ADDRESS
5077 015542 012702 005630 MOV #TSTLST,R2 ;LOAD TEST LIST ADDRESS

```

5078	015546	012022		4\$:	MOV	(R0)+(R2)+	; LOAD A DEFAULT VALUE	
5079	015550	022702	005644		CMP	#DFLTST,R2	; SEE IF DONE YET	
5080	015554	001374			BNE	4\$; BR IF NOT DONE YET	
5081	015556	105737	003116		TSTB	MDFLAG	; SEE IF DEFAULT MODE	
5082	015562	001234			BNE	ASKTMD	; BR IF NOT DEFAULT MODE	
5083	015564	000137	016550		JMP	LODFPT	; GO LOAD DEFAULT PARAMETERS	
5084	015570	005001		NULINP:	CLR	R1	; INITIALIZE TEST INDEX	
5085	015572	104401	010204		TYPE	TLSTHD	; TYPE HEADING "TEST ITERATIONS"	
5086					TYPE	CURRENT TEST AND ITERATION	NUMBER	
5087	015576	004737	034346	8\$:	JSR	PC,TYPTST	; TYPE A TEST NO. AND ITERATION NO.	
5088	015602	104401	013314		TYPE	,SPACE1	; TYPE A SPACE	
5089	015606	104401	013334		TYPE	PROMPT	; TYPE ASTERISK AND SPACE	
5090					; READ AND CHECK	INPUT, IF ANY		
5091	015612	004737	034034		JSR	PC,RDCHRS	; READ OCTAL DIGITS TYPED	
5092	015616	014434			DRVTST		; (↑C) RETURN ADDRESS	
5093	015620	015254			ASKTMD		; (↑Z) RETURN ADDRESS	
5094	015622	015576		8\$; (↑U) OR ERROR RETURN ADDRESS	
5095	015624	005700			TST	R0	; SEE IF ANY INPUT	
5096	015626	001012			BNE	12\$; BR IF ANY INPUT	
5097	015630	062701	000002	10\$:	ADD	#2,R1	; INCREMENT INDEX	
5098	015634	010102			MOV	R1,R2	; COPY INDEX	
5099	015636	062702	005630		ADD	#TSTLST,R2	; GET POSITION IN LIST	
5100	015642	022702	005644		CMP	#DFLTST,R2	; SEE IF DONE WITH LIST	
5101	015646	001353			BNE	8\$; BR IF NOT DONE YET	
5102	015650	000137	015254		JMP	ASKTMD	; GO ASK FOR NEW TEST INPUT MODE	
5103	015654	022737	000041	005264	12\$:	CMP	#',,BUFF0	; SEE IF (') TYPED
5104	015662	001431			BEQ	16\$; BR TO PROPAGATE CURRENT ITER. NO.	
5105	015664	005002			CLR	R2	; INITIALIZE (') INDICATOR	
5106	015666	122760	000041	005263	CMPB	#',,BUFF0-1(R0)	; SEE IF LAST CHAR IN BUF IS (')	
5107	015674	001004			BNE	14\$; BR IF NOT (')	
5108	015676	105060	005263		CLRB	BUFF0-1(R0)	; INSERT TERMINATOR BYTE	
5109	015702	005300			DEC	R0	; DECREMENT CHAR COUNT	
5110	015704	005202			INC	R2	; SET (') INDICATOR	
5111	015706	022700	000005	14\$:	CMP	#5,R0	; SEE HOW MANY CHARS NOW	
5112	015712	002433			BLT	20\$; BR IF TOO MANY (MAX ITER. NO. = 77776)	
5113	015714	012746	005264		MOV	#BUFF0,-(SP)	; GET BUF ADDR. ON STACK FOR OCTBIN	
5114	015720	004737	054500		JSR	PC,OCTBIN	; CHECK DIGITS AND CONVERT TO BINARY	
5115	015724	016002			20\$; ERROR RETURN ADDRESS FOR OCTBIN	
5116	015726	012600			MOV	(SP)+,R0	; GET ITERATION NUMBER	
5117	015730	020027	077776		CMP	R0,#77776	; SEE IF > 77776	
5118	015734	101022			BHI	20\$; BR IF TOO BIG	
5119	015736	010061	005630		MOV	R0,TSTLST(R1)	; PUT ITERATION NUMBER INTO TEST LIST	
5120	015742	005702			TST	R2	; SEE IF (') WAS TYPED	
5121	015744	001731			BEQ	10\$; BR IF NOT (')	
5122					PROPAGATE CURRENT	ITERATION NO.	TO THE END OF TSTLST	
5123	015746	010102		16\$:	MOV	R1,R2	; COPY INDEX	
5124	015750	062702	005630		ADD	#TSTLST,R2	; GET POSITION IN LIST	
5125	015754	022702	005642		CMP	#DFLTST-2,R2	; SEE IF AT END OF LIST	
5126	015760	001002			BNE	17\$; BR IF NOT DONE	
5127	015762	000137	015254		JMP	ASKTMD	; GO ASK FOR NEW TEST INPUT MODE	
5128	015766	016161	005630	005632	17\$:	MOV	TSTLST(R1),TSTLST	+2(R1); PROPAGATE TO NEXT WORD
5129	015774	062701	000002		ADD	#2,R1	; INCREMENT INDEX	
5130	016000	000762			BR	16\$; BR TO CONTINUE	
5131	016002	104401	005264	20\$:	TYPE	,BUFF0	; ECHO BAD INPUT	
5132	016006	104401	001314		TYPE	,SQUES	; TYPE '<' AND '<CR>,<LF>	
5133	016012	000671			BR	8\$; GO ASK AGAIN	


```

5134
5135
5136 016014 104401 010234
5137 016020 005037 005502
5138 016024 004737 031342
5139 016030 104401 010362
5140 016034 104401 013334
5141 016040 004737 034034
5142 016044 014434
5143 016046 016020
5144 016050 016020
5145 016052 005700
5146 016054 001005
5147 016056 104401 005264
5148 016062 104401 001314
5149 016066 000754
5150 016070 022737 000124 005264
5151 016076 001002
5152 016100 000137 016232
5153 016104 022737 000117 005264
5154 016112 001002
5155 016114 000137 016504
5156 016120 022737 000123 005264
5157 016126 001002
5158 016130 000137 016770
5159 016134 022737 000122 005264
5160 016142 001345
5161 016144 023737 005670 005672
5162 016152 003403
5163 016154 104401 010470
5164 016160 000717
5165 016162 023737 005674 005676
5166 016170 003405
5167 016172 104401 010520
5168 016176 104401 010501
5169 016202 000706
5170 016204 023737 005664 005666
5171 016212 003405
5172 016214 104401 010532
5173 016220 104401 010501
5174 016224 000675
5175 016226 000137 017224
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187 016232
5188 016232 005C01
5189 016234 004737 030576

;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
INPUTP: TYPE EXPLAN
ASKMDE: CLR STALLS ;EXPLAIN INPUT MODES
JSR PC,ENBCSR ;INHIBIT STALL BETWEEN OPERATIONS
;ENABLE MEMORY PARITY CHECK
TYPE ' ' ;ASK FOR DESIRED INPUT MODE
;TYPE "*"
;READ RESPONSE TO MODE QUESTION
;(+C) RETURN ADDRESS
;(+Z) RETURN ADDRESS
;(+U) OR ERROR RETURN ADDRESS
;SEE IF NULL INPUT
BR IF ANY INPUT
ECHO BAD INPUT
TYPE '<' AND '<CR>,<LF>'
GO ASK AGAIN
;SEE IF (T) TYPED
BR IF NOT (T)
JUMP TO THE TYPE LIST ROUTINE
;SEE IF (O) TYPED
BR IF NOT (O)
JUMP TO THE OPEN LIST ROUTINE
;SEE IF (S) TYPED
BR IF NOT (S)
JUMP TO THE SET INDIV. PARAM. ROUTINE
;SEE IF (R) TYPED
BR IF NOT (R) TO ECHO BAD INPUT
COMPARE 20(DEC) SECTOR LIMITS
BR IF SO NOT > S1
TYPE "SO>S1 NOT ALLOWED"
GO ASK AGAIN FOR PARAMETERS
COMPARE 22(DEC) SECTOR LIMITS
BR IF S2 NOT > S3
TYPE "S2>S3"
TYPE "NOT ALLOWED"
GO ASK AGAIN FOR PARAMETERS
COMPARE CHOSEN TRACK LIMITS
BR IF FT NOT > LT
TYPE "FT>LT"
TYPE "NOT ALLOWED"
GO ASK AGAIN FOR PARAMETERS
GO RUN THE TESTS

.SBTTL TO - TYPE (T) LIST ROUTINE
;THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
;CONSOLE, LINE BY LINE. EACH LINE HAS THE FOLLOWING
;FORMAT: XX=YYYYYY WHERE XX IS A PARAMETER MNEMONIC
;AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
;TYPING (+C) CAUSES IMMEDIATE RETURN TO DRVTST,
;AND (+Z) CAUSES RETURN TO ASKMDE.

TYPLST: CLR R1 ;INITIALIZE INDEX
JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT

```

```

5190 016240 004737 034402      1$: JSR PC,TYPPRM ;TYPE CURRENT PARAMETER AND VALUE
5191 016244 104401 001315      TYPE $CRLF ;TYPE <CR>,<LF>
5192 016250 005737 005522      TST INTCHR ;SEE IF ANY INPUT AT KBD
5193 016254 001420      BEQ 8$ ;BR IF NO INPUT
5194 016256 122737 000003 005522 CMPB #003,INTCHR ;SEE IF (↑C) TYPED
5195 016264 001002      BNE 4$ ;BR IF NOT (↑C)
5196 016266 000137 014434      JMP DRVTST ;JUMP TO ASK FOR DRIVE(S) AGAIN
5197 016272 122737 000032 005522 4$: CMPB #032,INTCHR ;SEE IF (↑Z) TYPED
5198 016300 001002      BNE 6$ ;BR IF NOT (↑Z)
5199 016302 000137 016020      JMP ASKMDE ;JUMP TO ASK FOR NEW MODE
5200 016306 004737 030616      6$: JSR PC,ECOBAD ;ECHO BAD INPUT
5201 016312 004737 030576      JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5202 016316 022761 040515 006010 8$: CMP #MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
5203 016324 001002      BNE 10$ ;BR IF NOT (MA)
5204 016326 062701 000002      ADD #2,R1 ;INCREMENT PARAMETER INDEX
5205 016332 062701 000002      10$: ADD #2,R1 ;INCREMENT PARAMETER INDEX
5206 016336 010102      MOV R1,R2 ;GET COPY OF INDEX
5207 016340 062702 005660      ADD #PRMLST,R2 ;COMPUTE POSITION IN LIST
5208 016344 022702 005706      CMP #PROFLT,R2 ;SEE IF DONE WITH LIST
5209 016350 001333      BNE 1$ ;BR IF NOT DONE YET
5210 016352 032737 100000 005700 BIT #BIT15,PT ;SEE IF PATTERN 15 SPECIFIED
5211 016360 001005      BNE 12$ ;BR IF PATTERN SPECIFIED
5212 016362 042777 000100 162554 BIC #BIT6,2$TKS ;DISABLE KBD INTERRUPT
5213 016370 000137 016020      JMP ASKMDE ;JUMP TO ASK FOR NEW MODE
5214      ;TYPE OUT USER-DEFINED PATTERN 15
5215 016374 104401 010657      12$: TYPE PF15TN ;TYPE "USER-DEFINED PATTERN 15 : "
5216 016400 005001      CLR R1 ;INITIALIZE WORD INDEX
5217 016402 005737 005522      14$: TST INTCHR ;SEE IF ANY INPUT
5218 016406 001420      BEQ 20$ ;BR IF NO INPUT
5219 016410 122737 000003 005522 CMPB #003,INTCHR ;SEE IF (↑C) TYPED
5220 016416 001002      BNE 16$ ;BR IF NOT (↑C)
5221 016420 000137 014434      JMP DRVTST ;JUMP TO ASK FOR DRIVE(S) AGAIN
5222 016424 122737 000032 005522 16$: CMPB #032,INTCHR ;SEE IF (↑Z) TYPED
5223 016432 001002      BNE 18$ ;BR IF NOT (↑Z)
5224 016434 000137 016020      JMP ASKMDE ;JUMP TO ASK FOR NEW MODE
5225 016440 004737 030616      18$: JSR PC,ECOBAD ;ECHO BAD INPUT
5226 016444 004737 030576      JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5227 016450 004737 034462      20$: JSR PC,TYPPAT ;TYPE WORD XX = YYYYYY
5228 016454 104401 001315      TYPE $CRLF ;TYPE <CR>,<LF>
5229 016460 062701 000002      ADD #2,R1 ;INCREMENT INDEX
5230 016464 022701 000040      CMP #32,R1 ;SEE IF 16 WORDS TYPED
5231 016470 001344      BNE 14$ ;BR IF NOT DONE YET
5232 016472 042777 000100 162444 BIC #BIT6,2$TKS ;DISABLE KBD INTERRUPT
5233 016500 000137 016020      JMP ASKMDE ;JUMP TO ASK FOR NEW MODE

```

.SBTTL TO - OPEN (O) LIST ROUTINE

;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
 ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
 ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
 ;*TTY INPUT. TYPING (↑C) CAUSES IMMEDIATE RETURN TO
 ;*DRVTST, AND (↑Z) CAUSES RETURN TO ASKMDE.

5245 016504

OPNLST:

5246	016504	104401	010575		TYPE	OFQUES	:ASK IF ALL DEFAULT VALUES DESIRED
5247	016510	004737	034034		JSR	PC,RDCHRS	:READ RESPONSE TO DEFAULT QUESTION
5248	016514	014434			DRVTST		: (1C) RETURN ADDRESS
5249	016516	016020			ASKMOE		: (1Z) RETURN ADDRESS
5250	016520	016504			OPNLST		: (1U) OR ERROR RETURN ADDRESS
5251	016522	005700			TST	RO	:SEE IF NULL INPUT
5252	016524	001433			BEQ	NOTDFT	:BR IF DEFAULTS NOT REQUESTED
5253	016526	022737	000104	005264	CMP	#D,BUFFO	:SEE IF (D) TYPED
5254	016534	001405			BEQ	LODFPT	:BR IF DEFAULTS DESIRED
5255	016536	104401	005264		TYPE	,BUFFO	:ECHO BAD INPUT
5256	016542	104401	001314		TYPE	,SQUES	
5257	016546	000756			BR	OPNLST	:GO ASK AGAIN
5258					;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST		
5259	016550	012700	005706		LODFPT: MOV	#PRDFLT,R0	:LOAD DEFAULT LIST ADDRESS
5260	016554	012702	005660		MOV	#PRMLST,R2	:LOAD PARAMETER LIST ADDRESS
5261	016560	012022			4\$: MOV	(R0)+,(R2)+	:LOAD A DEFAULT VALUE
5262	016562	022702	005706		CMP	#PRDFLT,R2	:SEE IF DONE YET
5263	016566	001374			BNE	4\$:BR IF MORE VALUES TO LOAD YET
5264	016570	105737	003116		TSTB	MDFLAG	:SEE IF DEFAULT MODE
5265	016574	001005			BNE	6\$:BR IF NOT DEFAULT MODE
5266	016576	012737	000001	024006	MOV	#1,\$EOPCT	:SET PASS COUNT = 1
5267	016604	000137	017306		JMP	STPASS	:GO RUN DFLT TESTS
5268	016610	000137	016020		6\$: JMP	ASKMOE	:JUMP TO ASK FOR NEW MODE
5269	016614	005001			NOTDFT: CLR	R1	:INITIALIZE INDEX
5270					:TYPE CURRENT PARAMETER AND VALUE		
5271	016616	004737	034402		8\$: JSR	PC,TYPPRM	:TYPE PARAMETER AND VALUE
5272	016622	104401	013314		TYPE	,SPACE1	:TYPE A SPACE
5273	016626	104401	013334		TYPE	,PROMPT	:TYPE ASTERISK AND SPACE
5274					;READ AND CHECK INPUT, IF ANY		
5275	016632	004737	034034		JSR	PC,RDCHRS	:READ OCTAL DIGITS TYPED
5276	016636	014434			DRVTST		: (1C) RETURN ADDRESS FOR RDCHRS
5277	016640	016020			ASKMOE		: (1Z) RETURN ADDRESS FOR RDCHRS
5278	016642	016616			8\$:		: (1U) OR ERROR RETURN ADDR. FOR RDCHRS
5279	016644	005700			TST	RO	:SEE IF ANY INPUT
5280	016646	001007			BNE	10\$:BR IF ANY INPUT
5281	016650	022761	040515	006010	CMP	#MA,PRMNM(R1)	:SEE IF (MA) JUST DEFAULTED
5282	016656	001023			BNE	12\$:BR IF NOT (MA)
5283	016660	062701	000002		ADD	#2,R1	:INCREMENT INDEX
5284	016664	000420			BR	12\$:BR TO MOVE ON TO NEXT PARAMETER
5285	016666	022700	000010		10\$: CMP	#10,R0	:SEE IF 8 CHARACTERS TYPED
5286	016672	002431			BLT	14\$:BR IF MORE THAN 8 TYPED
5287	016674	012746	005264		MOV	#BUFFO,-(SP)	:PUT BUFFER ADDR. ON STACK FOR OCTBIN
5288	016700	004737	054500		JSR	PC,OCTBIN	:CHECK DIGITS AND CONVERT TO BINARY
5289	016704	016756			14\$:		:ERROR RETURN ADDRESS FOR OCTBIN
5290	016706	012637	005470		MOV	(SP)+,LOWOCT	:GET LOW BINARY BITS
5291	016712	013737	054632	005472	MOV	\$HIOCT,HIGOCT	:GET HIGH BINARY BITS
5292					;CHECK PARAMETER VALUE AND PUT INTO LIST		
5293	016720	004737	035040		JSR	PC,CHKPRM	:CHECK VALIDITY OF PARAM VALUE
5294	016724	016756			14\$:		:ERROR RETURN ADDR. FOR CHKPRM
5295					;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY		
5296	016726	004737	034514		12\$: JSR	PC,MODP15	
5297					;MOVE ON TO NEXT PARAMETER		
5298	016732	062701	000002		ADD	#2,R1	:INCREMENT THE PARAMETER INDEX
5299	016736	010102			MOV	R1,R2	:GET COPY OF INDEX
5300	016740	062702	005660		ADD	#PRMLST,R2	:COMPUTE POSITION IN LIST
5301	016744	022702	005706		CMP	#PRDFLT,R2	:SEE IF DONE WITH LIST

5302 016750 001322
5303 016752 000137 016020
5304 016756 104401 005264
5305 016762 104401 001314
5306 016766 000713

14\$: BNE 8\$;BR IF NOT DONE YET
JMP ASKMOE ;JUMP TO ASK FOR NEW MODE
TYPE ,BUFFO ;ECHO BAD INPUT
TYPE ,SQUES ;TYPE (?) AND (CR), (LF)
BR 8\$;BR TO ASK AGAIN

.SBTTL TO - SET (S) INDIVIDUAL PARAM ROUTINE
;THIS ROUTINE IS USED TO ALLOW SETTING OF ANY INDIVIDUAL
;PARAMETER VALUE BY TTY INPUT. A PROMPTER (>) IS
;TYPED AND THE ROUTINE WAITS FOR INPUT OF THE FORM
; * XX=YYYYYY, WHERE XX IS THE MNEMONIC FOR ANY VALID
;PARAMETER IN THE LIST, AND YYYYYY IS THE DESIRED PARAMETER
;VALUE IN OCTAL DIGITS. THE PROGRAM THEN ASKS FOR ANOTHER
;PARAMETER BY TYPING ">" AGAIN. TYPING (↑) CAUSES
;IMMEDIATE RETURN TO DRVST, AND (↑Z) CAUSES RETURN TO
;ASKMOE.

5321 016770
5322 016770 104401 013337
5323
5324 016774 004737 034034
5325 017000 014434
5326 017002 016020
5327 017004 016770
5328 017006 020027 000004
5329 017012 002005
5330 017014 104401 005264
5331 017020 104401 001314
5332 017024 000761
5333 017026 020027 000013
5334 017032 003370
5335
5336 017034 005001
5337 017036 023761 005264 006010
5338 017044 001411
5339 017046 062701 000002
5340 017052 010102
5341 017054 062702 005660
5342 017060 022702 005706
5343 017064 001364
5344 017066 000752
5345
5346 017070 012702 000002
5347 017074 122762 000075 C05264
5348 017102 001411
5349 017104 122762 000040 005264
5350 017112 001340
5351 017114 005202
5352 017116 122762 000075 005264
5353 017124 001333
5354 017126 005202
5355 017130 020200
5356 017132 002330
5357 017134 122762 000040 005264

SETPRM: TYPE PRMPSP ;TYPE ">" PROMPTER
;READ AND CHECK INPUT, IF ANY
JSR PC, RDCHRS ;READ INPUT LINE
DRVST ;(↑) RETURN ADDRESS FOR RDCHRS
ASKMOE ;(↑Z) RETURN ADDRESS FOR RDCHRS
SETPRM ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
CMP RO, #4 ;SEE IF AT LEAST 4 CHARACTERS TYPED
BGE 6\$;BR IF AT LEAST 4 TYPED
4\$: TYPE ,BUFFO ;ECHO BAD INPUT
TYPE ,SQUES ;TYPE (?) AND (CR), (LF)
BR SETPRM ;BR TO ASK FOR INPUT AGAIN
6\$: CMP RO, #11. ;SEE IF ELEVEN OR LESS CHARS TYPED
BGT 4\$;BR IF MORE THAN ELEVEN TYPED
;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
CLR R1 ;INITIALIZE PARAMETER INDEX
8\$: CMP BUFFO, PRMNEM(R1) ;TRY TO MATCH 2-CHAR MNEMONIC
BEQ 10\$;BR IF PARAMETER FOUND IN TABLE
ADD #2, R1 ;INCREMENT INDEX
MOV R1, R2 ;GET COPY OF INDEX
ADD #PRMLST, R2 ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
CMP #PROFLT, R2
BNE 8\$;BR IF MORE TO CHECK YET
BR 4\$;BR TO ECHO BAD INPUT
;CHECK FOR VALID LINE FORMAT
10\$: MOV #2, R2 ;INITIALIZE BUFFER POINTER
CMPB #'=, BUFFO(R2) ;SEE IF NEXT CHAR IS "="
BEQ 12\$;BR IF IT IS "="
CMPB #040, BUFFO(R2) ;SEE IF NEXT CHAR IS A SPACE
BNE 4\$;BR TO ECHO BAD INPUT
INC R2 ;INCREMENT BUFFER POINTER
CMPB #'=, BUFFO(R2) ;SEE IF NEXT CHAR IS "="
BNE 4\$;BR TO ECHO INVALID INPUT
12\$: INC R2 ;INCREMENT BUFFER POINTER
CMP R2, RO ;SEE IF MORE CHARS LEFT IN BUFFER
BGE 4\$;BR TO ECHO INVALID INPUT
CMPB #040, BUFFO(R2) ;SEE IF NEXT CHARACTER IS SPACE

```

5358 017142 001003      BNE 14$      ;BR IF NOT A SPACE
5359 017144 005202      INC R2      ;INCREMENT BUFFER POINTER
5360 017146 020200      CMP R2,R0    ;SEE IF MORE CHARS LEFT IN BUFFER
5361 017150 002321      BGE 4$      ;BR IF NONE LEFT
5362 017152 160200      14$: SUB R2,R0  ;SUBTRACT POINTER FROM CHAR COUNT
5363 017154 020027      CMP R0,#10   ;SEE HOW MANY CHARS LEFT
5364 017160 003315      BGT 4$      ;BR IF TOO MANY LEFT
5365                      ;CHECK FOR LEGAL PARAMETER VALUE
5366 017162 062702 005264 ADD #BUFF0,R2 ;GET ADDRESS OF DIGITS
5367 017166 010246      MOV R2,-(SP)  ;PUT IT ON STACK FOR OCTBIN
5368 017170 004737 054500 JSR PC,OCTBIN ;CHECK DIGITS AND CONVERT TO BINARY
5369 017174 017014      4$          ;ERROR RETURN ADDR. FOR OCTBIN
5370 017176 012637 005470 MOV (SP)+,LOWOCT ;GET LOW BINARY BITS
5371 017202 013737 054632 005472 MOV $HIOCT,HIGOCT ;GET HIGH BINARY BITS
5372 017210 004737 035040 JSR PC,CHKPRM  ;CHECK VALIDITY OF PARAMETER VALUE
5373 017214 017014      4$          ;ERROR RETURN ADDR. FOR CHKPRM
5374                      ;OPEN USER-DEFINED PATTERN IS FOR MODIFICATION, IF NECESSARY
5375 017216 004737 034514 JSR PC,MODP15
5376 017222 000662      BR SETPRM    ;RETURN TO ASK FOR ANOTHER PARAMETER
5377
5378
5379
5380
5381      ;SBTTL TO - RUN (R) TESTS ROUTINE
5382      ;*THIS ROUTINE IS ENTERED TO SET UP THE RUNNING OF TESTS,
5383      ;*WHOSE PARAMETERS HAVE BEEN DETERMINED. IT FIRST ASKS
5384      ;*FOR THE DESIRED NUMBER OF PROGRAM PASSES, AND STORES
5385      ;*THIS IN $EOPCT FOR THE END OF PASS ROUTINE. THEN, THE
5386      ;*DRIVE LIST IS SCANNED FOR THE FIRST DESIRED DRIVE, AND
5387      ;*THE HEADER ON SECTOR 0 IS READ TO GET THE FORMAT FOR THIS
5388      ;*DRIVE. THIS IS STORED IN THE FORMAT BYTE, AND A JUMP IS
5389      ;*MADE TO THE FIRST TEST.
5390      ;*THE END OF PASS ROUTINE RETURNS TO "NEWDRV" TO SELECT
5391      ;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
5392      ;* (ALL DRIVES) IS COMPLETED.
5393
5394      RUNTST:
5395      ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
5396      4$: TYPE ENTPAS      ;ASK FOR NUMBER OF PASSES
5397      JSR PC,RDCHRS      ;READ RESPONSE
5398      DRVTST
5399      ASKMDE
5400      4$
5401      TST R0
5402      BEQ 6$
5403      CMP #5,R0
5404      BGE 8$
5405      TYPE ,BUFF0
5406      BR 4$
5407      6$: TYPE $QUES
5408      BR 4$
5409      8$: MOV #BUFF0,-(SP) ;GO ASK AGAIN
5410      JSR PC,OCTBIN      ;GET BUF ADDR ON STACK FOR OCTBIN
5411      6$
5412      MOV (SP)+,$EOPCT    ;CHECK DIGITS AND CONVERT TO BINARY
5413      BEQ 6$             ;ERROR RETURN ADDRESS FOR OCTBIN
5414                      ;SET DESIRED NO. OF PASSES (1-77777)
5415                      ;BR IF 0, TO ASK AGAIN
5416      ;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
5417      STPASS: CLR $PASS   ;INIT. THE PASS NUMBER

```

```

5414 017312 112737 000001 003120      MOV      #1,TSTING      ;SET "RUNNING TESTS" FLAG
5415 017320 012737 177777 005500 NEWPAS: MOV      #177777,DPIVE    ;INITIALIZE DRIVE NO. TO -1
5416 017326 005037 001330      CLR      $DEVCT      ;INIT APT DEVICE COUNT TO 0
5417      ;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
5418 017332 012737 000000 177776 NEWDRV: MOV      #PRO,2#PS    ;RE-ESTABLISH PRIORITY 0
5419 017340 012706 001100      MOV      #STACK,SP      ;RESTORE THE STACK
5420 017344 105037 003130      CLR      DRVERS      ;CLEAR ERROR COUNT FOR CURRENT DRIVE
5421 017350 005037 005522      CLR      INTCHR      ;CLEAR TTY INPUT BUFFER WORD
5422 017354 005237 005500      INC      DRIVE      ;INCREMENT DRIVE NUMBER
5423 017360 022737 000010 005500      CMP      #10,DRIVE    ;SEE IF DONE WITH THIS PASS
5424 017366 001002      BNE      2$      ;PR IF NOT DONE CHECKING LIST
5425 017370 000137 023754      JMP      DUNPAS      ;JMP IF DONE WITH THIS PASS
5426 017374 013700 005500 2$:      MOV      DRIVE,RO      ;GET NEW DRIVE NUMBER
5427 017400 105760 005610      TSTB     DRVLS1(RO)    ;SEE IF THIS DRIVE IS MARKED IN LIST
5428 017404 001752      BEQ      NEWDRV      ;BR IF NOT MARKED, TO CHECK ANOTHER
5429 017406 105037 001102      CLR      $TSTNM      ;INIT TEST NUMBER TO 0
5430 017412 005037 001304      CLR      $TIMES      ;INIT ITERATION COUNT
5431 017416 105760 005620      TYPED: TSTB     DTYS1(RO)    ;SEE IF RK07
5432 017422 001036      BNE      1$      ;BR IF YES
5433 017424 105037 003124      CLR      TYPFMT      ;ELSE LOAD ALL RK06 PARAM
5434 017430 012737 000624 020036      MOV      #624,LCM6
5435 017436 012737 000631 005736      MOV      #631,PRMLIM+2
5436 017444 012737 000632 005662      MOV      #632,LC
5437 017452 012737 000632 020040      MOV      #632,LSTCYL
5438 017460 012737 000632 005710      MOV      #632,PRDFLT+2
5439 017466 012737 000632 005742      MOV      #632,PRMLIM+6
5440 017474 012737 000400 020042      MOV      #400,HOLD1
5441 017502 012737 000025 020044      MOV      #25,HOLD3
5442 017510 012737 010025 020046      MOV      #10025,HOLD4
5443 017516 000436      BR      3$
5444
5445 017520 152737 000004 003124 1$:      BISB     #8.CDT,TYPFMT    ;LOAD RK07 PARAMETERS
5446 017526 012737 001450 020036      MOV      #1450,LCM6
5447 017534 012737 001455 005736      MOV      #1455,PRMLIM+2
5448 017542 012737 001456 005662      MOV      #1456,LC
5449 017550 012737 001456 020040      MOV      #1456,LSTCYL
5450 017556 012737 001456 005710      MOV      #1456,PRDFLT+2
5451 017564 012737 001456 005742      MOV      #1456,PRMLIM+6
5452 017572 012737 001000 020042      MOV      #1000,HOLD1
5453 017600 012737 002025 020044      MOV      #2025,HOLD3
5454 017606 012737 012025 020046      MOV      #12025,HOLD4
5455
5456 017614      3$:      TSTB     HDALI      ;ARE WE DOING JUST HEAD-ALIGNMENT?
5457      BEQ      13$      ;YES, SKIP NEXT
5458      TSTB     HDALI+1    ;WHICH RETURN
5459      BMI      29$      ;SECOND RETURN
5460      JMP      2#INTGO    ;GO TO ALIGNMENT CODE
5461      29$:      JMP      2#INTGON  ;GO TO AUTO ALIGNMENT CODE
5462 017614 010037 001332 13$:      MOV      RO,$UNIT      ;SET DRV NO FOR APT
5463      ;READ THE HEADER ON SECTOR 0, TRACK 0, CYL 0, AND GET THE DRIVE FORMAT
5464      JSR      PC,INITSS    ;INIT. DRIVER PARAMS AND S.S.
5465 017620 004737 032310 000001      MOV      #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
5466 017624 112765 000125      JSR      PC,DRVCAL      ;DO READ HEADER
5467 017632 004737 042542      MOV      HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT = 0
5468 017636 013712 020044      4$:      BIT      #BIT7,(R2)    ;CHECK FOR C. READY IN CS1
5469 017642 032712 000200

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 110
TO - RUN (R) TESTS ROUTINE

SEQ 0109

5470	017646	001775		BEQ	4\$;BR IF NOT RDY YET
5471	017650	013712	020046	MOV	HOLD4,(R2)	;ISSUE RD HDR WITH FMT BIT = 1
5472	017654	032712	000200	BIT	#BIT7,(R2)	;CHECK FOR C. READY IN CS1
5473	017660	001775		BEQ	6\$;BR IF NOT RDY YET
5474	017662	105037	003125	CLRB	FORMAT	;INITIALIZE FORMAT BYTE TO 0
5475	017666	013737	005674	MOV	S2,FS	;INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
5476	017674	013737	005676	MOV	S3,LS	
5477	017702	005762	000024	TST	RKDB(R2)	;POP SILO ONE TIME
5478	017706	032762	001000	BIT	#BIT9,RKDB(R2)	;TEST FORMAT BIT IN HEADER WORD 2
5479	017714	001411		BEQ	8\$;BR IF 22 SECTOR FORMAT
5480	017716	152737	000020	BISB	#B.CFMT,FORMAT	;SET 20 SECTOR FORMAT FOR THIS DRIVE
5481	017724	013737	005670	MOV	S0,FS	;INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
5482	017732	013737	005672	MOV	S1,LS	
5483	017740	013737	005704	MOV	ST,STALLS	;SET NUMBER OF UNIT STALLS DESIRED
5484	017746	004737	032310	JSR	PC,INITSS	;INIT THE S.S.
5485	017752	004737	041120	JSR	PC,REDBSF	;READ BAD SECTOR FILE FOR THIS DRIVE
5486	017756	113737	005500	MOVB	DRIVE,DRVNO	;GET DRIVE NO.
5487	017764	152737	000060	BISB	#'0,DRVNO	;CONVERT TO ASCII
5488	017772	104401	011112	TYPE	TSTDRN	;TYPE "TESTING DRIVE X"
5489	017776	005737	001326	TST	\$PASS	;SEE IF THIS IS FIRST PASS
5490	020002	001012		BNE	10\$;BR IF NOT FIRST PASS
5491	020004	004737	035344	JSR	PC,DRVSR	;TYPE "DRIVE SER. NO. XXX"
5492	020010	004737	035464	JSR	PC,CRTSR	;TYPE "CART. SER. NO. XXXXXXXXXXXX"
5493	020014	032737	000020	BIT	#BIT4,CS	;SEE IF BAD SECTORS SHOULD BE TYPED
5494	020022	001402		BEQ	10\$;BR IF NOT
5495	020024	004737	041410	JSR	PC,TYPBSF	;TYPE BAD SECTOR FILES
5496	020030	005037	005522	CLR	INTCHR	;INIT. TTY INPUT CHAR BUFFER
5497	020034	000405		BR	TST1	;GO TO FIRST TEST
5498						
5499	020036	000000		LCM6:	0	
5500	020040	000000		LSTCYL:	0	
5501	020042	000000		HOLD1:	0	
5502	020044	000000		HOLD3:	0	
5503	020046	000000		HOLD4:	0	
5504						
5505						
5506						
5507						

5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540
5541
5542
5543
5544
5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557
5558
5559
5560
5561
5562
5563

020050	000004		
020052	012737	000001	001324
020060	004737	033672	
020064	004737	033740	
020070	000137	021176	
020074	004737	032310	
020100	004737	030576	
020104	105737	003116	
020110	001005		
020112	005037	005506	
020116	012737	000012	005510
020124	013765	005660	000002
020132	113765	005506	000004
020140	105065	000005	
020144	013737	005662	005504
020152	023737	020040	005662
020160	001002		
020162	005337	005504	
020166	012703	072307	
020172	004737	037614	
020176	012703	167230	
020202	005365	000002	
020206	100402		
020210	004737	037614	
020214	062765	000002	000002
020222	023765	020040	000002

```

*****
*TEST 1      OFFSET-TO-FAILURE MEASUREMENTS
*IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL
*WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC.
*THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH
* 167230(OCT) (TO ESTABLISH A KNOWN PATTERN) BUT THEY ARE NOT TESTED.
*THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE
*OFFSETS UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE
*OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE
*CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR
*MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS
*ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER: SECTORS FS AND LS ON
*CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR
*TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF
*INCREASING TRACKS, AS FOLLOWS:
*
*  OFFSET-TO-FAILURE MEASUREMENTS:
*
*  TRACK  CYLN  SECT  +OFST  -OFST
*          (UIN)  (UIN)
*   X     XXX   XX   XXXX   XXXX
*   X     XXX   XX   XXXX   XXXX
*   X     XXX   XX   XXXX   XXXX
*          ETC.
*
*NOTE: IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS
*TO 10(DEC), FOR THIS TEST.
*****
TST1:  SCOPE
      MOV     #1, $TESTN      ;SET TEST NUMBER IN APT MAIL BOX
      JSR     PC, SETUP      ;SET UP FOR LOOP ON ERROR
      JSR     PC, CHKTR      ;SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP     TST2          ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKTR IF TEST SHOULD BE RUN
      JSR     PC, INITSS     ;INITIALIZE DRIVER PARAMS AND SUB-SYS
      JSR     PC, PREPKB     ;PREPARE FOR POSSIBLE KBD INPUT
;INITIALIZE PARAMETERS, WRITE THE SECTORS, AND WRITE CHECK THEM
      TSTB    MDFLAG        ;SEE IF ADRS 200 DEFAULT RUN
      BNE     3$            ;BR IF NOT
      CLR     FS            ;SET FS = 0
      MOV     #10, LS       ;SET LS = 10(DEC)
      MOV     FC, P.CYLN(R5) ;SET CYL = FC
      MOV     FS, P.SECT(R5) ;SET SECTOR = FS
      CLRB    P.TRCK(R5)    ;SET TRACK = 0
      MOV     LC, CYLNDR
      CMP     LSTCYL, LC    ;SEE IF LC = 632/1456
      BNE     4$            ;BR IF NOT 632/1456
      DEC     CYLNDR        ;MAKE IT 631 (PRESERVE BSF)
      MOV     #72307, R3    ;GET DATA PATTERN
      JSR     PC, WRTSEC     ;DO WRITE AND WRITE CHECK
      MOV     #167230, R3   ;DATA FOR ADJACENT CYLS
      DEC     P.CYLN(R5)    ;NEXT LOWER CYL
      BMI     6$            ;BR IF CYL NEGATIVE
      JSR     PC, WRTSEC     ;DO WRITE AND WRITE CHECK
      ADD     #2, P.CYLN(R5) ;NEXT HIGHER CYL
      CMP     LSTCYL, P.CYLN(R5) ;SEE IF CYL = 632/1456

```



```

5564 020230 001004      BNE      8$      ;BR IF NOT
5565 020232 122765 000002 000005      CMPB     #2,P.TRCK(R5) ;SEE IF BSF
5566 020240 001402      BEQ      10$     ;BR IF YES, TO PROTECT IT
5567 020242 004737      JSR      PC,WRTSEC ;DO WRITE AND WRITE CHECK
5568 020246 005365 000002      DEC      P.CYLN(R5) ;RESTORE CYL NO.
5569 020252 105265 000005      INCB     P.TRCK(R5) ;INCR TRACK
5570 020256 122765 000003 000005      CMPB     #3,P.TRCK(R5) ;TRACK = 3 YET ?
5571 020264 001340      BNE      4$      ;BR IF NOT YET
5572 020266 126537 000004 005510      CMPB     P.SECT(R5),LS ;SEE IF SECTOR = LS
5573 020274 001406      BEQ      14$     ;BR IF YES
5574 020276 113765 005510 000004      MOVB     LS,P.SECT(R5) ;SET SECTOR = LS
5575 020304 105065 000005      CLRB     P.TRCK(R5) ;SET TRACK = 0
5576 020310 000726      BR       4$      ;BR TO CONTINUE WRITING
5577 020312 026537 000002 005504 14$:  CMP      P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
5578 020320 001407      BEQ      16$     ;BR IF YES
5579 020322 013765 005504 000002      MOV      CYLNDR,P.CYLN(R5) ;SET CYL = LC
5580 020330 113765 005506 000004      MOVB     FS,P.SECT(R5) ;SET SECTOR = FS AGAIN
5581 020336 000762      BR       12$     ;BR TO CONTINUE WRITING
5582      ;INITIALIZE PARAMETERS FOR OFFSET MEASUREMENTS
5583 020340 032737 000002 005702 16$:  BIT      #BIT1,CS ;SEE IF REPORTS ARE INHIBITED
5584 020346 001002      BNE      18$     ;BR IF INHIBITED
5585 020350 104401 011240      TYPE     OFSHED ;TYPE HEADINGS
5586 020354 113765 005664 000005 18$:  MOVB     FT,P.TRCK(R5) ;SET TRACK = FT
5587 020362 013765 005660 000002      MOV      FC,P.CYLN(R5) ;SET CYLINDER = FC
5588 020370 113765 005506 000004      MOVB     FS,P.SECT(R5) ;SET SECTOR = FS
5589      ;LOAD THE R/W BUFFER FOR WRITE CHECKS
5590 020376 012700 065660      MOV      #RWBUF,R0 ;SET BUFFER ADDRESS
5591 020402 012701 000400      MOV      #400,R1 ;PREPARE TO LOAD 400(OCT) WORDS
5592 020406 012720 072307 19$:  MOV      #072307,(R0)+ ;LOAD A WORD INTO BUFFER
5593 020412 005301      DEC      R1 ;DECR COUNTER
5594 020414 001374      BNE      19$     ;BR IF NOT DONE YET
5595 020416 005004 20$:  CLR      R4 ;INIT (+) OFFSET VALUE
5596 020420 005000      CLRB     R0 ;INIT OFFSET NUMBER TO 0
5597 020422 012737 177777 005546      MOV      #-1,PLOFST ;INITIALIZE CURRENT FAILING (+) OFFSET
5598 020430 012737 177777 005550      MOV      #-1,NGOFST ;INITIALIZE CURRENT FAILING (-) OFFSET
5599      ;PERFORM OFFSETS AND WRITE CHECKS
5600 020436 112765 000117 000001 22$:  MOVB     #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5601 020444 004737 042542      JSR      PC,DRVCAL ;PERFORM SEEK
5602 020450 110065 000006      MOVB     R0,P.OFST(R5) ;SET CURRENT OFFSET
5603 020454 112765 000115 000001      MOVB     #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
5604 020462 004737 042542      JSR      PC,DRVCAL ;PERFORM OFFSET
5605 020466 112765 000131 000001      MOVB     #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
5606 020474 012737 021142 003046      MOV      #WRCKHD,A.ABNL ;SET SPECIAL ERROR HANDLER ADDRESS
5607 020502 105037 003140      CLRB     WCEFLG ;CLEAR WRITE CHECK ERROR FLAG
5608 020506 004737 042542      JSR      PC,DRVCAL ;PERFORM A WRITE CHECK
5609 020512 012737 044200 003046      MOV      #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADDRESS
5610 020520 112765 000177 000001      MOVB     #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR CMND
5611 020526 004737 042542      JSR      PC,DRVCAL ;CLEAR THE SUBSYSTEM
5612 020532 112765 000115 000001      MOVB     #OFFSET,P.CMND(R5) ;SET OFFSET CMND
5613 020540 105065 000006      CLRB     P.OFST(R5) ;SET OFFSET = 0
5614 020544 004737 042542      JSR      PC,DRVCAL ;OFFSET BACK TO 0
5615 020550 105737 003140      TSTB     WCEFLG ;SEE IF A WRITE CHECK ERROR OCCURRED
5616 020554 001020      BNE      26$     ;BR IF AN ERROR OCCURRED
5617 020556 010001      MOV      R0,R1 ;GET A COPY OF CURRENT OFFSET
5618 020560 005200      INC      R0 ;INCREMENT OFFSET
5619 020562 062704 000031      ADD      #25.,R4 ;INCR OFFSET VALUE BY 25(DEC) UIN

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 113
T1 OFFSET-TO-FAILURE MEASUREMENTS

SEQ 0112

5620	020566	042701	177700		BIC	#177700,R1	;MASK FOR MAGNITUDE BITS
5621	020572	022701	000060		CMP	#60,R1	;SEE IF MAX OFFSET APPLIED IN THIS DIRECTION
5622	020576	001317			BNE	22\$;BR IF NOT YET
5623	020600	032700	000200		BIT	#BIT7,R0	;SEE IF NEG OFFSETS DONE YET
5624	020604	001015			BNE	32\$;BR IF YES, TO PRINT CURRENT SECTOR RESULT
5625	020606	012700	000200	24\$:	MOV	#200,R0	;INIT FOR NEG OFFSETS
5626	020612	005004			CLR	R4	;INIT (-) OFFSET VALUE TO 0
5627	020614	000710			BR	22\$;BR TO APPLY NEG OFFSETS
5628	020616	032700	000200	26\$:	BIT	#BIT7,R0	;SEE IF NEG OFFSETS DONE YET
5629	020622	001403			BEQ	28\$;BR IF NOT YET
5630	020624	010437	005550		MOV	R4,NGOFST	;SAVE THIS FAILING (-) OFFSET VALUE
5631	020630	000403			BR	32\$;BR TO PRINT CURRENT SECTOR RESULT
5632	020632	010437	005546	26\$:	MOV	R4,PLOFST	;SAVE THIS FAILING (+) OFFSET VALUE
5633	020636	000763			BR	24\$;BR TO APPLY NEGATIVE OFFSETS
5634	020640	032737	000002	005702 32\$:	BIT	#BIT1,CS	;SEE IF REPORTS INHIBITED
5635	020646	001051			BNE	43\$;BR IF INHIBITED
5636					;TYPE OFFSET RESULTS FOR THIS SECTOR		
5637	020650	116501	000005		MOVB	P.TRCK(R5),R1	;GET TRACK NO.
5638	020654	010146			MOV	R1,-(SP)	;PUT IT ON STACK
5639	020656	104403			TYPDS		;TYPE IT
5640	020660	003			.BYTE	3	;3 DIGITS
5641	020661	000			.BYTE	0	;SUPPRESS
5642	020662	104401	013312		TYPE	SPACE3	
5643	020666	016546	000002		MOV	P.CYLN(R5),-(SP)	;PUT CYL ON STACK
5644	020672	104403			TYPDS		;TYPE IT
5645	020674	004			.BYTE	4	;4 DIGITS
5646	020675	000			.BYTE	0	;SUPPRESS
5647	020676	104401	013313		TYPE	SPACE2	
5648	020702	116501	000004		MOVB	P.SECT(R5),R1	;GET SECTOR
5649	020706	010146			MOV	R1,-(SP)	;PUT IT ON STACK
5650	020710	104404			TYPON		;TYPE SECTOR
5651	020712	104401	013314		TYPE	SPACE1	
5652	020716	005737	005546		TST	PLOFST	;SEE IF THERE WAS A FAILING (+) OFFSET
5653	020722	100003			BPL	36\$;BR IF THERE WAS
5654	020724	104401	011231		TYPE	NOFALS	;TYPE " NONE"
5655	020730	000403			BR	38\$;CONTINUE
5656	020732	013746	005546	36\$:	MOV	PLOFST,-(SP)	;PUT (+) OFFSET VALUE ON STACK
5657	020736	104405			TYPDS		;TYPE IT IN DECIMAL
5658	020740	104401	013314	38\$:	TYPE	SPACE1	
5659	020744	005737	005550		TST	NGOFST	;SEE IF THERE WAS A FAILING (-) OFFSET
5660	020750	100003			BPL	40\$;BR IF THERE WAS
5661	020752	104401	011231		TYPE	NOFALS	;TYPE " NONE"
5662	020756	000403			BR	42\$;CONTINUE
5663	020760	013746	005550	40\$:	MOV	NGOFST,-(SP)	;PUT (-) OFFSET VALUE ON STACK
5664	020764	104405			TYPDS		;TYPE IT IN DECIMAL
5665	020766	104401	001315	42\$:	TYPE	SCRLF	;TYPE <CR>,<LF>
5666					;SET UP TO TEST NEXT SECTOR		
5667	020772	126537	000004	005510 43\$:	CMPB	P.SECT(R5),LS	;SEE IF SECTOR = LS
5668	021000	001404			BEQ	44\$;BR IF YES
5669	021002	113765	005510	000004	MOVB	LS,P.SECT(R5)	;SET SECTOR = LS
5670	021010	000602			BR	20\$;GO TEST THIS SECTOR
5671	021012	026537	000002	005504 44\$:	CMP	P.CYLN(R5),CYLNDR	;SEE IF CYL = LC
5672	021020	001410			BEQ	48\$;BR IF YES
5673	021022	013765	005504	000002	MOV	CYLNDR,P.CYLN(R5)	;SET CYL = LC
5674	021030	113765	005506	000004 46\$:	MOVB	FS,P.SECT(R5)	;SET SECTOR = FS
5675	021036	000137	020416		JMP	20\$;GO TEST THIS SECTOR

```

5676 021042 105265 000005      48$: INCB P.TRCK(R5) ; INCREMENT TRACK
5677 021046 122765 000003 000005 CMPB #3,P.TRCK(R5) ; SEE IF TRACK = 3
5678 021054 001404 BEQ 50$ ; BR IF YES
5679 021056 013765 005660 000002 MOV FC,P.CYLN(R5) ; SET CYL = FC
5680 021064 000761 BR 46$ ; GO TEST THIS SECTOR
5681 021066 112765 000113 000001 50$: MOVB #PECAL,P.CMND(R5) ; SET RECAL COMMAND
5682 021074 004737 042542 JSR PC,DRVCL ; DO CLEANUP RECALIBRATE
5683 021100 013737 005674 005506 MOV S2,FS ; RESTORE FS,LS FOR 22 SECTORS
5684 021106 013737 005676 005510 MOV S3,LS
5685 021114 105737 003125 TSTB FORMAT ; SEE IF 22 SECTORS
5686 021120 001406 BEQ 54$ ; BR IF YES
5687 021122 013737 005670 005506 MOV S0,FS ; RESTORE FS,LS FOR 20 SECTORS
5688 021130 013737 005672 005510 MOV S1,LS
5689 021136 54$: JMP TST2 ; JUMP TO NEXT TEST
5690 021136 000137 021176
5691
5692 ; *THIS IS THE ABNORMAL RETURN FROM THE DRIVER, USED WHEN A
5693 ; * "DATA TYPE" ERROR IS EXPECTED (AND VALID).
5694 021142 032765 040000 000020 WRCKHD: BIT #WCE,P.CS2(R5) ; SEE IF WRITE CHECK ERROR OCCURRED
5695 021150 001006 BNE 4$ ; BR IF WCE
5696 ; CHECK FOR OTHER "DATA TYPE" ERROR
5697 021152 032765 130400 000034 BIT #HVRC!DTE!OPI!DCK,P.ER(R5)
5698 021160 001002 BNE 4$ ; BR IF A DATA ERROR OCCURRED
5699 021162 000137 044200 JMP ERRHDL ; GO HANDLE OTHER ERROR
5700 021166 105237 003140 4$: INCB WCEFLG ; SET WRITE CHECK ERROR FLAG
5701 021172 000137 046356 JMP RETNML ; TAKE NORMAL RETURN
5702
5703
5704
5705 ; *****
5706 ; *TEST 2 NPR/MEMORY WORD ADDRESSING TEST
5707 ; *IN THIS TEST, MEMORY WORD ADDRESSING CAPABILITY DURING RK06 NPR'S,
5708 ; *IS TESTED.
5709 ; *BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
5710 ; *THEN, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK
5711 ; *BEYOND ADDRESS RWBUF+6000(OCTAL), WRITE UNIQUE NUMBERS
5712 ; *(STARTING WITH 1), INTO EACH OF UP TO 64K WORDS (DEPENDING UPON THE
5713 ; *AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES
5714 ; *WITHIN THE 64K BLOCK. NEXT, WRITE THE 64K WORDS (MAX) ONTO THE DISK
5715 ; *AT ADDRESS FC,FT,FS (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW
5716 ; *OR DESTRUCTION OF THE BAD SECTOR FILE). THEN ZERO THE ENTIRE BLOCK
5717 ; *IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL
5718 ; *ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING
5719 ; *VIRTUAL (IF MEM. MGT.) AS WELL AS PHYSICAL ADDRESSES, AND THE GOOD
5720 ; *AND BAD DATA, FOR UP TO THE FIRST 10(DEC) FAILING LOCATIONS.
5721 ; * IF MEMORY MANAGEMENT IS INSTALLED AND THERE IS ADDITIONAL MEMORY TO
5722 ; *EXERCISE, REPEAT THE ABOVE WORD ADDRESSING TEST, FOR EACH OF THE
5723 ; *REMAINING 64K MEMORY BLOCKS.
5724 ; *****
5725 021176 000004 000002 001324 1ST2: SCOPE
5726 021200 012737 MOV #2,$TESTN ; SET TEST NUMBER IN APT MAIL BOX
5727 021206 004737 033672 JSR PC,SETUP ; SET UP FOR LOOP ON ERROR
5728 021212 004737 033740 JSR PC,CHKITR ; SEE IF ITER. NO. = 0 FOR THIS TEST
5729 021216 000137 021506 JMP TST3 ; JUMP TO NEXT TEST
5730 ; RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5731 021222 004737 032310 JSR PC,INITSS ; INITIALIZE DRIVER PARAMS AND SUB-SYS

```

```

5732 021226 004737 030576      JSR      PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT
5733      ;SAVE XXDP LOADER, IF PRESENT
5734 021232 004737 031770      JSR      PC,FNDXDP      ;COMPUTE STARTING ADR OF XXDP
5735 021236 012737 065660 005542  MOV      #RWBUFF,XDPSAV ;SET XXDP SAVE AREA ADR
5736 021244 005037 005544      CLR      XDPSAV+2
5737 021250 004737 032042      JSR      PC,SAVXDP      ;GO SAVE XXDP LOADER
5738 021254 112737 000001 003133  MOV      #1,XOVLAD      ;SET INDICATORS
5739 021262 112737 000001 003134  MOV      #1,XDPSVD
5740 021270 012737 177777 005536  MOV      #-1,PATRN      ;SET FLAG FOR SVPRMS
5741      ;INIT PARAMETERS, BEGIN TESTING
5742 021276 004737 040644      JSR      PC,SETUPM      ;INIT PARAMS
5743 021302 004737 035270 4$:      JSR      PC,MXWRDC      ;COMPUTE MAX WORD COUNT FOR THIS MA
5744 021306 005701      TST      R1      ;SEE IF ALL MEMORY TESTED YET
5745 021310 100467      BMI      50$      ;BR IF ALL DONE
5746 021312 005065 000012      CLR      P.WC(R5)      ;INIT WORD COUNT TO 65,536(DEC)
5747 021316 005701      TST      R1      ;SEE IF WRD CNT SHOULD BE 65,536
5748 021320 001003      BNE      6$      ;BR IF YES
5749 021322 005400      NEG      R0
5750 021324 010065 000012      MOV      R0,P.WC(R5)      ;SET WORD COUNT
5751 021330 013765 005600 000010 6$:      MOV      MA,P.BALO(R5)      ;SET BA BITS 0-15
5752 021336 013701 005602      MOV      MA+2,R1      ;GET MA BITS 16-21
5753 021342 042701 177774      BIC      #177774,R1      ;MASK FOR LO 2 BITS
5754 021346 150165 000007      BISB     R1,P.BAHI(R5)      ;SET BA BITS 16,17
5755 021352 005737 057732      TST      $K+11      ;SEE IF MEM MGT PRESENT
5756 021356 100010      BPL      14$      ;BR IF NOT
5757      ;SET UP MEM MGT
5758 021360 013737 005600 005604  MOV      MA,PMA      ;SET BUFFER ADDRESS
5759 021366 013737 005602 005606  MOV      MA+2,PMA+2
5760 021374 004737 031646      JSR      PC,PREPAR      ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5761      ;PERFORM THE TESTS
5762 021400 012737 021400 001110 14$:      MOV      #,$LPERR      ;SET NEW LOOP ON ERROR ADRS
5763 021406 004737 033672      JSR      PC,SETUP      ;SET UP FOR LOOP ON ERROR
5764 021412 004737 042052      JSR      PC,SVPRMS      ;SAVE PARAMS FOR THIS XFER
5765 021416 004737 037710      JSR      PC,LDMEM1      ;LOAD THIS MEM BLK WITH INCREMENTING DATA
5766 021422 112765 000123 000001  MOV      #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5767 021430 004737 042164      JSR      PC,TRANSFR      ;WRITE THE DATA
5768 021434 005003      CLR      R3      ;SET DATA WORD = 0
5769 021436 004737 037722      JSR      PC,LDMEM2      ;LOAD MEM BLK WITH ALL ZEROS
5770 021442 112765 000121 000001  MOV      #RDATA,P.CMND(R5) ;SET READ COMMAND
5771 021450 004737 042164      JSR      PC,TRANSFR      ;READ THE DATA FROM DISK
5772 021454 004737 040054      JSR      PC,CKMEM1      ;PERFORM SOFTWARE COMPARE OF DATA
5773 021460 104411      SCOPER      ;CHECK FOR INTERNAL LOOP ON ERROR
5774      ;GET NEW MA FOR NEXT TRANSFER
5775 021462 004737 041052      JSR      PC,INCRMA      ;COMPUTE NEXT MA
5776 021466 000705      BR      4$      ;GO SEE IF MORE MEMORY TO TEST
5777 021470 004737 032050 50$:      JSR      PC,GETXDP      ;RESTORE XXDP LOADER
5778 021474 105737 003143      TSTB     UBMPRS      ;SEE IF UNIBUS MAP PRESENT
5779 021500 001402      BEQ      54$      ;BR IF NOT
5780 021502 005037 172516      CLR      @#SR3      ;DISABLE UNIBUS MAP
5781 021506
5782
5783
5784
5785      ;*****
5786      ;*TEST 3      NPR/MEMORY BLOCK ADDRESSING TEST
5787      ;*IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06

```

```
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802 021506 000004
5803 021510 012737 000003 001324
5804 021516 004737 033672
5805 021522 004737 033740
5806 021526 000137 022046
5807
5808 021532 004737 032310
5809 021536 004737 030576
5810
5811 021542 004737 031770
5812 021546 012737 065660 005542
5813 021554 005037 005544
5814 021560 004737 032042
5815 021564 112737 000001 003133
5816 021572 112737 000001 003134
5817 021600 012737 177777 005536
5818
5819 021606 005037 005524
5820 021612 004737 040644
5821 021616 012704 000001
5822 021622 004737 035270
5823 021626 005701
5824 021630 100473
5825 021632 005065 000012
5826 021636 005701
5827 021640 001003
5828 021642 005400
5829 021644 010065 000012
5830 021650 013765 005600 000010
5831 021656 013701 005602
5832 021662 042701 177774
5833 021666 150165 000007
5834 021672 005737 057732
5835 021676 100010
5836
5837 021700 013737 005600 005604
5838 021706 013737 005602 005606
5839 021714 004737 031646
5840 021720 005737 005524
5841 021724 001026
5842
5843 021726 004737 042052
```

```

; *NPR'S IS TESTED.
; *BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
; *MEMORY WILL BE TESTED STARTING AT THE FIRST ADRS OF THE NEXT 1K
; *BLOCK BEYOND RWBUF+6000(OCTAL).
; *THE TEST BEGINS BY WRITING A SINGLE NUMBER INTO ALL LOCATIONS OF
; *EACH MEMORY BLOCK OF UP TO 64K WORDS. THIS NUMBER UNIQUELY DEFINES
; *EACH BLOCK. THEN WRITE EACH BLOCK ONTO DISK AT ADDRESS FC FS FT
; *(SCALED, IF NECESSARY TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
; *SECTOR FILE). ZERO THE BLOCK IN MEMORY, AND THEN READ THE DATA
; *BACK INTO THE PROPER PHYSICAL ADDRESSES. DO THIS FOR ALL 64K BLOCKS,
; *AND THEN COMPARE THE CONTENTS OF ALL THE MEMORY TO EXPECTED VALUES.
; *TYPE FAILURES AS DESCRIBED ABOVE FOR THE FIRST 10(DEC) FAILING
; *LOCATIONS IN EACH 64K MEMORY BLOCK.
; *****
TST3: SCOPE
      MOV #3, $TESTN ; SET TEST NUMBER IN APT MAIL BOX
      JSR PC, SETUP ; SET UP FOR LOOP ON ERROR
      JSR PC, CHKTR ; SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP TST4 ; JUMP TO NEXT TEST
; RETURN HERE FROM CHKTR IF TEST SHOULD BE RUN
      JSR PC, INITSS ; INITIALIZE DRIVER PARAMS AND SUB-SYS
      JSR PC, PREPKB ; PREPARE FOR POSSIBLE KBD INPUT
; SAVE XXDP LOADER, IF PRESENT
      JSR PC, FNDXDP ; COMPUTE STARTING ADRS OF XXDP
      MOV #RWBUF, XDPSAV ; SET XXDP SAVE AREA ADDRESS
      CLR XDPSAV+2
      JSR PC, SAVXDP ; GO SAVE XXDP LOADER
      MOV #1, XOVLD ; SET INDICATORS
      MOV #1, XDPSVD
      MOV #-1, PATRN ; SET FLAG FOR SVPRMS
; INIT PARAMETERS, WRITE AND READ
      CLR SELECT ; BACK ALL MEMORY BLOCKS
      JSR PC, SETUPM ; INIT COMPARE FLAG
      MOV #1, R4 ; INIT PARAMS
      JSR PC, MXWRDC ; INIT DATA WORD
      TST R1 ; COMPUTE MAX WRD CNT FOR THIS MA
      BMI 30$ ; SEE IF ALL MEMORY TESTED YET
      CLR P.WC(R5) ; BR IF DONE WRITING AND READING
      TST R1 ; INIT WRD CNT TO 65,536(DEC)
      BNE 6$ ; SEE IF WRD CNT SHOULD BE 65536
      NEG R0 ; BR IF YES
      MOV R0, P.WC(R5) ; SET WORD COUNT
      MOV MA, P.BALO(R5) ; SET BA BITS 0-15
      MOV MA+2, R1 ; GET MA BITS 16-21
      BIC #177774, R1 ; MASK FOR LO 2 BITS
      BISB R1, P.BAHI(R5) ; SET BA BITS 16,17
      TST $K11 ; SEE IF MEM MGT PRESENT
      BPL 14$ ; BR IF NO
; SET UP MEMORY MANAGEMENT
      MOV MA, PMA ; SET BUFFER ADDRESS
      MOV MA+2, PMA+2
      JSR PC, PREPAR ; PREPARE MEM MGT REG'S AND UNIBUS MAP
      TST SELECT ; SEE IF COMPARES SHOULD BE DONE YET
      BNE 20$ ; BR IF YES
; WRITE AND READ THIS MEM BLK ON RK06
      JSR PC, SVPRMS ; SAVE PARAMS FOR THIS XFER
```

```

5844 021732 010403      MOV      R4,R3      ;GET DATA WORD
5845 021734 004737 037722 JSR      PC,LDMEM2    ;LOAD MEMORY BLK WITH THIS WORD
5846 021740 112765 000123 000001 MOVB     #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
5847 021746 004737 042164 JSR      PC,TRNSFR    ;WRITE THE DATA
5848 021752 005003      CLR      R3
5849 021754 004737 037722 JSR      PC,LDMEM2    ;ZERO THE BLK IN MEMORY
5850 021760 112765 000121 000001 MOVB     #RDATA,P.CMND(R5) ;SET READ COMMAND
5851 021766 004737 042164 JSR      PC,TRNSFR    ;READ THIS DATA BACK
5852 021772 005204      INC      R4      ;INCREMENT THE DATA WORD
5853 021774 004737 041052 JSR      PC,INCRMA    ;COMPUTE NEW MA TO TRY
5854 022000 000710      BR       4$      ;GO WRITE AND READ NEXT BLK
5855      ;PERFORM SOFTWARE COMPARES OF ALL MEMORY BLOCKS
5856 022002 010403 20$: MOV      R4,R3      ;GET DATA WORD
5857 022004 004737 040066 JSR      PC,CKMEM2    ;COMPARE THIS BLK TO GOOD DATA WORD
5858 022010 005204      INC      R4      ;INCREMENT THE DATA WORD
5859 022012 004737 041052 JSR      PC,INCRMA    ;GET NEW MA TO TRY
5860 022016 000701      BR       4$      ;GO COMPARE NEXT MEM BLK
5861
5862 022020 005137 005524 30$: COM      SELECT    ;COMPLEMENT THE FLAG
5863 022024 001401      BEQ      50$      ;BR IF ALL DONE NOW
5864 022026 000671      BR       2$      ;BR IF COMPARES SHOULD BE DONE NOW
5865 022030 004737 032050 50$: JSR      PC,GETXDP    ;RESTORE SAVED XXDP
5866 022034 105737 003143 TSTB     UBMPRS      ;SEE IF UNIBUS MAP PRESENT
5867 022040 001402      BEQ      54$      ;BR IF NOT
5868 022042 005037 172516 CLR       0#SR3      ;DISABLE UNIBUS MAP
5869 022046
5870
5871
5872
5873 *****
5874 *TEST 4      NPR/MEMORY DATA PATTERN TEST
5875 *BEFORE TESTING, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS
5876 *RWBUFF. THEN, ALL PHYSICAL MEMORY LOCATIONS, STARTING AT THE
5877 *FIRST ADRS OF THE NEXT 1K MEMORY BLOCK BEYOND RWBUF+6000(OCT),
5878 *ARE EXERCISED WITH UP TO 16 DATA PATTERNS
5879 *CHOSEN IN PARAMETER PT. (THESE ARE THE SAME PATTERNS
5880 *PROVIDED FOR USE IN THE READ/WRITE DATA TEST). IF PT = 0, THE
5881 *DATA PATTERNS WILL DEFAULT TO PATS 08,09,10, AND 11.
5882 *MEMORY IS EXERCISED IN BLOCKS OF UP TO 64K WORDS, STARTING AT
5883 *THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE BUFFER.
5884 *EACH BLOCK IS LOADED WITH DATA, AND WRITTEN ONTO DISK AT ADDRESS
5885 *FC FT FS (SCALED TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
5886 *SECTOR FILE). THEN THE MEMORY BLOCK IS ZEROED AND READ BACK AND
5887 *COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN,
5888 *POSSIBLY INCLUDING THE USER-DEFINED PATTERN 15, IF SPECIFIED.
5889 *****
5890 022046 000004 5891 022050 012737 000004 001324 5892 022056 004737 033672
5893 022062 004737 033740
5894 022066 000137 022416
5895      ;RETURN HERE FROM CHKITA IF TEST SHOULD BE RUN
5896 022072 004737 032310 JSR      PC,INITSS    ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5897 022076 004737 030576 JSR      PC,PREPKB    ;PREPARE FOR POSSIBLE KBD INPUT
5898
5899 022102 004737 031770 JSR      PC,FNDXDP    ;COMPUTE STARTING ADR OF XXDP

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A('052) 10-JAN-78 12:02 PAGE 118
T4 NPR/MEMORY DATA PATTERN TEST

SEQ 0117

5900	022106	012737	065660	005542	MOV	#RWBUF,XDPSAV	;SET XXDP SAVE AREA ADR
5901	022114	005037	005544		CLR	XDPSAV+2	
5902	022120	004737	032042		JSR	PC,SAVXDP	;GO SAVE XXDP LOADER
5903	022124	112737	000001	003133	MOV	#1,XOVLAD	;SET INDICATORS
5904	022132	112737	000001	003134	MOV	#1,XDPSVD	
5905	022140	004737	036034		JSR	PC,LODP14	;GENERATE PSEUDO-RAND PAT 14
5906	022144	013737	005700	005536	MOV	PT,PATRN	;GET A COPY OF PT
5907	022152	001003			BNE	2\$;BR IF PT NON-ZERO
5908	022154	012737	007400	005536	MOV	#7400,PATRN	;SET DEFLT PATRNS = PAT 8,9,10,11
5909					;INIT PARAMETERS BEGIN TESTING		
5910	022162	004737	040644		2\$: JSR	PC,SETUPM	;INIT PARAMS
5911	022166	004737	035270		4\$: JSR	PC,MXWRDC	;COMPUTE MAX WORD COUNT FOR THIS MA
5912	022172	005701			TST	R1	;SEE IF ALL MEMORY TESTED YET
5913	022174	100501			BMI	50\$;BR IF ALL DONE
5914	022176	005065	000012		CLR	P.WC(R5)	;INIT WORD COUNT TO 65,536(DEC)
5915	022202	005701			TST	R1	;SEE IF WRD CNT SHOULD BE 65,536
5916	022204	001003			BNE	6\$;BR IF YES
5917	022206	005400			NEG	R0	
5918	022210	010065	000012		MOV	R0,P.WC(R5)	;SET WORD COUNT
5919	022214	013765	005600	000010	6\$: MOV	MA,P.BALO(R5)	;SET BA BITS 0-15
5920	022222	013701	005602		MOV	MA+2,R1	;GET MA BITS 16-21
5921	022226	042701	177774		BIC	#177774,R1	;MASK FOR LO 2 BITS
5922	022232	150165	000007		BISB	R1,P.BAHI(R5)	;SET BA BITS 16,17
5923	022236	005737	057732		TST	\$K+11	;SEE IF MEM MGT PRESENT
5924	022242	100010			BPL	14\$;BR IF NOT
5925					;SET UP MEM MGT		

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 119
T4 NPR/MEMORY DATA PATTERN TEST

SEG 0118

5926	022244	013737	005600	005604	MOV	MA,PMA	;SET BUFFER ADDRESS
5927	022252	013737	005602	005606	MOV	MA+2,PMA+2	
5928	022260	004737	031646		JSR	PC,PREPAR	;PREPARE MEM MGT REG'S AND UNIBUS MAP
5929	022264	012704	000001	14\$:	MOV	#1,R4	;SET PATTERN BIT POINTER
5930	022270	030437	005536	16\$:	BIT	R4,PATRN	;SEE IF THIS PATTERN IS CHOSEN
5931	022274	001003			BNE	20\$;BR IF YES
5932	022276	006304		18\$:	ASL	R4	;SHIFT TO POINT TO NEXT PATTERN
5933	022300	001434			BEQ	24\$;BR IF NO MORE LEFT TO CHECK
5934	022302	000772			BR	16\$;GO CHECK FOR ANOTHER PATTERN
5935	022304	010401		20\$:	MOV	R4,R1	;GET A COPY OF BIT POINTER
5936							
5937	022306	012737	022306	001110	22\$:	MOV	#,\$LPERR
5938	022314	004737	033672		JSR	PC,SETUP	;SET NEW LOOP ON ERROR ADRS
5939	022320	004737	042052		JSR	PC,SVPRMS	;SET UP FOR LOOP ON ERROR
5940	022324	004737	036250		JSR	PC,LOADBUF	;SAVE THE PARAMS FOR THIS XFER
5941	022330	112765	000123	000001	MOV	#WADATA,P.CMND(R5)	;LOAD THIS MEM BLK WITH SELECTED DATA
5942	022336	004737	042164		JSR	PC,TRANSFR	;SET WRITE DATA COMMAND
5943	022342	005003			CLR	R3	;WRITE THE DATA
5944	022344	004737	037722		JSR	PC,LDMEM2	;SET DATA WORD = 0
5945	022350	112765	000121	000001	MOV	#RADATA,P.CMND(R5)	;LOAD MEM BLK WITH ALL ZEROS
5946	022356	004737	042164		JSR	PC,TRANSFR	;SET READ COMMAND
5947	022362	004737	036476		JSR	PC,CMPBUF	;READ THE DATA FROM DISK
5948	022366	104411			SCOPER		;PERFORM SOFTWARE COMPARE OF DATA
5949	022370	000742			BR	18\$;CHECK FOR INTERNAL LOOP ON ERROR
5950							;CHECK FOR NEXT PATTERN
5951	022372	004737	041052		24\$:	JSR	PC,INCRMA
5952	022376	000673			BR	4\$;COMPUTE NEXT MA
5953	022400	004737	032050		50\$:	JSR	PC,GETXDP
5954	022404	105737	003143		TSTB	UBMPRS	;GO SEE IF MORE MEMORY TO TEST
5955	022410	001402			BEQ	54\$;RESTORE XXDP LOADER
5956	022412	005037	172516		CLR	#SR3	;SEE IF UNIBUS MAP PRESENT
5957	022416						;BR IF NOT PRESENT
5958							;DISABLE UNIBUS MAP
5959							
5960							
5961							
5962							
5963							
5964							
5965							
5966							
5967							
5968							
5969							
5970							
5971							
5972							
5973							
5974							
5975							
5976							
5977							
5978							
5979							
5980							
5981							

```

*****
*TEST 5 UNIBUS CONTENTION TEST
*THE PURPOSE OF THIS TEST IS TO EXERCISE THE NPR CONTROL LOGIC
*WITHIN THE CONTROLLER IN SUCH A MANNER THAT A CONTENTION FOR
*UNIBUS CYCLES ARISES BETWEEN THE PROCESSOR AND THE RK06.
*
*FIRST, A WRITE DATA IS BEGUN AT ADDRESS FC,FT,SECTOR 0, WITH
*THE BUS ADDRESS INCREMENT INHIBIT BIT (BAI) SET IN THE CONTROLLER.
*USING A WORD COUNT OF 12000(OCT) OR 13000(OCT) (DEPENDING ON THE
*FORMAT), THE ENTIRE TRACK IS WRITTEN WITH THE FIRST WORD OF PATTERN
*15 (IF SELECTED) OR THE FIRST WORD OF PAT 13 (BY DEFAULT).
* THEN, AN INSTRUCTION LOOP IS ENTERED IN WHICH A NON-
*EXISTENT UNIBUS DEVICE ADDRESS IS REFERENCED ONCE PER PASS. THIS
*WILL RESULT IN A PROCESSOR SEIZURE OF THE UNIBUS, FOR 5-20 USEC,
*(DEPENDING ON THE PROCESSOR), FOLLOWED BY A NON-EXISTENT MEMORY
*PROCESSOR TRAP, ONCE PER PASS. THUS, FOR EACH PASS THROUGH THE
*INSTRUCTION LOOP THE CONTROLLER IS FORCED TO LOSE FROM ONE
*TO FOUR NPR BUS CYCLES. THE EXECUTION TIME OF THE LOOP DETERMINES THE
*REPETITION RATE OF THE NON-EXISTENT MEMORY REFERENCES, AND THIS
*IS CHOSEN TO ALLOW ENOUGH NPR'S SO THAT THERE SHOULD BE NO
*DATA LATE ERRORS IN A FAULT-FREE CONTROLLER.

```


5982
5983
5984
5985
5986
5987
5988
5989
5990
5991 022416 000004
5992 022420 012737 000005 001324
5993 022426 004737 033672
5994 022432 004737 033740
5995 022436 000137 022722
5996
5997 022442 004737 032310
5998 022446 004737 030576
5999
6000 022452 013765 005660 000002
6001 022460 113765 005664 000005
6002 022466 105065 000004
6003 022472 012765 165000 000012
6004 022500 105737 003125
6005 022504 001403
6006 022506 012765 166000 000012
6007 022514 052765 100000 000014
6008
6009 022522 012700 006776
6010 022526 005737 005700
6011 022532 100402
6012 022534 012700 006676
6013 022540 012701 000020
6014 022544 012703 000070
6015 022550 010065 000010
6016 022554 105037 003141
6017 022560 112737 000001 003142
6018 022566 112765 000123 000001
6019 022574 004737 040750
6020 022600 105737 003141
6021 022604 001011
6022 022606 032737 000002 005474
6023 022614 001042
6024 022616 112765 000131 000001
6025 022624 004737 042542
6026
6027 022630 012765 065660 000010
6028 022636 005037 065660
6029 022642 112765 000121 000001
6030 022650 004737 040750
6031 022654 105737 003141
6032 022660 001014
6033 022662 022037 065660
6034 022666 001411
6035 022670 004737 043720
6036 022674 016037 177776 001174
6037 022702 013737 065660 001176

;;THE DATA IS THEN WRITE CHECKED FOR VERIFICATION, AND THE ABOVE
;;TEST IS REPEATED USING A READ DATA COMMAND. NO ERRORS ARE EXPECTED.
;;
;;THE ABOVE OPERATIONS ARE REPEATED 15 MORE TIMES, USING THE NEXT WORD
;;OF THE PROPER PATTERN (13 OR 15) EACH TIME, TO WRITE AND READ
;;THE ENTIRE TRACK. IF A DATA LATE ERROR OCCURS ON A GIVEN TRANSFER,
;;THAT DATA IS NOT WRITE-CHECKED OR COMPARED.

TST5: SCOPE
MOV #5, \$TESTN ; SET TEST NUMBER IN APT MAIL BOX
JSR PC, SETUP ; SET UP FOR LOOP ON ERROR
JSR PC, CHKITR ; SEE IF ITER. NO. = 0 FOR THIS TEST
JMP TST6 ; JUMP TO NEXT TEST
; RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
JSR PC, INITSS ; INITIALIZE DRIVER PARAMS AND SUB-SYS
JSR PC, PREPKB ; PREPARE FOR POSSIBLE KBD INPUT
; INITIALIZE PARAMETERS
MOV FC, P.CYL(R5) ; SET CYL = FC
MOVB FT, P.TRCK(R5) ; SET TRACK = FT
CLRB P.SECT(R5) ; SET SECTOR = 0
MOV #-13000, P.WC(R5) ; INIT WORD COUNT FOR FULL TRACK
TSTB FORMAT ; DETERMINE THE FORMAT
BEQ 4\$; BR IF 22 SECTORS
MOV #-12000, P.WC(R5) ; SET WORD COUNT FOR FULL TRACK
4\$: BIS #DIBH11, P.PRST(R5) ; SET BUS ADDRESS INCREMENT INHIBIT
; WRITE DATA WITH ALLOWABLE UNIBUS CONTENTION, AND WRITE CHECK IT
MOV #PAT15, RO ; SET DATA PATTERN ADDRESS
PT ; SEE IF USER-DEFINED PATTERN 15 SELECTED
TST 14\$; BR IF YES
MOV #PAT13, RO ; SET DATA PATTERN 13 ADDRESS
14\$: MOV #16, R1 ; SET COUNTER = 16
MOV #70, R3 ; SET NON-EXISTENT MEMORY TIMER
16\$: MOV RO, P.BALO(R5) ; SET BA BITS
CLRB DLFLG ; CLEAR DATA LATE FLAG
MOVB #1, NORTRY ; SET "NO-RETRY" FLAG
MOVB #WRDATA, P.CMND(R5) ; SET WRITE DATA COMMAND
JSR PC, REFNEM ; WRITE DATA DURING NEM REF.
TSTB DLFLG ; SEE IF DATA LATE ERROR OCCURRED
BNE 18\$; BR IF NOT
BIT #BSERR, RECODE ; SEE IF BAD SECTOR ERROR
BNE 24\$; BR IF YES
MOVB #WATCHK, P.CMND(R5) ; SET WRITE CHECK COMMAND
JSR PC, DRVCAL ; DO WRITE CHECK
; READ DATA WITH ALLOWABLE UNIBUS CONTENTION, AND COMPARE LAST WORD
18\$: MOV #RWBUF, P.BALO(R5) ; SET BA = RWBUF ADDRESS
CLRB RWBUF ; CLEAR THE BUFFER WORD
MOVB #RDATA, P.CMND(R5) ; SET READ COMMAND
JSR PC, REFNEM ; READ DATA DURING NEM REF.
TSTB DLFLG ; SEE IF DATA LATE ERROR OCCURRED
BNE 20\$; BR IF YES
CMP (RO)+, RWBUF ; COMPARE LAST DATA WORD
BEQ 20\$; BR IF EQUAL
JSR PC, REPSUP ; GATHER STATUS FOR PRINTOUT
MOV -2(RO), \$REG5 ; GOOD DATA WORD
MOV RWBUF, \$REG6 ; BAD DATA WORD

6038 022710 104110
6039
6040 022712 005301
6041 022714 001402
6042 022716 000137 022550
6043 022722
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074 022722 000004
6075 022724 012737 000006
6076 022732 004737 033672
6077 022736 004737 033740
6078 022742 000137 023742
6079
6080 022746 004737 032310
6081 022752 004737 030576
6082
6083 022756 013703 005660
6084 022762 005004
6085 022764 012700 003234
6086 022770 020310
6087 022772 001006
6088 022774 005203
6089 022776 020337 020040
6090 023002 103767
6091 023004 000137 023742
6092 023010 062700 000004
6093 023014 020027 004224

ERROR 110 ;"READ ERROR WHILE BAI SET"
;SET UP FOR NEXT PASS THROUGH LOOP
20\$: DEC R1 ;SEE IF ALL DONE YET
BEQ 24\$;BR IF YES
JMP 16\$;JUMP TO CONTINUE TESTING
24\$:

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

;TEST 6: SCOPE
MOV #6, \$TESTN ;SET TEST NUMBER IN APT MAIL BOX
JSR PC, SETUP ;SET UP FOR LOOP ON ERROR
JSR PC, CHKTR ;SEE IF ITER. NO. = 0 FOR THIS TEST
JMP \$EOP ;JUMP TO END-OF-PASS ROUTINE
;RETURN HERE FROM CHKTR IF TEST SHOULD BE RUN
JSR PC, INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
JSR PC, PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
;SEARCH BAD SECTOR FILES FOR 2 ADJACENT ERROR-FREE CYLS TO USE
MOV FC, R3 ;GET FC
40\$: CLR R4
42\$: MOV #BSOFT+10, R0 ;POINTER TO SOFTWARE BSF
52\$: CMP R3, (R0) ;SEE IF THIS CYL LISTED
BNE 44\$;BR IF NOT
INC R3 ;INCR CYL
CMP R3, LSTCYL ;SEE IF CYL < 632/1456
BLO 40\$;BR IF YES
JMP MULT1 ;GO SKIP THIS TEST
44\$: ADD #4, R0 ;INCR BSF POINTER
CMP R0, #BSFACT ;SEE IF STILL IN SOFTWARE BSF

6094	023020	001002			BNE	46\$;BR IF YES
6095	023022	062700	000010		ADD	#10,RO		;INCR POINTER FOR FACTORY BSF
6096	023026	020027	005224		46\$: CMP	RO,#BSFACT+512.		;SEE IF END OF FACTORY BSF REACHED
6097	023032	001356			BNE	52\$;BR IF NOT YET
6098	023034	005704			TST	R4		;SEE IF CAME HERE BEFORE
6099	023036	001003			BNE	50\$;BR IF YES
6100	023040	005204			INC	R4		
6101	023042	005203			INC	R3		;INC CYL
6102	023044	000747			BR	42\$;CONTINUE LOOKING
6103	023046	005303			50\$: DEC	R3		;DECR CYL
6104	023050	010337	003202		MOV	R3,CYL		;GET CYL TO USE
6105					;CHECK DRVLST FOR OTHER AVAILABLE DRIVES, AND RECALIBRATE THEM			
6106	023054	112765	000117	000001	MOV	#SEEK,P.CMND(R5)		;SET SEEK COMMAND
6107	023062	005001			CLR	R1		;INIT MULTIPLE-DRIVES FLAG
6108	023064	005000			CLR	RO		;INIT DRIVE NO. TO 0
6109	023066	005065	000002		CLR	P.CYLN(R5)		;SET CYL = 0
6110	023072	022700	000010		4\$: CMP	#10,RO		;SEE IF ALL DRIVES CHECKED YET
6111	023076	001415			BEQ	8\$;BR IF ALL DONE
6112	023100	105760	005610		TSTB	DRVLST(RO)		;SEE IF THIS DRIVE IS MARKED IN LIST
6113	023104	001410			BEQ	6\$;BR IF NOT MARKED
6114	023106	120037	005500		CMPB	RO,DRIVE		;SEE IF MARKED DRIVE IS DRIVE UNDER TEST
6115	023112	001405			BEQ	6\$;BR IF YES
6116	023114	110065	000070		MOV	RO,P.DRVN(R5)		;SET DRIVE NO. PARAMETER
6117	023120	004737	023624		JSR	PC,TSTTYP		;SEEK TO CYL 0 ON CURRENT DRIVE
6118	023124	005201			INC	R1		;SET FLAG TO INDICATE MORE THAN 1 DRIVE
6119	023126	005200			6\$: INC	RO		;INCR DRIVE NO.
6120	023130	000760			BR	4\$;KEEP CHECKING DRIVES
6121	023132	005701			8\$: TST	R1		;SEE IF MORE THAN 1 DRIVE PRESENT
6122	023134	001002			BNE	9\$;BR IF YES, TO RUN TEST
6123	023136	000137	023742		JMP	MULTI1		;NO SKIP TEST
6124					;SEEK TO CYLINDER CYL ON DRIVE UNDER TEST, SET PARAMS FOR DATA XFER			
6125	023142	004737	032310		9\$: JSR	PC,INITSS		;INIT S.S.
6126	023146	013765	003202	000002	MOV	CYL,P.CYLN(R5)		;SET CYLNDR = CYL
6127	023154	005065	000004		CLR	P.SECT(R5)		;SET TRACK AND SECTOR = 0
6128	023160	012765	136400	000012	MOV	#-17152.,P.WC(R5)		;SET WC FOR 67(DEC) SECTORS
6129	023166	105737	003125		TSTB	FORMAT		;CHECK THE FORMAT
6130	023172	001403			BEQ	12\$;BR IF 22-SECTOR FORMAT
6131	023174	012765	141400	000012	MOV	#-15616.,P.WC(R5)		;SET WC FOR 61(DEC) SECTORS
6132	023202	012765	006676	000010	12\$: MOV	#PAT13,P.BALO(R5)		;SET BA BITS
6133	023210	052765	100000	000014	BIS	#DTBA11,P.PRST(R5)		;SET BUS ADDRESS INCREMENT INHIBIT
6134	023216	112765	000117	000001	MOV	#SEEK,P.CMND(R5)		;SET SEEK COMMAND
6135	023224	004737	042542		JSR	PC,DRVCAL		;SEEK TO CYL ON DRIVE UNDER TEST
6136					;DO PSEUDO-RANDOM SEEKS ON ALL OTHER DRIVES			
6137	023230	105037	003120		CLRB	TSTING		;INHIBIT +C, +Z ESCAPE
6138	023234	012701	000007		MOV	#7,R1		;SET LOOP COUNTER
6139	023240	012700	003204		MOV	#CYLLST,RO		;ADRS OF RAND CYL LIST
6140	023244	004737	035666		15\$: JSR	PC,RNDADR		;GENERATE A PSEUDO-RANDOM CYL NO.
6141	023250	042737	177145	005504	BIC	#177145,CYLNDR		;ALWAYS KEEP UNDER 632
6142	023256	013720	005504		MOV	CYLNDR,(RO)+		;SET CYL NO. IN TABLE
6143	023262	001002			BNE	16\$;BR IF NOT 0
6144	023264	005260	177776		INC	-2(RO)		;MAKE IT NON-ZERO
6145	023270	005301			16\$: DEC	R1		;DECREMENT LOOP COUNTER
6146	023272	001364			BNE	15\$;BR IF NOT DONE LOADING LIST YET
6147	023274	012701	003204		MOV	#CYLLST,R1		;GET ADRS OF RAND CYL TABLE
6148	023300	010537	023372		MOV	R5,19\$;SET PARAM BLK ADRS FOR DRIVER
6149	023304	005000			CLR	RO		;INIT DRIVE NO. TO 0

6150	023306	022700	000010		18\$:	CMP	#10,RO	;SEE IF ALL DRIVES CHECKED YET
6151	023312	001432				BEQ	22\$;BR IF YES
6152	023314	105760	005610			TSTB	DRVLST(RO)	;SEE IF THIS DRIVE PRESENT
6153	023320	001425				BEQ	20\$;BR IF NO
6154	023322	120037	005500			CMPB	RO,DRIVE	;SEE IF THIS IS DRIVE UNDER TEST
6155	023326	001422				BEQ	20\$;BR IF YES
6156	023330	110065	000000			MOVB	RO,P.DRVN(R5)	;SET DRIVE NO. PARAMETER
6157	023334	012165	000002			MOV	(R1)+,P.CYLN(R5)	;SET CYLINDER NO.
6158	023340	004737	042622			JSR	PC,STACMD	;STORE PREV AND CURRENT CMNDS
6159	023344	105760	005620			TSTB	DTYLS(TRO)	;SEE IF RK06
6160	023350	001404				BEQ	1\$;BR IF YES
6161	023352	152765	000004	000007		BISB	#B.CDT,P.CS1H(R5)	;ELSE SET FOR RK07
6162	023360	000402				BR	2\$	
6163	023362	105065	000007		1\$:	CLRB	P.CS1H(R5)	
6164	023366	004737	052604		2\$:	JSR	PC,C.INIT	;GO START SEEK ON THIS DRIVE
6165	023372	000000			19\$:	WORD		;P.B. ADRS GOES HERE
6166	023374	005200			20\$:	INC	RO	;INCR DRIVE NO.
6167	023376	000743				BR	18\$;BR TO SEEK ON NEXT DRIVE
6168								;WRITE DATA ON DRIVE UNDER TEST WITH BAI SET
6169	023400	112765	000123	000001	22\$:	MOVB	#WRDATA,P.CMND(R5)	;SET WRITE DATA COMMAND
6170	023406	113765	005500	000000		MOVB	DRIVE,P.DRVN(R5)	;SET DRIVE NO.
6171	023414	013765	003202	000002		MOV	CYL,P.CYLN(R5)	;SET CYL NO.
6172	023422	012737	023654	003044		MOV	#MULHDL,A.NORM	;SET SPECIAL DRIVER RETURN ADDRESS
6173	023430	113700	005500			MOVB	DRIVE,RO	
6174	023434	004737	023624			JSR	PC,TSTTYP	;WRITE 1 CYL + 1 SECTOR AT CYL FC
6175	023440	032737	000002	005474		BIT	#BSERR,RECODE	;SEE IF BSE ERROR
6176	023446	001135				BNE	MULTI1	;BR IF YES
6177	023450	112765	000131	000001		MOVB	#WATCHK,P.CMND(R5)	;SET WRITE CHECK COMMAND
6178	023456	004737	042542			JSR	PC,DRVCAL	;WRITE CHECK THE DATA WRITTEN
6179								;CHECK ATTENTIONS AND CHECK CYLINDER ADDRESSES FROM ALL OTHER DRIVES
6180	023462	112737	000001	003120		MOVB	#1.TSTING	ALLOW IC 12 ESCAPE
6181	023470	112765	000141	000001		MOVB	#R0STAT,P.CMND(R5)	;SET READ DRIVE STATUS COMMAND
6182	023476	012701	003204			MOV	#CYLLST,R1	ADRS OF RAND CYL LIST
6183	023502	005000				CLR	RO	INIT DRIVE NO.
6184	023504	022700	000010		26\$:	CMP	#10,RO	;SEE IF ALL DRIVES CHECKED YET
6185	023510	001514				BEQ	MULTI1	;BR IF YES
6186	023512	105760	005610			TSTB	DRVLST(RO)	;SEE IF THIS DRIVE MARKED IN LIST
6187	023516	001440				BEQ	32\$;BR IF NOT
6188	023520	120037	005500			CMPB	RO,DRIVE	;SEE IF THIS IS DRIVE UNDER TEST
6189	023524	001435				BEQ	32\$;BR IF YES
6190	023526	136037	003102	003200		BITB	I.DRV(RO),SAVWRD	;SEE IF GOT ATT'N FROM THIS DRIVE
6191	023534	001003				BNE	28\$;BR IF YES
6192	023536	004737	043720			JSR	PC,REPSUP	;PREPARE STATUS FOR PRINTOUT
6193	023542	104113				ERROR	113	"NO ATTENTION ON SEEK"
6194	023544	110065	000000		28\$:	MOVB	RO,P.DRVN(R5)	;SET THIS DRIVE NO.
6195	023550	004737	023624			JSR	PC,TSTTYP	;READ STATUS OF THIS DRIVE
6196	023554	016503	000052			MOV	P.B10(R5),R3	;GET STATUS BYTE B10
6197	023560	006003				ROR	R3	;GET CYL ADRS RIGHT-JUSTIFIED
6198	023562	006003				ROR	R3	
6199	023564	006003				ROR	R3	
6200	023566	006003				ROR	R3	
6201	023570	042703	177000			BIC	#177000,R3	;CLEAR OFF UNUSED BITS
6202	023574	022103				CMP	(R1)+,R3	;COMPARE TO EXPECTED CYLINDER
6203	023576	001410				BEQ	32\$;BR IF EQUAL
6204	023600	016137	177776	001212		MOV	-2(R1),SREG14	;GOOD CYL NO.
6205	023606	010337	001214			MOV	R3,SREG15	;BAD CYL NO.

```

6206 023612 004737 043720      JSR    PC,REPSUP      ;GATHER STATUS FOR PRINTOUT
6207 023616 104114              ERROR    114          ;"DRIVE'S CYLINDER INCORRECT"
6208 023620 005200              32$:    INC    R0          ;INCREMENT DRIVE NO.
6209 023622 000730              BR        26$          ;BR IF NOT DONE WITH ALL DRIVES YET
6210
6211      ;THIS ROUTINE TESTS FOR THE DRIVE TYPE AND SETS P.CS1H ACCORDINGLY
6212
6213 023624 105760 005620      TSTTYP: TSTB    DTYLST(R0)      ;SEE IF RK06
6214 023630 001404              BEQ     1$              ;BR IF YES
6215 023632 152765 000004 000007      BISH    #B.CDT,P.CS1H(R5)      ;ELSE SETUP FOR RK07
6216 023640 000402              BR        2$              ;
6217 023642 105065 000007      1$:    CLRB    P.CS1H(R5)      ;SETUP FOR RK06
6218 023646 004737 042542      2$:    JSR     PC,DRVCAL
6219 023652 000207              RTS     PC
6220
6221      ;THIS IS THE NORMAL RETURN FROM THE DRIVER WHICH IS USED TO CHECK
6222      ;THE RESULTS OF MULTI-DRIVE OPERATIONS.
6223
6224 023654 032737 000200 003000      MULHDL: BIT     #RDY,T.CS1      ;SEE IF CNTRLR RDY IS SET
6225 023662 001010              BNE     6$              ;BR IF RDY SET
6226 023664 004737 052204              JSR     PC,I.CSTS      ;READ CONTROLLER REGISTERS
6227 023670 013765 003000 000016      MOV     T.CS1,P.CS1(R5)      ;GET CS1 BITS
6228 023676 004737 043720              JSR     PC,REPSUP      ;PREPARE STATUS FOR PRINTOUT
6229 023702 104112              ERROR    112          ;"INTRPT WHEN CNTRLR NOT RDY"
6230 023704 013737 003014 003200      6$:    MOV     T,ASOF,SAVWRD      ;SAVE ATT'N SUMMARY
6231 023712 000337 003200              SWAB    SAVWRD      ;GET ATT'N BITS IN BITS 0-7
6232 023716 112765 000177 000001      MOV     #SUBCLR,P.CMND(R5)      ;SET SUBSYSTEM CLEAR COMMAND
6233 023724 012737 042732 003044      MOV     #ERRFRE,A.NORM      ;RESTORE NORMAL DRIVER RETURN ADDRESS
6234 023732 004737 042542              JSR     PC,DRVCAL      ;CLEAR SUB-SYS
6235 023736 000137 042732              JMP     ERRFRE      ;TAKE NORMAL RETURN
6236
6237 023742      MULTI1:
6238
6239
6240
6241      .SBTTL  END OF PASS ROUTINE
6242
6243      ;*****
6244      ;*INCREMENT THE PASS NUMBER ($PASS)
6245      ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
6246      ;*IF THERES A MONITOR GO TO IT
6247      ;*IF THERE ISN'T JUMP TO APTPAS
6248
6249      $EOP:
6250 023742 000004              SCOPE
6251 023744 005237 001330              INC     $DEVCT      ;INCREMENT DEVICE COUNT FOR APT
6252 023750 000137 017332              JMP     NEWDRV      ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
6253 023754              DUNPAS: ;THIS IS TRULY THE END OF A PASS
6254 023754 042777 000100 155162      BIC     #BIT6,$STKS      ;DISABLE ITY KBD INTERRUPT
6255 023762 005037 001102              CLR     $STNM      ;ZERO THE TEST NUMBER
6256 023766 005037 001304              CLR     $TIMES      ;ZERO THE NUMBER OF ITERATIONS
6257 023772 005237 001326              INC     $PASS      ;INCREMENT THE PASS NUMBER
6258 023776 042737 100000 001326      BIC     #100000,$PASS      ;DON'T ALLOW A NEG. NUMBER
6259 024004 005327              DEC     (PC)+      ;LOOP?
6260 024006 000001              $EOPCT: .WORD    1
6261 024010 003022              BGT     $DOAGN      ;:YES

```

```

SENDCT:  MOV      (PC)+,a(PC)+      ;;RESTORE COUNTER
        .WORD    1
        SEOPCT
        TYPE     $ENDMG              ;;TYPE "END PASS #"
        MOV      $PASS,-(SP)         ;;SAVE $PASS FOR TYPEOUT
        TYPDS    ;;GO TYPE--DECIMAL ASCII WITH SIGN
        TYPE     $ENULL              ;;TYPE A NULL CHARACTER
$GET42:  MOV      a#42,RO             ;;GET MONITOR ADDRESS
        BEQ      $DOAGN              ;;BRANCH IF NO MONITOR
        RESET    ;;CLEAR THE WORLD
$ENDAD:  JSR      PC,(RO)             ;;GO TO MONITOR
        NOP      ;;SAVE ROOM
        NOP      ;;FOR
        NOP      ;;ACT11
$DOAGN:
$RTNAD:  JMP      a(PC)+              ;;RETURN
        .WORD    APTPAS
$ENULL:  .BYTE    -1,-1,0            ;;NULL CHARACTER STRING
$ENDMG:  .ASCIZ   <15><12>/END PASS #/

APTAS:   CMPB     #APTENV,$ENV        ;RUN UNDER APT ?
        BNE      2$ ;BRANCH IF NOT
        CMP      $PASS,#2            ;TWO PASSES ?
        BLO      2$ ;BRANCH IF NOT
1$:      INC      $TSTNM              ;INCREMENT TEST NUMBER
        BR       1$ ;WAITING LOOP
2$:      JMP      NEWPAS              ;JMP TO THE MAIN PROGRAM

.SBTTL   RK06 HEAD ALIGNMENT AID

```

```

6292 024134 012737 000340 177776 ASTART: MOV #PR7,2#PS
6293 .SBTTL INITIALIZE THE COMMON TAGS
6294 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
6295 024142 012706 001100 MOV #SCMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
6296 024146 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
6297 024150 022706 001140 CMP #SWR,R6 ;;DONE?
6298 024154 001374 BNE -6 ;;LOOP BACK IF NO
6299 024156 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
6300 ;;INITIALIZE A FEW VECTORS
6301 024162 012737 057170 000020 MOV #SSCOPE,2#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
6302 024170 012737 000340 000022 MOV #340,2#IOTVEC+2 ;;LEVEL 7
6303 024176 012737 056464 000030 MOV #SEERROR,2#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
6304 024204 012737 000340 000032 MOV #340,2#EMTVEC+2 ;;LEVEL 7
6305 024212 012737 060276 000034 MOV #STRAP,2#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
6306 024220 012737 000340 000036 MOV #340,2#TRAPVEC+2 ;;LEVEL 7
6307 024226 012737 057034 000024 MOV #SPWRDN,2#PWAVEC ;;POWER FAILURE VECTOR
6308 024234 012737 000340 000026 MOV #340,2#PWAVEC+2 ;;LEVEL 7
6309 024242 013737 024014 024006 MOV $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
6310 024250 005037 001304 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
6311 024254 005037 001306 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
6312 024260 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
6313 024266 012737 024266 001106 MOV #.,$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
6314 024274 012737 024274 001110 MOV #.,$LPERR ;;SETUP THE ERROR LOOP ADDRESS
6315 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
6316 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
6317 024302 013746 000004 MOV 2#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
6318 024306 012737 024342 000004 MOV #64,$2#ERRVEC ;;SET UP ERROR VECTOR
6319 024314 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
6320 024322 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
6321 024330 022777 177777 154602 CMP #-1,2#SWR ;;TRY TO REFERENCE HARDWARE SWR
6322 024336 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
6323 ;;AND THE HARDWARE SWR IS NOT = -1
6324 024340 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
6325 024342 012716 024350 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
6326 024346 000002 RTI
6327 024350 012737 000176 001140 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
6328 024356 012737 000174 001142 MOV #DISPREG,DISPLAY
6329 024364 012637 000004 66$: MOV (SP)+,2#ERRVEC ;;RESTORE ERROR VECTOR
6330
6331 024370 005037 001326 CLR $PASS ;;CLEAR PASS COUNT
6332 024374 132737 000200 001341 BITB #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
6333 024402 001403 BEQ 67$ ;;YES,USE NON-APT SWITCH
6334 024404 012737 001342 001140 MOV #SSWREG,SWR ;;NO,USE APT SWITCH REGISTER
6335 024412 67$:
6336 024412 005037 003150 CLR AUTOFG ;;CLEAR THE SPECIAL FLAG
6337 .SBTTL RK06 HEAD ALIGNMENT AID
6338 024416 000005 RESET ;;RESET UNIBUS
6339 024420 104401 007036 TYPE ,DZR6N ;;TYPE PROGRAM I.D. FOR PART 2
6340 024424 104401 007050 TYPE ,SUBVER
6341 024430 104401 007136 TYPE ,PART2
6342 024434 012737 000176 001140 MOV #SWREG,SWR ;;ADRS OF SOFTWARE SWR
6343 024442 005077 154472 CLR 2#SWR ;;MAKE SWR BITS ALL 0
6344 024446 012737 030436 000060 MOV #KBDHDL,2#TKVEC ;;LOAD VECTOR FOR TTY KBD
6345 024454 012737 000200 000062 MOV #PR4,2#TKVEC+2 ;;SET KBD PRIORITY = 4
6346 024462 013701 003040 MOV RKVEC,R1 ;;GET ADDRESS OF VECTOR STORAGE
6347 024466 012721 047304 MOV #I.INTR,(R1)+ ;;SET IT TO INTERRUPT HNDLR

```

```

6348 024472 012711 000340      MOV      #PR7,(R1)      ;SET INTERRUPT HANDLER PR7
6349 024476 012737 000000      MOV      #PR0,#PS      ;ALLOW ALL INTERRUPTS
6350 024504 012737 000000      MOV      #0,FC      ;SET FIRST CYL = 0
6351
6352 024512      GIVEID:      MOV      #176543,$HINUM      ;INITIALIZE PSEUDO-RANDOM NUMBERS
6353 024512 012737 176543      MOV      #123456,$LONUM
6354 024520 012737 123456      CLRB      MDFLAG      ;CLEAR PARAM INP FLAG
6355 024526 105037 003116      CLRB      TSTING      ;CLEAR "RUNNING TESTS" FLAG
6356 024532 105037 003120      CLR      STALLS      ;DON'T STALL ON OPERATIONS
6357 024536 005037 005502      MOV      #PRVCMO,R1      ;ZERO OUT PREV COMMAND
6358 024542 012701 005224      MOV      #6,R0
6359 024546 012700 000006      CLRB      (R1)+
6360 024552 005021      42$:      DEC      R0
6361 024554 005300      BNE      42$
6362 024556 001375      CLRB      ERRCNT      ;CLEAR ERROR COUNT FOR RESTARTS
6363 024560 105037 003126      TYPE      ,IDENT      ;TYPE PROGRAM IDENTIFICATION
6364 024564 104401 011365      TSTB      HLPVOL      ;SEE IF HELP FILE OVERLAID
6365 024570 105737 003145      BNE      ASKMOD      ;BR IF YES
6366
6367 024574 001011      BNE      ASKTYP      ;ASK FOR ALL DRIVE TYPES
6368 024576 104401 011431      TYPE      ,FORHLP      ;ASK IF HELP DESIRED
6369 024602 104406      RDCHR      ;READ RESPONSE
6370 024604 112601      MOV      (SP)+,R1      ;GET CHARACTER
6371 024606 122701 000015      CMP      #015,R1      ;SEE IF <CR> TYPED
6372
6373 024612 001402      BEQ      ASKMOD      ;BR IF NO HELP NEEDED
6374 024614 104401 066277      BEQ      ASKTYP      ;ASK FOR ALL DRIVE TYPES
6375
6376 024620 104401 001315      TYPE      ,HLPFIL      ;HELP REQUESTED- TYPE INFO.
6377 024624 104401 012554      ASKTYP:  TYPE      ,SCRLF      ;TO ENTER ALL DRIVE TYPES
6378 024630 005000      TYPE      ,DTYMSG      ;MESSAGE TO ENTER DRIVE TYPE
6379 024632 104401 001315      CLRB      RD      ;START FROM DRIVE 0
6380 024636 105037 007530      1$:      TYPE      ,SCRLF      ;CR-LF
6381 024642 104401 007522      CLRB      BADDRV+6      ;CLEAR THE LOC 6 FOR TYPE
6382 024646 010046      TYPE      ,BADDRV      ;DRIVE
6383 024650 104402      MOV      RD,-(SP)      ;DRIVE NUMBER ON STACK
6384 024652 104401 012576      TYPOC      ;TYPE THE DRIVE NUMBER
6385 024656 004737 030576      TYPE      ,SOR7      ;ENTER 6 OR 7
6386 024662 005737 005522      JSR      PC,PREPKB      ;READ THE NUMBER
6387 024666 001775      2$:      TST      INTCHR      ;ANY THING READ IN YET ?
6388 024670 105060 005620      BEQ      2$      ;BRANCH IF NOT
6389 024674 013701 005522      CLRB      DTYLST(R0)      ;ASSUME IT'S A RK06
6390 024700 022701 000067      MOV      INTCHR,R1      ;READ THE CH
6391 024704 001003      CMP      #'7,R1      ;IT'S A RK07 ?
6392 024706 112760 000004      BNE      3$      ;BRANCH IF NOT
6393 024714 005200 005620      MOV      #B.CDT,DTYLST(R0)      ;SET RK07 FLAG
6394 024716 022700 000007      3$:      INC      R0      ;TO NEXT DRIVE
6395 024722 103343      CMP      #7,R0      ;ALL 8 DRIVES ARE SET UP YET ?
6396
6397      BHIS      1$      ;BRANCH IF NOT
6398
6398 024724 104401 011466      ;DETERMINE DESIRED OPERATIONAL MODE
6399 024730 004737 030576      ASKMOD:  TYPE      ,TYPMOD      ;ASK FOR DESIRED OPERATIONAL MODE
6400 024734 005737 005522      JSR      PC,PREPKE      ;PREPARE FOR KBD INPUT
6401 024740 001775      1$:      TST      INTCHR      ;CHECK FOR TTY INPUT
6402 024742 013701 005522      BEQ      1$      ;BR IF NO INPUT YET
6403 024746 022701 000022      MOV      INTCHR,R1      ;GET INPUT CHAR INTO R1
        CMP      #022,R1      ;SEE IF (↑R) TYPED

```


6404	024752	001657			BEQ	GIVEID	; BR IF (↑R) TO RESTART
6405	024754	022701	000115		CMP	#'M,R1	; IS IT MANUAL MODE ?
6406	024760	001002			BNE	2\$; BR IF NOT MANUAL
6407	024762	000137	025006		JMP	MANUAL	; JUMP TO MANUAL MODE ROUTINE
6408	024766	022701	000101	2\$:	CMP	#'A,R1	; IS IT AUTO MODE ?
6409	024772	001002			BNE	3\$; BR IF NOT AUTO
6410	024774	000137	026144		JMP	AUTO	; JUMP TO AUTO MODE ROUTINE
6411	025000	004737	030616	3\$:	JSR	PC,ECOBAD	; ECHO BAD INPUT
6412	025004	000747			BR	ASKMOD	; BR TO ASK FOR MODE AGAIN
6413							
6414							
6415							
6416							
6417	025006						
6418	025006	005037	003150		MANUAL:	CLR	AUTOFG
6419	025012	104401	011527			TYPE	,TYPMAN
6420	025016	104401	011562		ASKDRV:	TYPE	,ENTDRV
6421	025022	004737	030576			JSR	PC,PREPKB
6422	025026	005737	005522	1\$:	TST	INTCHR	; SEE IF ANY INPUT
6423	025032	001775			BEQ	1\$; BR IF NONE TO CHECK AGAIN
6424	025034	013701	005522		MOV	INTCHR,R1	; GET CHARACTER INTO R1
6425	025040	020127	000022		CMP	R1,#022	; TEST FOR (↑R) TYPED
6426	025044	001622			BEQ	GIVEID	; BR IF (↑R) TO RESTART
6427	025046	020127	000060		CMP	R1,#'0	; COMPARE TO ASCII 0
6428	025052	002512			BLT	BA1FLU	; BR IF <0 (BAD INPUT)
6429	025054	020127	000067		CMP	R1,#'7	; COMPARE TO ASCII 7
6430	025060	003107			BGT	BA1FLU	; BR IF >7 (BAD INPUT)
6431	025062	042701	000060		BIC	#'0,R1	; STRIP ASCII BITS
6432	025066	010137	005500		MOV	R1,DRIVE	; STORE DRIVE NUMBER
6433	025072	013700	005500		MOV	DRIVE,R0	; AND SET UP TO SIZE DRIVE
6434	025076	116037	005620	003124	MOVB	DTYLS1(R0),TYPFMT	; LOAD THE FLAG FOR 6 OR 7
6435	025104	012737	000624	020036	MOV	#624,LCM6	
6436	025112	012737	000631	005736	MOV	#631,PRMLIM+2	
6437	025120	012737	000632	005662	MOV	#632,LC	
6438	025126	012737	000632	005710	MOV	#632,PRDFLT+2	
6439	025134	012737	000632	005742	MOV	#632,PRMLIM+6	
6440	025142	012737	000400	020042	MOV	#100,HOLD1	
6441	025150	012737	000025	020044	MOV	#25,HOLD3	
6442	025156	012737	010025	020046	MOV	#10025,HOLD4	
6443	025164	012737	000632	020040	MOV	#632,LSTCYL	; LOAD THE LAST CYLINDER NUMBER
6444	025172	105737	003124		TSTB	TYPFMT	; IS AN RK07 ?
6445	025176	001433			BEQ	9\$; BRANCH IF NOT
6446	025200	012737	001450	020036	MOV	#1450,LCM6	
6447	025206	012737	001455	005736	MOV	#1455,PRMLIM+2	
6448	025214	012737	001456	005662	MOV	#1456,LC	
6449	025222	012737	001456	005710	MOV	#1456,PRDFLT+2	
6450	025230	012737	001456	005742	MOV	#1456,PRMLIM+6	
6451	025236	012737	002025	020044	MOV	#2025,HOLD3	
6452	025244	012737	012025	020046	MOV	#12025,HOLD4	
6453	025252	012737	001000	020042	MOV	#1000,HOLD1	
6454	025260	012737	001456	020040	MOV	#1456,LSTCYL	; OTHERWISE SET UP THE LAST CYLINDER
6455	025266	000240			9\$:	NOP	
6456						MOVB	#-1,HDALI
6457						JMP	2#TYPED
6458					INTGO:	CLR	HDALI
6459	025270	013737	020040	005662	MOV	LSTCYL,LC	; SET PROGRAM STATUS FLAG ; GET DRIVE STATISTICS ; HOUSEKEEP ; AND SETUP MAX CYLINDER CNT

6460	025276	000403		BAIFLU:	BR	BAIA	: PROCEED WITH DESIRED DRIVE
6461	025300	004737	030616		JSR	PC,ECOBAD	: ECHO BAD INPUT
6462	025304	000644			BR	ASKDRV	: GO BACK TO ASK AGAIN
6463	025306	004737	032310	BAIA:	JSR	PC,INITSS	: INITIALIZE SUBSYSTEM
6464				:CHECK	DESIRED DRIVE FOR PROPER STATUS		
6465	025312	004737	033252		JSR	PC,CHKDRV	: CHECK DRIVE FOR ON-LINE,READY,WRITE LOCK
6466	025316	025016			ASKDRV		: ADDRESS OF ERROR RETURN FOR CHKDRV
6467	025320	004737	035344		JSR	PC,DRVSR	: TYPE "DRIVE SER. NO. XXX"
6468				:SET UP	DESIRED SUB-MODE OF OPERATION		
6469	025324			SUBMOD:			
6470	025324	105037	011364		CLRB	VERIFY	: CLEAR VERIFY MODE FLAG
6471	025330	104401	012253		TYPE	ENTMOD	: ASK FOR ALIGN OR EXERCISE SUB-MODE
6472	025334	004737	030576		JSR	PC,PREPKB	: PREPARE FOR KBD INPUT
6473	025340	005737	005522	1\$:	TST	INTCHR	: SEE IF ANY INPUT
6474	025344	001775			BEQ	1\$: BR IF NONE TO CHECK AGAIN
6475	025346	013701	005522		MOV	INTCHR,R1	: GET CHAR INTO R1
6476	025352	020127	000022		CMP	R1,#022	: SEE IF (↑R) TYPED
6477	025356	001002			BNE	2\$: BR IF NOT (↑R)
6478	025360	000137	024512		JMP	GIVEID	: JUMP TO RESTART
6479	025364	020127	000003	2\$:	CMP	R1,#003	: SEE IF (↑C) TYPED
6480	025370	001002			BNE	3\$: BR IF NOT (↑C)
6481	025372	000137	025016		JMP	ASKDRV	: JUMP TO ASK FOR DRIVE AGAIN
6482	025376	020127	000101	3\$:	CMP	R1,#'A	: SEE IF ALIGNMENT REQUESTED
6483	025402	001415			BEQ	MANAL	: BR IF MANUAL ALIGNMENT REQUESTED
6484	025404	020127	000105		CMP	R1,#'E	: SEE IF EXERCISES REQUESTED
6485	025410	001002			BNE	4\$: BR IF NOT EXERCISES
6486	025412	000137	025710		JMP	MANEX	: JUMP TO DO MANUAL EXERCISES
6487	025416	020127	000126	4\$:	CMP	R1,#'V	: SEE IF VERIFY REQUESTED
6488	025422	001002			BNE	6\$: BR IF NOT VERIFY
6489	025424	000137	025672		JMP	MANVR	: JUMP TO VERIFY MODE
6490	025430	004737	030616	6\$:	JSR	PC,ECOBAD	: ECHO BAD INPUT
6491	025434	000733			BR	SUBMOD	: ASK FOR SUB-MODE AGAIN
6492							
6493							
6494							
6495							
6496	025436			:MANUAL ALIGNMENT MODE			
6497	025436	104401	012413	MANAL:	TYPE	MANALN	: REPORT MANUAL ALIGNMENT MODE
6498				:REQUEST AND CHECK DESIRED HEAD			
6499	025442	104401	012620	ASKHED:	TYPE	ENTHED	: ASK FOR DESIRED HEAD
6500	025446	004737	030576		JSR	PC,PREPKB	: PREPARE FOR KBD INPUT
6501	025452	005737	005522	1\$:	TST	INTCHR	: SEE IF ANY INPUT
6502	025456	001775			REQ	1\$: BR IF NONE TO CHECK AGAIN
6503	025460	013701	005522		MOV	INTCHR,R1	: GET CHAR INTO R1
6504	025464	020127	000022		CMP	R1,#022	: SEE IF (↑R)
6505	025470	001007			BNE	2\$: BR IF NOT (↑R)
6506	025472	112765	000113	000001	MOVB	#RECAL,P.CMND(R5)	: SET RECALIBRATE COMMAND
6507	025500	004737	042542		JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6508	025504	000137	024512		JMP	GIVEID	: JUMP TO RESTART
6509	025510	020127	000003	2\$:	CMP	R1,#003	: SEE IF (↑C)
6510	025514	001007			BNE	3\$: BR IF NOT (↑C)
6511	025516	112765	000113	000001	MOVB	#RECAL,P.CMND(R5)	: SET RECALIBRATE COMMAND
6512	025524	004737	042542		JSR	PC,DRVCAL	: RECALIBRATE DRIVE
6513	025530	000137	025016		JMP	ASKDRV	: JUMP TO ASK FOR DRIVE AGAIN
6514	025534	020127	000032	3\$:	CMP	R1,#032	: SEE IF (↑Z)
6515	025540	001007			BNE	4\$: BR IF NOT (↑Z)

```

6516 025542 112765 000113 000001      MOVB      #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6517 025550 004737 042542      JSR        PC,DRVCL      ;RECALIBRATE DRIVE
6518 025554 000137 025324      JMP        SUBMOD        ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
6519 025560 020127 000060      4$:      CMP        R1,#'0      ;COMPARE TRACK TO ASCII 0
6520 025564 002410      BLT        5$              ;BR IF <0 (BAD INPUT)
6521 025566 020127 000062      CMP        R1,#'2      ;COMPARE TRACK TO ASCII 2
6522 025572 003005      BGT        5$              ;BR IF >2 (BAD INPUT)
6523 025574 042701 000060      BIC        #'0,R1      ;STRIP ASCII BITS
6524 025600 010137 005520      MOV        R1,TRACK      ;STORE TRACK (HEAD) NUMBER
6525 025604 000404      BR        6$              ;GO SELECT HEAD
6526 025606 004737 030616      5$:      JSR        PC,ECOBAD      ;ECHO BAD INPUT
6527 025612 000137 025442      JMP        ASKHED        ;ASK AGAIN FOR HEAD NUMBER
6528      ;UNLOAD HEADS,AND WHEN READY, START SPINDLE AND SEEK TO ALN CYL
6529      6$:      TSTB      VERIFY      ;SEE IF VERIFY MODE
6530 025622 001002      BNE        8$              ;BR IF VERIFY
6531 025624 004737 033566      JSR        PC,WAIT4R      ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
6532 025630 113765 005520      000005 8$:      MOVB      TRACK,P.TRCK(R5) ;SET DESIRED HEAD NUMBER
6533 025636 004737 033512      JSR        PC,ALNSEK      ;SEEK TO ALIGNMENT CYLINDER
6534 025642 104401 012510      TYPE      HEDPOS        ;TYPE HEADS POSITIONED MSG
6535 025646 013701 005520      MOV        TRACK,R1      ;GET BINARY TRACK NO.
6536 025652 152701 000060      BISB      #'0,R1      ;CONVERT TO ASCII
6537 025656 110137 012704      MOVB      R1,HEADNO      ;GET HEAD NO. INTO MSG BUFF.
6538 025662 104401 012677      TYPE      HEDSEL        ;TYPE HEAD SELECTED MSG
6539 025666 000137 025442      JMP        ASKHED        ;GO BACK TO ASK FOR NEW HEAD SELECT
6540
6541
6542
6543      ;MANUAL SELECT VERIFY MODE
6544 025672 112737 000001 011364      MANVR:      MOVB      #'1,VERIFY      ;SET VERIFY MODE FLAG
6545 025700 104401 012453      TYPE      MANVRF        ;REPORT MANUAL VERIFY MODE
6546 025704 000137 025442      JMP        ASKHED        ;GO ASK FOR DESIRED HEAD
6547
6548
6549
6550      ;MANUAL RANDOM SEEK EXERCISE ROUTINE
6551      MANEX:
6552 025710 004737 033566      JSR        PC,WAIT4R      ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
6553 025714 013701 005500      MOV        DRIVE,R1      ;GET DRIVE NUMBER
6554 025720 052701 000060      BISB      #'0,R1      ;CONVERT TO ASCII
6555 025724 110137 012405      MOVB      R1,DR1EXR      ;PUT DRIVE NUMBER INTO OUTPUT BUFFER
6556 025730 104401 012326      TYPE      MANEXR        ;TYPE RANDOM SEEKS MSG
6557 025734 012700 016514      MOV        #RNDLNG,RO      ;INITIALIZE RANDOM SEEK COUNTER
6558 025740 112765 000117 000001      MOVB      #SEEK,P.CMND(R5) ;SET SEEK COMMAND
6559 025746 004737 030576      1$:      JSR        PC,PREPKB      ;PREPARE FOR KBD INPUT
6560 025752 004737 035666      2$:      JSR        PC,RNDADR      ;SELECT RANDOM CYLINDER ADDRESS
6561 025756 013765 005504 000002      MOV        CYLNDR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
6562 025764 004737 042542      JSR        PC,DRVCL      ;DO A RANDOM SEEK
6563      ;CHECK FOR TTY INPUT DURING EXERCISES
6564 025770 005737 005522      TST        INTCHR        ;SEE IF ANY INPUT
6565 025774 001016      BNE        3$              ;BR IF A CHAR WAS TYPED
6566 025776 005300      DEC        RO              ;DECREMENT SEEK COUNTER
6567 026000 001364      BNE        2$              ;BR IF NOT DONE SEEKING YET
6568 026002 042777 000100 153134      BIC        #BIT6,ISTKS      ;DISABLE KBD INTERRUPT
6569 026010 112765 000113 000001      MOVB      #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6570 026016 004737 042542      JSR        PC,DRVCL      ;RECALIBRATE DRIVE
6571 026022 104401 001310      TYPE      ,SBELL        ;RING BELL TO SIGNIFY END OF EXERCISES

```

```

6572 026026 000137 025324      JMP      SUBMOD      ;GO ASK FOR NEW MANUAL SUB-MODE
6573 026032 013701 005522      3$: MOV      INTCHR,R1      ;GET CHARACTER INTO R1
6574 026036 020127 000022      CMP      R1,#022      ;SEE IF (↑R)
6575 026042 001007      BNE      4$      ;BR IF NOT (↑R)
6576 026044 112765 000113 000001 MOVB     #RECAL.P.CMND(R5) ;SET RECALIBRATE COMMAND
6577 026052 004737 042542      JSR      PC,DRVCL      ;RECALIBRATE DRIVE
6578 026056 000137 024512      JMP      GIVEID      ;JUMP TO RESTART
6579 026062 020127 000003      4$: CMP      R1,#003      ;SEE IF (↑C)
6580 026066 001007      BNE      5$      ;BR IF NOT (↑C)
6581 026070 112765 000113 000001 MOVB     #RECAL.P.CMND(R5) ;SET RECALIBRATE COMMAND
6582 026076 004737 042542      JSR      PC,DRVCL      ;RECALIBRATE DRIVE
6583 026102 000137 025016      JMP      ASKDRV      ;JUMP TO ASK FOR NEW DRIVE NUMBER
6584 026106 020127 000032      5$: CMP      R1,#032      ;SEE IF (↑Z)
6585 026112 001007      BNE      6$      ;BR IF NOT (↑Z)
6586 026114 112765 000113 000001 MOVB     #RECAL.P.CMND(R5) ;SET RECALIBRATE COMMAND
6587 026122 004737 042542      JSR      PC,DRVCL      ;RECALIBRATE DRIVE
6588 026126 000137 025324      JMP      SUBMOD      ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
6589 026132 004737 030616      6$: JSR      PC,ECOBAD      ;ECHO BAD INPUT
6590 026136 104401 012326      TYPE     MANEXR      ;VERIFY STILL IN MANUAL EXERCISES
6591 026142 000701      BR       1$      ;BR TO CONTINUE EXERCISES
6592
6593
6594
6595      :AUTO SELECT MODE ROUTINE
6596      AUTO:
6597 026144 012737 177777 003150 MOV      #-1,AUTOFG      ;SET SPECIAL FLAG
6598 026152 004737 032310      JSR      PC,INITSS      ;INITIALIZE SUBSYSTEM
6599 026156 104401 012721      TYPE     TYPAUT      ;TYPE "AUTO SELECT MODE"
6600 026162 105037 011364      1$: CLRB     VERIFY      ;CLEAR VERIFY MODE FLAG
6601 026166 104401 012253      TYPE     ENTMOD      ;ASK FOR ALIGN,VERIFY, OR EXERCISE
6602 026172 004737 030576      JSR      PC,PREPKB      ;PREPARE FOR KEYBOARD INPUT
6603 026176 005737 005522      2$: TST      INTCHR      ;SEE IF ANY INPUT
6604 026202 001775      BEQ      2$      ;BR IF NONE, TO CHECK AGAIN
6605 026204 013701 005522      MOV      INTCHR,R1      ;GET CHAR INTO R1
6606 026210 020127 000022      CMP      R1,#022      ;SEE IF (↑R) TYPED
6607 026214 001002      BNE      3$      ;BR IF NOT (↑R)
6608 026216 000137 024512      JMP      GIVEID      ;JUMP TO RESTART
6609 026222 020127 000101      3$: CMP      R1,#'A      ;SEE IF AUTO ALIGNMENT REQUESTED
6610 026226 001417      BEQ      AUTAL      ;BR IF AUTO ALIGNMENT REQUESTED
6611 026230 020127 000105      CMP      R1,#'E      ;SEE IF AUTO EXERCISES REQUESTED
6612 026234 001002      BNE      4$      ;BR IF EXERCISES NOT REQUESTED
6613 026236 000137 027520      JMP      AUTEX      ;JUMP TO DO AUTO EXERCISES
6614 026242 020127 000126      4$: CMP      R1,#'V      ;SEE IF VERIFY REQUESTED
6615 026246 001004      BNE      6$      ;BR IF NOT VERIFY
6616 026250 112737 000001 011364 MOVB     #1,VERIFY      ;SET VERIFY MODE FLAG
6617 026256 000403      BR       AUTAL      ;PROCEED IN VERIFY MODE
6618 026260 004737 030616      6$: JSR      PC,ECOBAD      ;ECHO BAD INPUT
6619 026264 000736      BR       1$      ;BR TO ASK FOR A OR E AGAIN
6620
6621
6622
6623      :AUTO ALIGNMENT MODE
6624      AUTAL:
6625 026266 105737 011364      TSTB     VERIFY      ;SEE IF AUTO VERIFY MODE
6626 026272 001403      BEQ      3$      ;BR IF NOT
6627 026274 104401 013010      TYPE     ,AUTVRF      ;REPORT AUTO VERIFY MODE

```

6628	026300	000402			BR	45	
6629	026302	104401	012752		3\$: TYPE	AUTALN	;TYPE ALIGNMENT MESSAGE
6630	026306	004737	030576		4\$: JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
6631	026312	005037	003024		CLR	T.MR2	;CLEAR MR2 STORAGE WORD
6632	026316	005037	005524		CLR	SELECT	;INITIALIZE AUTO-SELECTED DRIVE INDICATOR
6633	026322	012737	000377	005526	MOV	#377,ONLINE	;INIT. OLD ONLINE DRIVE INDICATOR
6634	026330	012737	000377	005530	MOV	#377,NEWON	;INIT. NEW ONLINE DRIVE INDICATOR
6635	026336	012737	000377	005534	MOV	#377,SCRACH	;INITIALIZE LAST DRIVE INDICATOR
6636	026344	005737	005522		DRVLUP: TST	INTCHR	;SEE IF ANY KBD INPUT
6637	026350	001423			BEQ	65	;BR IF NO INPUT
6638	026352	113765	005534	000000	MOV	SCRACH,P.DRVN(R5)	;DRIVE NO. FOR POSSIBLE RECALIBRATE
6639	026360	123727	005522	000032	CMP	INTCHR,#032	;SEE IF (↑2) TYPED
6640	026366	001002			BNE	25	;BR IF NOT (↑2)
6641					CMP	#377,SCRACH	;SEE IF ANY DRIVE SELECTED YET
6642					BEQ	15	;BR IF NONE
6643					MOV	#RECAL,P.CMND(R5)	;SET RECALIBRATE COMMAND
6644					JSR	PC,DRVCAL	;RECALIBRATE DRIVE
6645	026370	000137	026144		JMP	AUTO	;RESTART AUTO MODE
6646	026374	123727	005522	000022	1\$: CMP	INTCHR,#022	;SEE IF (↑R) TYPED
6647	026402	001002			2\$: BNE	45	;BR IF NOT (↑R)
6648					CMP	#377,SCRACH	;SEE IF ANY DRIVE SELECTED YET
6649					BEQ	35	;BR IF NONE
6650					MOV	#RECAL,P.CMND(R5)	;SET RECALIBRATE COMMAND
6651					JSR	PC,DRVCAL	;RECALIBRATE DRIVE
6652	026404	000137	024512		3\$: JMP	GIVEID	;RESTART ALIGNMENT AID
6653	026410	004737	030616		4\$: JSR	PC,ECOBAD	;ECHO BAD INPUT
6654	026414	004737	030576		JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
6655	026420	005037	005500		6\$: CLR	DRIVE	;ZERO THE DRIVE NUMBER
6656	026424	012700	000001		MOV	#1,RO	;INITIALIZE DRIVE BIT MASK
6657	026430	012737	030360	003046	INTG: MOV	#NEDHDL,A.ABNL	;SET NED ABNORMAL RETURN ADDRESS
6658	026436	050037	005530		BIS	RO,NEWON	;SET ON-LINE BIT FOR THIS DRIVE
6659					;SELECT	CURRENT	DRIVE, CHECK FOR NON-EXISTENT DRIVE (NED) INDICATION
6660	026442	013701	005500		MOV	DRIVE,R1	;LOAD DRIVE NUMBER
6661	026446	116137	005620	003124	MOV	DTYLS(R1),TYPFMT	;LOAD THE RK06-RK07 FLAG
6662	026454	012737	000624	020036	MOV	#624,LCM6	
6663	026462	012737	000631	005736	MOV	#631,PRMLIM+2	
6664	026470	012737	000632	005662	MOV	#632,LC	
6665	026476	012737	000632	020040	MOV	#632,LSTCYL	
6666	026504	012737	000632	005710	MOV	#632,PROFLT+2	
6667	026512	012737	000632	005742	MOV	#632,PRMLIM+6	
6668	026520	012737	000400	020042	MOV	#400,HOLD1	
6669	026526	012737	000025	020044	MOV	#25,HOLD3	
6670	026534	012737	010025	020046	MOV	#10025,HOLD4	
6671	026542	105737	003124		TSTB	TYPFMT	;IS AN RK07 ?
6672	026546	001433			BEQ	95	;BRANCH IF NOT
6673	026550	012737	001450	020036	MOV	#1450,LCM6	
6674	026556	012737	001455	005736	MOV	#1455,PRMLIM+2	
6675	026564	012737	001456	005662	MOV	#1456,LC	
6676	026572	012737	001456	020040	MOV	#1456,LSTCYL	
6677	026600	012737	001456	005710	MOV	#1456,PROFLT+2	
6678	026606	012737	001456	005742	MOV	#1456,PRMLIM+6	
6679	026614	012737	001000	020042	MOV	#1000,HOLD1	
6680	026622	012737	002025	020044	MOV	#2025,HOLD3	
6681	026630	012737	012025	020046	MOV	#12025,HOLD4	
6682	026636	000240			9\$: NOP		;EXIT
6683	026640	004737	032436		JSR	PC,SCNDRV	;CHECK THE SELECTED DRIVE

```

6684 026644 026706          99$          ;ERROR EXIT
6685 026646 000240          NOP
6686 026650 000240          NOP
6687 026652 000240          NOP
6688 026654 113765 005500 000000  MOVB    DRIVE P.DRVN(R5) ;SET DRIVE NUMBER
6689 026662 112765 000101 000001  MOVB    #SELDV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
6690 026670 000137 027150          JMP     23$          ;BRANCH TO EXECUTE THE DRIVE
6691          :          MOV    DRIVE R0          ;GET THE SPECIFIED DRIVE
6692          :          MOV    #-1,HDALI          ;SET UP FOR PROGRAM MARKER
6693          :          JMP     2#TYPED          ;LOAD THE DEVICE TABLE
6694          :          CLR     HDALI          ;CLEAR THIS STATUS
6695 026674 013737 020040 005662  :INTGON: MOV    LSTCYL,LC          ;GET THE MAX CYLINDER SIZE FOR THIS DEVICE
6696 026702 004737 042542          JSR     PC,DRVCAL          ;SELECT THIS DRIVE
6697 026706 022737 000007 005500 99$: CMP    #7,DRIVE          ;SEE IF WE JUST CHECKED DRIVE 7
6698 026714 001405          BEQ     10$          ;BR IF IT WAS DRIVE 7
6699 026716 005237 005500          INC    DRIVE          ;ADD 1 TO DRIVE NUMBER
6700 026722 000241          CLC          ;CLEAR CARRY BEFORE ROTATE
6701 026724 006100          ROL    R0          ;SHIFT BIT POINTER
6702 026726 000640          BR     INTG          ;BR TO SELECT NEXT DRIVE
6703 026730 013701 005530          10$: MOV    NEWON,R1
6704          :          ;***** TO BE REMOVED *****
6705 026734 023737 005530 005526  CMP    NEWON,ONLINE
6706 026742 001002          BNE     93$
6707 026744 000137 026344          JMP     DRVLUP
6708 026750          93$:
6709          :          BEQ     DRVLUP
6710 026750 012703 000001          :          MOV    #1,R3          ;NUMBER OF 200 MILLI-SEC STALLS
6711          :          :LOOP1: MOV    #177777,R0
6712          :          :LOOP2: DEC    R0
6713          :          :          TST    R0
6714          :          :          BNE     LOOP2
6715          :          :          DEC    R3
6716          :          :          BNE     LOOP1
6717          :          :*****
6718 026754 043701 005526          :          BIC    ONLINE,R1          ;GET CHANGED ONLINE BITS
6719 026760 013737 005530 005526  MOV    NEWON,ONLINE          ;UPDATE ONLINE DRIVE BITS
6720 026766 005701          TST    R1          ;SEE IF ANY DRIVE JUST WENT ON-LINE
6721          :          BNE     97$
6722 026770 000137 026344          :          JMP     DRVLUP
6723 026774          97$:
6724          :          BEQ     DRVLUP          ;BR IF NO DRIVE JUST WENT ONLINE
6725          :          ;SERVICE THE DRIVE WHICH WAS JUST SELECTED
6726 026774 012737 044200 003046  MOV    #ERRHDL,A.ABNL          ;RESTORE USUAL ERROR HANDLER ADDRESS
6727 027002 005037 005500          CLR    DRIVE          ;INITIALIZE DRIVE NO.
6728 027006 010103          MOV    R1,R3          ;COPY NEW SELECTED DRIVE BITS
6729 027010 012700 000001          MOV    #1,R0          ;INITIALIZE BIT POINTER
6730 027014 030003          12$: BIT    R0,R3          ;SEE IF THIS BIT IS SET
6731 027016 001005          BNE     14$          ;BR IF THIS BIT SET
6732 027020 005237 005500          INC    DRIVE          ;INCREMENT DRIVE NO.
6733 027024 000241          CLC          ;CLEAR CARRY BEFORE ROTATE
6734 027026 006100          ROL    R0          ;SHIFT BIT POINTER
6735 027030 000771          BR     12$          ;TRY AGAIN
6736 027032 040003          14$: BIC    R0,R3          ;CLEAR OUT THIS BIT
6737 027034 001415          BEQ     16$          ;BR IF ONLY ONE DRIVE SELECTED
6738          :          ;ERROR - MORE THAN ONE DRIVE SELECTED SIMULTANEOUSLY
6739 027036 104401 012163          TYPE   ,MULSEL          ;REPORT ERROR

```

```

6740 027042 042762 000100 000000 BIC #IE,RKCS1(R2) ;INHIBIT RK06 INTERRUPT
6741 027050 000000 HALT ;HALT FOR MANUAL INTERVENTION
6742 ;PRESS 'CONT' TO PROCEED
6743 027052 112765 000177 000001 MOVB #SUBCLR.P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6744 027060 004737 042542 JSR PC,DRVCAL ;CLEAR THE SUBSYSTEM
6745 027064 000137 026266 JMP AUTAL ;RESTART AUTO ALIGNMENT
6746 ;CONTINUE WITH SELECTED DRIVE
6747 027070 020137 005524 16$: CMP R1,SELECT ;SEE IF STILL ALIGNING SAME DRIVE
6748 027074 001505 BEQ 26$ ;BR IF SAME DRIVE
6749 ;PROCEED WITH NEWLY-SELECTED DRIVE
6750 027076 022737 000377 005534 CMP #377,SCRACH ;SEE IF THIS IS FIRST DRIVE SELECTED
6751 027104 001421 BEQ 23$ ;BR IF FIRST DRIVE
6752 027106 113765 005534 000000 MOVB SCRACH,P.DRVN(R5) ;GET LAST DRIVE NUMBER
6753 027114 112765 000141 000001 MOVB #RDSTAT.P.CMND(R5) ;SET READ STATUS COMMAND
6754 027122 004737 042542 JSR PC,DRVCAL ;READ STATUS OF PREVIOUS DRIVE
6755 027126 032765 000040 000044 BIT #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLOADED
6756 027134 001405 BEQ 23$ ;BR IF HEADS NOT UNLOADED
6757 027136 112765 000111 000001 MOVB #SRTSPL.P.CMND(R5) ;SET START SPINDLE COMMAND
6758 027144 004737 042542 JSR PC,DRVCAL ;START SPINDLE ON PREVIOUS DRIVE
6759 027150 113737 005500 005534 23$: MOVB DRIVE,SCRACH ;STORE DRIVE NO. FOR LATER CLEANUP STRT SPL
6760 027156 010137 005524 MOV R1,SELECT ;UPDATE SELECTED DRIVE NUMBER
6761 027162 013701 005500 MOV DRIVE,R1 ;GET DRIVE NUMBER
6762 027166 052701 000060 BIS #'0,R1 ;CONVERT TO ASCII
6763 027172 110137 013050 MOVB R1,DRVSEL ;GET DRIVE NUMBER INTO OUTPUT BUFFER
6764 027176 104401 013043 TYPE DRIVESEL ;TYPE "DRIVE X SELECTED"
6765 027202 004737 033252 JSR PC,CHKDRV ;CHECK DRIVE FOR RDY, WRITE PROT.
6766 027206 026266 AUTAL ;ERROR RETURN ADDRESS
6767 027210 112737 000377 011363 MOVB #377,UNLOD ;SET DRIVE UNLOADED INDICATOR
6768 027216 012737 000000 005520 MOV #0,TRACK ;SET TO TRACK 0
6769 ; MOV #2,TRACK ;SET TRACK = 2
6770 027224 113765 005500 000000 ; MOVB DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
6771 027232 000421 BR 32$ ;SKIP THE REST PROGRAM FLOW
6772 027234 105737 011364 24$: TSTB VERIFY ;SEE IF AUTO VERIFY
6773 027240 001016 BNE 32$ ;SKIP UNLOAD IF VERIFY
6774 027242 112765 000107 000001 MOVB #UNLOD.P.CMND(R5) ;SET UNLOAD COMMAND
6775 027250 004737 042542 JSR PC,DRVCAL ;UNLOAD THE DRIVE
6776 027254 112765 000141 000001 MOVB #RDSTAT.P.CMND(R5) ;SET READ STATUS COMMAND
6777 027262 004737 042542 25$: JSR PC,DRVCAL ;READ STATUS OF DRIVE
6778 027266 032765 000040 000044 BIT #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLOADED YET
6779 027274 001772 BEQ 25$ ;BR IF NOT YET
6780 027276 112765 000177 000001 32$: MOVB #SUBCLR.P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6781 027304 004737 042542 JSR PC,DRVCAL ;CLEAR THE S.S.
6782 JMP DRVLUP ;GO BACK TO SCAN DRIVES AGAIN
6783 ;PROCEED WITH SAME DRIVE
6784 027310 113765 005500 000000 26$: MOVB DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6785 027316 105137 011363 COMB UNLOD ;COMPLEMENT UNLOAD INDICATOR
6786 ; BNE 24$ ;BR IF DRIVE SHOULD BE UNLOADED
6787 027322 105737 011364 ; TSTB VERIFY ;SEE IF AUTO VERIFY
6788 027326 001005 BNE 33$ ;SKIP START SPL IF VERIFY
6789 027330 112765 000113 000001 MOVB #RECAL.P.CMND(R5) ;SET RECAL COMMAND
6790 ; MOVB #SRTSPL.P.CMND(R5) ;SET START SPINDLE COMMAND
6791 027336 004737 042542 JSR PC,DRVCAL ;START SPINDLE ON THIS DRIVE
6792 027342 022737 000002 005520 33$: CMP #2,TRACK ;SEE IF LAST TRACK WAS 2
6793 027350 103005 BHS 30$ ;NOT EXCEEDS HEAD 2
6794 ; BNE 28$ ;BR IF IT WAS NOT 2
6795 027352 005037 005520 CLR TRACK ;SET TRACK = 0

```

6796	027356	000402			BR	30\$;GO SEEK TO ALIGNMENT CYLINDER
6797	027360	005237	005520		INC	TRACK		;INCREMENT TRACK NUMBER
6798	027364	113765	005520	000005	30\$:	MOVB	TRACK,P.TRCK(R5)	;SET TRACK NO.
6799	027372	004737	033512		JSR	PC,ALNSEK		;SEEK IN INCREMENTS TO ALIGN. CYL
6800	027376	104401	012510		TYPE	HEDPOS		;TYPE HEADS POSITIONED MSG
6801	027402	013701	005520		MOV	↑TRACK,R1		;GET CURRENT TRACK NO.
6802	027406	152701	000060		BISB	#'0,R1		;CONVERT TO ASCII
6803	027412	110137	012704		MOVB	R1,HEADNO		;GET TRACK INTO MSG BUFFER
6804	027416	104401	012677		TYPE	HEDSEL		;TYPE "HEAD X SELECTED"
6305	027422	005237	005520		INC	↑TRACK		;INCREMENT THE TRACK NUMBER
6806	027426	022737	000003	005520	CMP	#3,TRACK		;ALL TRACK ARE DONE?
6807	027434	101353			BHI	30\$;BRANCH IF NOT
6808	027436	004737	030576		46\$:	JSR	PC,PREPKB	;READ THE KEYBOARD
6809	027442	005737	005522		47\$:	TST	INTCHR	;READ ANY THING YET ?
6810	027446	001775			BEQ	47\$;BRANCH IF NOT
6811	027450	123727	005522	000032	CMPB	INTCHR,#032		;Z ?
6812	027456	001002			BNE	48\$;BRANCH IF NOT
6813	027460	000137	026144		JMP	AUTO		
6814	027464	123727	005522	000022	48\$:	CMPB	↑NTCHR,#022	;R ?
6815	027472	001002			BNE	49\$		
6816	027474	000137	026144		JMP	AUTO		
6817	027500	123727	005522	000003	49\$:	CMPB	INTCHR,#003	;C ?
6818	027506	001353			BNE	46\$;WAIT
6819	027510	000137	026144		JMP	AUTO		;TO NEXT DRIVE
6820	027514	000137	026344		JMP	DRVLUP		;GO BACK TO SCAN DRIVES AGAIN
6821								
6822								
6823								
6824	027520							
6825	027520	004737	032310		AUTEX:	JSR	PC,INITSS	;INITIALIZE SUBSYSTEM
6826	027524	004737	030576		JSR	PC,PREPKB		;PREPARE FOR KBD INPUT
6827	027530	104401	013066		TYPE	AUTEXR		;ASK FOR SHORT OR LONG SEEK EXERCISES
6828	027534	005737	005522	1\$:	TST	INTCHR		;SEE IF ANY INPUT
6829	027540	001775			BEQ	1\$;BR IF NO INPUT
6830	027542	023727	005522	000022	CMP	INTCHR,#022		;SEE IF (↑R) TYPED
6831	027550	001002			BNE	2\$;BR IF NOT (↑R)
6832	027552	000137	024512		JMP	GIVEID		;JUMP TO RESTART ALIGNMENT AID
6833	027556	023727	005522	000032	2\$:	CMP	INTCHR,#032	;SEE IF (↑Z) TYPED
6834	027564	001002			BNE	3\$;BR IF NOT (↑Z)
6835	027566	000137	026144		JMP	AUTO		;JUMP TO RESTART AUTO MODE
6836	027572	023727	005522	000123	3\$:	CMP	INTCHR,#'S	;SEE IF SHORT EXERCISES DESIRED
6837	027600	001004			BNE	4\$;BR IF NOT SHORT
6838	027602	012737	002734	005534	MOV	#RNDST,SCRACH		;SET 1 MINUTE SEEK COUNT
6839	027610	000413			BR	6\$;PROCEED WITH SHORT EXERCISES
6840	027612	023727	005522	000114	4\$:	CMP	INTCHR,#'L	;SEE IF LONG EXERCISES DESIRED
6841	027620	001004			BNE	5\$;BR IF NOT LONG
6842	027622	012737	016514	005534	MOV	#RNDLNG,SCRACH		;SET 5 MINUTE SEEK COUNT
6843	027630	000403			BR	6\$;PROCEED WITH LONG EXERCISES
6844	027632	004737	030616		5\$:	JSR	PC,ECOBAD	;ECHO BAD INPUT
6845	027636	000730			BR	AUTEX		;BR TO ASK AGAIN
6846	027640	005037	005500		6\$:	CLR	DRIVE	;SET DRIVE = 0
6847					;SELECT	CURRENT	DRIVE AND CHECK	FOR NON-EXISTENT DRIVE INDICATION
6848	027644	012737	030360	003046	8\$:	MOV	#NEDHOL,A.ABNL	;SET NED ABNORMAL RETURN ADDRESS
6849	027652	113765	005500	000000	MOVB	DRIVE,P.DRVN(R5)		;SET DRIVE NUMBER
6850	027660	013701	005500		MOV	DRIVE,R1		;LOAD THE DRIVE NUMBER
6851	027664	116137	005620	003124	MOVB	DTYLS↑(R1),TYPFMT		;LOAD THE RK06-RK07 FLAG

6852	027672	012737	000624	020036	MOV	#624,LCM6	
6853	027700	012737	000631	005736	MOV	#631,PRMLIM+2	
6854	027706	012737	000632	005662	MOV	#632,LC	
6855	027714	012737	000632	020040	MOV	#632,LSTCYL	
6856	027722	012737	000632	005710	MOV	#632,PRDFLT+2	
6857	027730	012737	000632	005742	MOV	#632,PRMLIM+6	
6858	027736	012737	000400	020042	MOV	#400,HOLD1	
6859	027744	012737	000625	020044	MOV	#25,HOLD3	
6860	027752	012737	010025	020046	MOV	#10025,HOLD4	
6861	027760	105737	003124		TSTB	TYPFMT	; IS AN RK07 ?
6862	027764	001433			BEQ	9\$; BRANCH IF NOT
6863	027766	012737	001450	020036	MOV	#1450,LCM6	
6864	027774	012737	001455	005736	MOV	#1455,PRMLIM+2	
6865	030002	012737	001456	005662	MOV	#1456,LC	
6866	030010	012737	001456	020040	MOV	#1456,LSTCYL	
6867	030016	012737	001456	005710	MOV	#1456,PRDFLT+2	
6868	030024	012737	001456	005742	MOV	#1456,PRMLIM+6	
6869	030032	012737	001000	020042	MOV	#1000,HOLD1	
6870	030040	012737	002025	020044	MOV	#2025,HOLD3	
6871	030046	012737	012025	020046	MOV	#12025,HOLD4	
6872	030054	000240			9\$: NOP		; EXIT
6873	030056	004737	032436		JSR	PC,SCNDRV	
6874	030062	030322			18\$		
6875	030064	000240			NOP		
6876	030066	000240			NOP		
6877	030070	000240			NOP		
6878	030072	112765	000101	000001	MOVB	#SELDRV,P.CMND(R5)	; SET SELECT DRIVE COMMAND
6879	030100	012737	000377	005530	MOV	#377,NEWON	; INITIALIZE ON-LINE INDICATOR
6880	030106	004737	042542		JSR	PC,DRVCL	; SELECT THIS DRIVE
6881	030112	012737	044200	003046	MOV	#ERRHDL,A.ABNL	; RESTORE ABNORMAL DRIVER RETURN ADDRESS
6882	030120	022737	000377	005530	CMP	#377,NEWON	; SEE IF NED INDICATION ON THIS DRIVE
6883	030126	001075			BNE	18\$; BR IF NED, TO SKIP THIS DRIVE
6884							
6885	030130	004737	033252		JSR	PC,CHKDRV	; CHECK THIS DRIVE
6886	030134	027520			AUTEX		; ERROR RETURN ADDRESS FOR CHKDRV
6887							
6888	030136	013701	005500		MOV	DRIVE R1	; GET DRIVE NUMBER
6889	030142	052701	000060		BIS	#0,R1	; CONVERT TO ASCII
6890	030146	110137	013176		MOVB	R1,AUTODR	; GET DRIVE NUMBER INTO OUTPUT BUFFER
6891	030152	104401	013157		TYPE	AUTOEX	; TYPE "EXERCISING DRIVE X"
6892	030156	112765	000117	000001	MOVB	#SEEK,P.CMND(R5)	; SET SEEK COMMAND
6893	030164	013700	005534		MOV	SCRACH,R0	; SEEK COUNT
6894	030170	004737	030576		JSR	PC,PREPKB	; PREPARE FOR POSSIBLE KBD INPUT
6895	030174	004737	035666		JSR	PC,RNDADR	; SELECT RANDOM CYLINDER ADDRESS
6896	030200	013765	005504	000002	MOV	CYLNDR,P.CYLN(R5)	; SET RANDOM CYLINDER ADDRESS
6897	030206	004737	042542		JSR	PC,DRVCL	; DO A RANDOM SEEK
6898	030212	005737	005522		TST	INTCHR	; SEE IF ANY KBD INPUT
6899	030216	001432			BEQ	16\$; BR IF NO INPUT
6900	030220	023727	005522	000022	CMP	INTCHR,#022	; SEE IF (↑R) TYPED
6901	030226	001007			BNE	12\$; BR IF NOT (↑R)
6902	030230	112765	000113	000001	MOVB	#RECAL,P.CMND(R5)	; SET RECALIBRATE COMMAND
6903	030236	004737	042542		JSR	PC,DRVCL	; RECALIBRATE DRIVE
6904	030242	000137	024512		JMP	GIVEID	; JUMP TO RESTART ALIGNMENT AID
6905	030246	023727	005522	000032	12\$: CMP	INTCHR,#032	; SEE IF (↑Z) TYPED
6906	030254	001007			BNE	14\$; BR IF NOT (↑Z)
6907	030256	112765	000113	000001	MOVB	#RECAL,P.CMND(R5)	; SET RECALIBRATE COMMAND

```

6908 030264 004737 042542      JSR    PC,DRVCL      ;RECALIBRATE DRIVE
6909 030270 000137 026144      JMP    AUTO          ;JUMP TO RESTART AUTO MODE
6910 030274 004737 030616      14$:   JSR    PC,ECOBAD     ;ECHO BAD INPUT
6911 030300 004737 030576      JSR    PC,PREPKB     ;ENABLE KBD INPUT AGAIN
6912 030304 005300              16$:   DEC    RO              ;DECREMENT SEEK COUNT
6913 030306 001330              BNE    10$              ;BR IF NOT DONE WITH THIS DRIVE
6914 030310 112765 000113 000001  MOVB   #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6915 030316 004737 042542      JSR    PC,DRVCL      ;RECALIBRATE DRIVE
6916 030322 005237 005500      18$:   INC    DRIVE          ;INCREMENT DRIVE NUMBER
6917 030326 022737 000010 005500  CMP    #10,DRIVE      ;SEE IF DONE WITH ALL DRIVES
6918 030334 001402              BEQ    99$              ;
6919 030336 000137 027644              JMP    8$              ;
6920 030342              99$:   ;
6921 030342 042777 000100 150574  BNE    8$              ;BR IF MORE DRIVES TO EXERCISE YET
6922 030350 104401 001310      BIC    #BIT6,2$TKS    ;DISABLE KBD INTERRUPT
6923 030354 000137 026144      TYPE   $BELL          ;RING BELL AT END OF AUTO-EXERCISES
6924 030354 000137 026144      JMP    AUTO          ;RESTART AUTO MODE
6925
6926
6927 ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
6928 ; WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
6929 ; (NED = NON-EXISTENT DRIVE)
6930 030360 032765 010000 000020 NEDHDL: BIT    #NED,P.CS2(R5) ;SEE IF NED ON DRIVE SELECT
6931 030366 001002              BNE    1$              ;BR IF NED SET
6932 030370 000137 044200      JMP    ERRHDL          ;GO HANDLE OTHER ERROR
6933 030374 010046      1$:   MOV    RO,-(SP)          ;SAVE RO,R1
6934 030376 010146      MOV    R1,-(SP)
6935 030400 012700 000001      MOV    #1,RO              ;SET BIT POINTER
6936 030404 005001              CLR    R1              ;CLEAR COUNTER
6937 030406 120137 005500      2$:   CMPB   R1,DRIVE      ;SEE IF R1 = CURRENT DRIVE NUMBER
6938 030412 001403              BEQ    3$              ;BR IF =
6939 030414 005201              INC    R1              ;INCREMENT COUNTER
6940 030416 006300              ASL    RO              ;SHIFT BIT POINTER
6941 030420 000772              BR     2$              ;TRY AGAIN
6942 030422 040037 005530      3$:   BIC    RO,NEWON      ;CLEAR ONLINE BIT FOR THIS DRIVE
6943 030426 012601              MOV    (SP)+,R1          ;RESTORE RO,R1
6944 030430 012600              MOV    (SP)+,RO
6945 030432 000137 046356      JMP    RETNML          ;TAKE NORMAL RETURN
6946
6947
6948 ;*****
6949 ;$BTTL KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
6950 ;*****
6951
6952 030436 010146      KBDHDL: MOV    R1,-(SP)          ;SAVE R1 ON STACK
6953 030440 017701 150502      MOV    2$TKB,R1          ;READ A CHAR FROM KBD BUFFER
6954 030444 042701 177600      BIC    #177600,R1          ;MASK OUT UNUSED BITS
6955 030450 120127 000172      CMPB   R1,#172          ;SEE IF LOWER CASE TYPED
6956 030454 003005      BGT     20$              ;BR IF NOT
6957 030456 120127 000141      CMPB   R1,#141          ;BR IF NOT
6958 030462 002402      BLT     20$              ;MAKE IT UPPER CASE
6959 030464 042701 000040      BIC    #BITS,R1          ;SAVE INPUT CHARACTER
6960 030470 010137 005522      20$:  MOV    R1,INCHR          ;SEE IF (↑C) TYPED
6961 030474 122701 000003      CMPB   #003,R1          ;BR IF NOT (↑C)
6962 030500 001007      BNE    2$              ;ECHO (↑C)
6963 0 0502 104401 013246      TYPE   ,CNTRLC

```

```

6964 030506 042777 000100 150430 1$: BIC #BIT6,2$TKS ;CLEAR KBD INTERRUPT ENABLE BIT
6965 030514 012601 ;RESTORE R1
6966 030516 000002 RTI ;RETURN
6967 030520 122701 000032 2$: CMPB #032,R1 ;SEE IF (↑Z) TYPED
6968 030524 001003 BNE 3$ ;BR IF NOT (↑Z)
6969 030526 104401 013253 TYPE ,CNTRLZ ;ECHO (↑Z)
6970 030532 000765 BR 1$ ;RETURN
6971 030534 122701 000022 3$: CMPB #022,R1 ;SEE IF (↑R) TYPED
6972 030540 001003 BNE 4$ ;BR IF NOT (↑R)
6973 030542 104401 013260 TYPE ,CNTRLR ;ECHO (↑R)
6974 030546 000757 BR 1$ ;RETURN
6975 030550 122701 000007 4$: CMPB #007,R1 ;SEE IF (↑G) TYPED
6976 030554 001003 BNE 5$ ;BR IF NOT (↑G)
6977 030556 104401 013272 TYPE ,CNTRLG ;ECHO (↑G)
6978 030562 000751 BR 1$ ;RETURN
6979 030564 104401 005522 5$: TYPE ,INTCHR ;ECHO INPUT
6980 030570 104401 001315 TYPE ,$CRLF ;DO <CR> AND <LF>
6981 030574 000744 BR 1$ ;RETURN
6982
6983
6984
6985 ;*****
6986 ;SBTTL PREPKB - PREPARE FOR KEYBOARD INPUT
6987 ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
6988 ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
6989 ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
6990 ;*ENABLES KBD INTERRUPT.
6991 ;* CALL:
6992 ;* JSR PC,PREPKB
6993 ;*****
6994
6995 030576 005077 150344 PREPKB: CLR 2$TKB ;CLEAR KBD BUFFER AND DONE BIT
6996 030602 005037 005522 CLR INTCHR ;CLEAR TTY INPUT WORD
6997 030606 052777 000100 150330 BIS #BIT6,2$TKS ;ENABLE KBD INTERRUPT
6998 030614 000207 RTS PC ;RETURN
6999
7000
7001
7002 ;*****
7003 ;SBTTL ECOBAD - ECHO BAD TTY INPUT
7004 ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
7005 ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
7006 ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
7007 ;*AND <LF> ARE DONE.
7008 ;* CALL - JSR PC,ECOBAD
7009 ;*****
7010
7011 030616 104401 005522 ECOBAD: TYPE ,INTCHR ;ECHO BAD CHARACTER
7012 030622 104401 001314 TYPE , $QUES ;TYPE '<?>' AND <CR>, <LF>
7013 030626 000207 RTS PC ;RETURN
7014
7015
7016
7017 ;*****
7018 ;$SIZMEM - SIZE MEMORY, SET LIMITS
7019 ;*THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE. AND

```

```

7020      ;*IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
7021      ;*IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D),
7022      ;*AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70).
7023      ;*****
7024      SIZMEM: SAVREG      ;SAVE R0-R5
7025      MOV      #200,$KT11 ;SET MEM MGT KEY FOR $SIZE
7026      JSR      PC,$SIZE  ;SIZE MEMORY
7027      TST      $KT11     ;SEE IF MEM MGT PRESENT
7028      BMI      8$        ;BR IF MEM MGT PRESENT
7029      MOV      $LSTAD,MAHILM ;SET MEM LIMIT LO BITS
7030      CLR      MAHILM+2    ;CLEAR MEM LIMIT HI BITS
7031      BR       16$        ;GO TYPE LAST ADDRESS
7032      ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
7033      8$: MOV      $LSTBK,R0 ;LO BITS
7034      CLR      R1          ;HI BITS
7035      MOV      #6,R2      ;SET LOOP COUNT = 6
7036      12$: CLC          ;ROTATE LOOP
7037      ROL      R0
7038      ROL      R1
7039      DEC      R2
7040      BNE      12$
7041      ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
7042      ADD      $LSTAD,R0    ;ADD LO BITS
7043      ADC      R1          ;HI BITS
7044      MOV      R0,MAHILM   ;SET LO BITS OF MEM LIMIT
7045      MOV      R1,MAHILM+2 ;SET HI BITS OF MEM LIMIT
7046      CLC
7047      ROL      R0
7048      ROL      R1
7049      ROL      R0
7050      ROL      R1
7051      ROL      R0
7052      ROL      R1
7053      CMP      R1,#37
7054      BLO      16$
7055      MOV      #1,UBMPRS   ;SET "UNIBUS MAP PRESENT" FLAG
7056      ;TYPE "LAST PHYS MEM ADR = XXXXXXXX"
7057      16$: TYPE    ,LSTMEM  ;TYPE "LAST PHYS MEM ADR ="
7058      MOV      MAHILM,-(SP) ;PUT POINTER ON STACK
7059      JSR      PC,@#$DB20  ;CONVERT BINARY TO OCTAL ASCII
7060      JSR      PC,@#$SUPRS ;TYPE "XXXXXXXX"
7061      TYPE     , $CRLF     ;TYPE <CR>,<LF>
7062      TYPE     , $CRLF
7063      RESREG
7064      RTS      PC          ;RESTORE R0-R5
7065      ;RETURN
7066
7067      ;*****
7068      ;* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
7069      ;*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
7070      ;*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
7071      ;*TOP OF STACK.
7072      ;*****
7073      GETPRM: MOV      2(SP),-(SP) ;GET OLD VALUE
7074      TYPOS
7075      ;TYPE IT

```

7076	031024	006			.BYTE	6		:SIX DIGITS
7077	031025	000			.BYTE	0		:SUPPRESS LEADING ZEROS
7078	031026	104401	007475		TYPE	NEWMSG		:TYPE "NEW" = "
7079	031032	004737	034034		JSR	PC, RDCHRS		:READ NEW VALUE FROM KBD
7080	031036	031070			7\$: (↑C) RETURN ADDRESS
7081	031040	031070			7\$: (↑Z) RETURN ADDRESS
7082	031042	031070			7\$: (↑U) OR ERROR RETURN ADDRESS
7083	031044	005700			TST	RO		:SEE IF ANY CHARS TYPED
7084	031046	001001			BNE	4\$:BR IF YES
7085	031050	000207			RTS	PC		:RETURN - OLD VALUE UNCHANGED
7086	031052	020027	000006	4\$:	CMP	RO, #6		:SEE IF > 6 CHARS TYPED
7087	031056	003407			BLE	8\$:BR IF NOT BAD
7088	031060	104401	005264	6\$:	TYPE	, BUFFO		:ECHO BAD INPUT
7089	031064	104401	001314		TYPE	, \$QUES		
7090	031070	162716	000010	7\$:	SUB	#10, (SP)		:FIX ERROR RETURN PC
7091	031074	000207			RTS	PC		:ERROR RETURN
7092	031076	012746	005264	8\$:	MOV	#BUFFO, -(SP)		:PUT POINTER TO CHARS ON STACK
7093	031102	004737	054500		JSR	PC, OCTBIN		:CONVERT DIGITS TO BINARY
7094	031106	031060			6\$:ERROR RETURN ADDRESS
7095	031110	012600			MOV	(SP)+, RO		:GET NEW BINARY VALUE
7096	031112	005737	054632		TST	\$HI OCT		:SEE IF HI BITS ARE 0
7097	031116	001360			BNE	6\$:BR IF NOT
7098	031120	010066	000002		MOV	RO, 2(SP)		:PUT NEW VALUE ON STACK
7099	031124	000207			RTS	PC		:RETURN
7100								
7101								
7102								
7103								
7104								
7105								
7106								
7107								
7108								
7109								
7110	031126	022737	000176	001140	GSWRG:	CMP	#SWREG, SWR	:SEE IF SOFTWARE SWR SELECTED
7111	031134	001010			BNE	6\$:BR IF NOT
7112	031136	013746	000176		MOV	SWREG, -(SP)		:PUT OLD VALUE ON STACK
7113	031142	104401	007466		TYPE	SWMSG		:TYPE "SWR" = "
7114	031146	004737	031016		JSR	PC, GETPRM		:TYPE OLD, GET NEW SWREG VALUE
7115	031152	012637	000176		MOV	(SP)+, SWREG		:STORE NEW VALUE
7116	031156	000207		6\$:	RTS	PC		:RETURN
7117								
7118								
7119								
7120								
7121								
7122								
7123								
7124								
7125								
7126								
7127								
7128								
7129								
7130	031160	104407			INITMM:	SAVREG		:SAVE RO-R5
7131	031162	005001			CLR	R1		:INIT FOR PAR LOADING

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 141
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0140

7132	031164	012702	172340		MOV	#KIPAR0,R2	;ADDR OF FIRST PAR
7133	031170	012703	000010		MOV	#8,R3	;LOAD 8 PAR'S AND 8 PDR'S
7134	031174	012762	077406	177740 4\$:	MOV	#77406,-40(R2)	;PDR = 4K UP,READ/WRITE
7135	031202	010122			MOV	R1,(R2)+	;LOAD A PAR
7136	031204	062701	000200		ADD	#200,R1	;UPDATE FOR NEXT PAR
7137	031210	005303			DEC	R3	;DECREMENT LOOP COUNTER
7138	031212	071370			BNE	4\$;LOOP UNTIL ALL 8 ARE LOADED
7139	031214	012742	177600		MOV	#177600,-(R2)	;SET UP KIPAR7 FOR I/O PAGE
7140	031220	105737	003143		TSTB	UBMPRS	;SEE IF 22-BIT ADDRESSES
7141	031224	001403			BEQ	10\$;BR IF NOT
7142	031226	012737	000060	172516	MOV	#60,#SR3	;ENABLE 22-BIT MODE,AND UNIBUS MAP
7143	031234	012737	001000	177572 10\$:	MOV	#BIT9,#SRO	;ENABLE KT.1 TRAPS
7144	031242	104410			RESREG		;RESTORE R0-R5
7145	031244	000207			RTS	PC	;RETURN
7146							
7147							
7148							
7149							
7150							
7151							

;;*****
;;SERVICE ROUTINE FOR MEM MGT TRAPS
;;*****

L11

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 142
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0141

7152	031246	004737	043720		KTERHD: JSR	PC, REPSUP	; GATHER STATUS FOR PRINTOUT
7153	031252	013737	177572	001174	MOV	@SR0, \$REG5	; GET SR0
7154	031260	013737	177574	001176	MOV	@SR1, \$REG6	; GET SR1
7155	031266	013737	177576	001200	MOV	@SR2, \$REG7	; GET SR2
7156	031274	012737	031320	000004	MOV	#8\$, @ERRVEC	; SET TIME-OUT VECTOR

M11

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 143
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0142

7157	031302	012737	000340	000006	MOV	#PR7,2*ERRVEC+2	
7158	031310	013737	172516	001202	MOV	2*SR3,\$REG10	;GET SR3
7159	031316	000406			BR	10\$	
7160	031320	022626			CMP	(SP)+,(SP)+	;CLEAN UP STACK

7161 031322 112737 000003
7162 031330 105037 064272
7163 031334 104121
7164 031336 000137 046150
7165
7166
7167
7168
7169
7170

065632

MOV B #3,DF30+18. ;FIX PRINTOUT FOR NO SR3
CLRB DH704+11.
10\$: ERROR 121 ;KT11 FAILURE
JMP HLT PRG ;ABORT !!!

7171
7172
7173
7174
7175
7176
7177
7178
7179
7180
7181
7182
7183
7184
7185
7186
7187
7188
7189
7190
7191
7192

031342 010146
031344 013746 000004
031350 013746 000006
031354 012737 031376 000004
031362 012701 172100
031366 005011
031370 012711 000001
031374 000401
031376 022626
031400 062701 000002
031404 020127 172140
031410 001366
031412 012637 000006
031416 012637 000004
031422 012601
031424 000207

000004

;ENBCSR - ENABLE MEMORY CSR'S FOR PARITY ERRORS

ENBCSR: MOV R1, -(SP) ;SAVE R1
MOV @ERRVEC, -(SP) ;SAVE OLD VECTORS
MOV @ERRVEC+2, -(SP)
MOV #8\$, @ERRVEC ;SET TIME-OUT VECTOR
MOV @MEMCSR, R1 ;ADRS OF MEMORY CSR'S
4\$: CLR (R1) ;ZERO THIS CSR
MOV @BIT0, (R1) ;SET ENABLE IN THIS CSR
10\$
BR
8\$: CMP (SP)+, (SP)+ ;CLEAN UP THE STACK
10\$: ADD #2, R1 ;INCREMENT TO NEXT CSR ADRS
CMP R1, @MEMCSR+40 ;SEE IF DONE CHECKING YET
BNE 4\$;BR IF NOT
MOV (SP)+, @ERRVEC+2 ;RESTORE OLD VECTORS
MOV (SP)+, @ERRVEC
MOV (SP)+, R1 ;RESTORE R1
RTS PC ;RETURN

7193
7194
7195
7196
7197
7198
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216

031426 010146
031430 013746 000004
031434 013746 000006
031440 004737 043720
031444 016637 000006 001174
031452 012737 031576 000004
031460 012737 000340 000006
031466 013737 177740 001176
031474 013737 177742 001200
031502 013737 177744 001202
031510 012737 064326 065654
031516 112737 000004 065656
031524 104122
031526 005737 001200
031532 001011
031534 023727 001176 073660
031542 103005
031544 105737 003144
031550 001002
031552 000137 046150
031556 012637 JU0006
031562 012637 000004
031566 012601

;SERVICE ROUTINE FOR MEM PARITY ERRORS

MPERHD: MOV R1, -(SP) ;SAVE R1
MOV @ERRVEC, -(SP) ;SAVE OLD VECTORS
MOV @ERRVEC+2, -(SP)
JSR PC, REPSUP ;GATHER STATUS FOR PRINTOUT
MOV 6(SP), \$REG5 ;GET PC OF ERROR
MOV #10\$, @ERRVEC ;SET T.O. VECTOR
MOV @PR7, @ERRVEC+2
;HANDLE 11/70 MEMORY PARITY ERROR
MOV @LOERAD, \$REG6 ;LOW ERROR ADRS REG
MOV @HIERAD, \$REG7 ;HI ERROR ADRS REG
MOV @MEMSYS, \$REG10 ;MEMORY SYSTEM REG
MOV @DH707, DF31+16. ;FIX ERROR MSG FOR 11/70
MOV B #4, DF31+18.
ERROR 122 ;11/70 MEM PARITY ERROR
TST \$REG7 ;SEE IF BAD MEMORY IS IN PROGRAM AREA
BNE 6\$;BR IF NOT
CMP \$REG6, @RWBUF+6000 ;SEE IF BAD MEM IS IN PROG. AREA
BNE 6\$;BR IF NOT
4\$: TSTB MEMABT ;SEE IF ABORT DESIRED
BNE 6\$;BR IF NOT
JMP HLT PRG ;ABORT !!!
6\$: MOV (SP)+, @ERRVEC+2 ;RESTORE T.O. VECTOR
MOV (SP)+, @ERRVEC
MOV (SP)+, R1 ;RESTORE R1

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 145
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0144

7217	031570	004737	031342		JSR	PC,ENBCSR	;GO CLEAR AND ENABLE CSR'S
7218	031574	000002			RTI		;RETURN
7219					;HANDLE	ALL OTHER MEMORY PARITY ERRORS (NON-11/70)	
7220	031576	022626			10\$:	CMP	(SP)+,(SP)+
7221	031600	012737	031630	000004		MOV	#14\$,\$ERRVEC
7222	031606	012701	172100			MOV	#MEMCSR,R1
7223	031612	011137	001200		12\$:	MOV	(R1),\$REG7
7224	031616	100005				BPL	16\$
7225	031620	010137	001176			MOV	R1,\$REG6
7226	031624	104122				ERROR	122
7227	031626	000746				BR	4\$
7228	031630	022626			14\$:	CMP	(SP)+,(SP)+
7229	031632	062701	000002		16\$:	ADD	#2,R1
7230	031636	020127	172140			CMP	R1,#MEMCSR+40
7231	031642	103763				BLO	12\$
7232	031644	000744				BR	6\$
7233							;RETURN
7234							
7235							
7236							
7237							
7238							
7239							
7240							
7241							
7242	031646	104407					
7243	031650	004737	031160		PREPAR: SAVREG		;SAVE R0-R5
7244	031654	013700	005604		JSR	PC,INITMM	;INIT MEM MGT REGISTERS
7245	031660	042700	017777		MOV	PMA,R0	;LO BITS OF MA
7246	031664	013701	005606		BIC	#17777,R0	;MASK FOR BITS 13-15
7247	031670	010003			MOV	PMA+2,R1	;HI BITS OF MA
7248	031672	010104			MOV	R0,R3	;SAVE THESE BITS
7249	031674	006100			MOV	R1,R4	
7250	031676	006101			ROL	R0	;GET MA BITS 13-21 INTO R1 BITS 7-15
7251	031700	006100			ROL	R1	
7252	031702	006101			ROL	R0	
7253	031704	000301			ROL	R1	
7254	031706	006100			SWAB	R1	
7255	031710	106001			ROL	R0	
7256	031712	010137	003176		RORB	R1	
7257	031716	105737	003143		MOV	R1,SAVPAR	;CONSTANT FOR LOADING PAR6 LATER
7258	031722	001420			TSTB	UBMPRS	;SEE IF UNIBUS MAP ENABLED
7259					BEQ	9\$;BR IF NOT (NO UNIBUS MAPPING)
7260	031724	012700	170200		;PREPARE FOR 22-BIT ADDRESSING -		LOAD UNIBUS MAP REGISTERS
7261	031730	012701	000037		MOV	#MAPLOO,R0	;STARTING ADDR OF MAP REGISTERS
7262	031734	010320			MOV	#31,R1	;SET REGISTER COUNTER
7263	031736	010420			4\$:	MOV	R3,(R0)+
7264	031740	062703	020000			MOV	R4,(R0)+
7265	031744	005504				ADD	#20000,R3
7266	031746	077106				ADC	R4
7267	031750	042765	160000	000010		SOB	R1,4\$
7268	031756	142765	000003	000007		BIC	#160000,P.BALO(R5)
7269	031764	104410				BICB	#3,P.BAHI(R5)
7270	031766	000207			9\$:	RESREG	;RESTORE R0-R5
7271						RTS	PC
7272							;RETURN

```

7273
7274
7275
7276
7277
7278 031770 005737 005576
7279 031774 001404
7280 031776 012737 160000 005540
7281 032004 000412
7282 032006 023727 005574 157776
7283 032014 103370
7284 032016 013737 005574 005540
7285 032024 062737 000002 005540
7286 032032 162737 006000 005540
7287 032040 000207
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302 032042 104407
7303 032044 005004
7304 032046 000410
7305 032050 104407
7306 032052 105037 003133
7307 032056 105737 003134
7308 032062 001425
7309 032064 012704 000001
7310 032070 005737 000170
7311 032074 001020
7312 032076 012702 003000
7313 032102 013703 005540
7314 032106 013700 005542
7315 032112 013701 005544
7316 032116 005737 057732
7317 032122 100417
7318 032124 005704
7319 032126 001005
7320 032130 012320
7321 032132 005302
7322 032134 001375
7323 032136 104410
7324 032140 000207
7325 032142 062700 006000
7326 032146 062703 006000
7327 032152 014043
7328 032154 005302

*****
* FNDXDP - FIND STARTING ADR OF XXDP LOADER , AND STORE IT
* IN XXDPAD.
*****
FNDXDP: TST MAHILM+2 ;TEST HI BITS OF UPPER MEMORY LIMIT
        BEQ 6$ ;BR IF HI BITS ARE 0
        MOV #160000,XXDPAD ;START OF 2BK IS END OF XXDP
        BR 8$
        CMP MAHILM,#157776 ;SEE IF MEM LIMIT > OR = 157776
        BHS 4$ ;BR IF YES
        MOV MAHILM,XXDPAD ;HI MEM LIMIT IS END OF XXDP
        ADD #2,XXDPAD ;ADD 2 BYTES
        SUB #6000,XXDPAD ;COMPUTE START OF XXDP
        RTS PC ;RETURN

*****
* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
* THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
* PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
* STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
*
* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
* THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
* STORED IN XDPSAV, XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
* ADDRESS).
*****
SAVXDP: SAVREG ;SAVE R0-R5
        CLR R4 ;SET INDICATOR TO SAVE XXDP
        BR XDP1
GETXDP: SAVREG ;SAVE R0-R5
        CLR R4 ;CLEAR XXDP OVERLAD INDICATOR
        TST XDPSVD ;SEE IF XXDP WAS SAVED
        BEQ XDP2 ;BR IF NOT SAVED
        MOV #1,R4 ;SET INDICATOR TO GET XXDP
XDP1: TST KILLDR ;SEE IF OK TO OVERLAY LOADER
        BNE XDP2 ;BR IF YES, TO EXIT
        MOV #1536,R2 ;GET SET TO MOVE 1536(DEC) WORDS
        MOV XXDPAD,R3 ;GET ORIG ADR OF START OF XXDP
        MOV XDPSAV,R0 ;GET SAVE ADR LO BITS
        MOV XDPSAV+2,R1 ;HI BITS
        TST $KT11 ;SEE IF MEM MGT PRESENT
        BMI XDP3 ;BR IF PRESENT
        TST R4 ;SEE IF WANT TO SAVE OR GET XXDP
        BNE XDP4 ;BR IF WANT TO GET XXDP
        MOV (R3)+,(R0)+ ;SAVE A WORD
        DEC R2 ;SEE IF 1536(DEC) WORDS YET
        BNE 6$ ;BR IF NOT YET
XDP2: RESREG ;RESTORE R0-R5
        RTS PC ;RETURN
XDP4: ADD #6000,R0 ;POINT TO END OF SAVE AREA
        ADD #6000,R3 ;POINT TO END OF XXDP LOADER AREA
        MOV -(R0),-(R3) ;GET A WORD
        DEC R2 ;SEE IF 1536(DEC) WORDS YET

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 147
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0146

7329	032156	001375			BNE	8\$;BR IF NOT YET
7330	032160	000766			BR	XDP2	;GO EXIT
7331							
7332	032162	004737	031160		:COME HERE IF MEM MGT		
7333	032166	006100			XDP3: JSR	PC,INITMM	;INIT MEM MGT REGISTERS
7334	032170	006101			ROL	R0	;GET ADR BITS 13-21 INTO R1 BITS 7-15
7335	032172	006100			ROL	R1	
7336	032174	006101			ROL	R0	
7337	032176	000301			ROL	R1	
7338	032200	006100			SWAB	R1	
7339	032202	106001			ROL	R0	
7340	032204	010137	172354		RORB	R1	
7341	032210	013701	005542		MOV	R1,2#KIPAR6	;SET UP PAR6
7342	032214	042701	160000		MOV	XDP5AV,R1	;GET LO ADRS BITS
7343	032220	052701	140000		BIC	#160000,R1	;FIX UP R1 TO REFERENCE PAR6
7344	032224	005704			BIS	#140000,R1	
7345	032226	001007			16\$: TST	R4	;SEE IF WANT TO SAVE OR GET XXDP
7346	032230	012305			BNE	18\$;BR IF WANT TO GET XXDP
7347	032232	005237	177572		MOV	(R3)+,R5	;SAVE A WORD
7348	032236	010521			INC	2#SR0	;TURN ON MEMORY MANAGEMENT
7349	032240	005337	177572		MOV	R5,(R1)+	
7350	032244	000406			DEC	2#SR0	;TURN OFF MEMORY MANAGEMENT
7351	032246	005237	177572		BR	20\$	
7352	032252	012105			18\$: INC	2#SR0	;TURN ON MEMORY MANAGEMENT
7353	032254	005337	177572		MOV	(R1)+,R5	;GET A WORD
7354	032260	010523			DEC	2#SR0	;TURN OFF MEMORY MANAGEMENT
7355	032262	032701	020000		MOV	R5,(R3)+	
7356	032266	001405			20\$: BIT	#BIT13,R1	;SEE IF OVERFLOW TO PAGE 7
7357	032270	042701	020000		BEQ	22\$;BR IF NOT
7358	032274	062737	000200	172354	BIC	#BIT13,R1	;SET PAGE = 6 AGAIN
7359	032302	005302			ADD	#200,2#KIPAR6	;UPDATE PAR BY 4K
7360	032304	001347			22\$: DEC	R2	;SEE IF 1536 WORDS YET
7361	032306	000713			BNE	16\$;BR IF NOT YET
7362					BR	XDP2	;BR TO RETURN
7363							
7364							
7365							
7366							
7367							
7368							
7369							
7370							
7371							
7372							
7373							
7374							
7375	032310	012737	044200	003046	INITSS: MOV	#ERRHDL,A.ABNL	;SET UP ABNORMAL ERROR RETURN ADDRESS
7376	032316	012737	042732	003044	MOV	#ERRFRE,A.NORM	;SET UP NORMAL RETURN ADDRESS
7377	032324	012702	003036		MOV	RKBAS,R2	;GET ADDRESS OF RK611 REGISTERS
7378	032330	012705	002630		MOV	#PARM0,R5	;GET ADDRESS OF PARAMETER BLOCK
7379	032334	105037	003142		CLRB	NORTRY	;CLEAR "NO-RETRY" FLAG
7380	032340	004737	032410		JSR	PC,CLAPRM	;CLEAR DRIVER INPUT PARAMETERS
7381	032344	112765	000177	000001	MOVB	#SUBCLR,P.CMND(R5)	;SET SUBSYSTEM CLEAR COMMAND
7382	032352	004737	042542		JSR	PC,DRVCAL	;DO SUBSYSTEM CLEAR
7383	032356	113765	005500	000000	MOVB	DRIVE,P.DRVN(R5)	;SET CURRENT DRIVE NO.
7384	032364	113737	005500	002714	MOVB	DRIVE,PARM1	;SET DRIVE NO. IN ALTERNATE P.B.

```

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

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 148
INITSS - INITIALIZE SUBSYSTEM

SEQ 0147

```

7385 032372 113765 003125 000007      MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
7386 032400 153765 003124 000007      BISB    TYPFMT,P.CS1H(R5) ;SET DRV TYP
7387 032406 000207                      RTS     PC          ;SUBROUTINE EXIT
7388
7389
7390
7391 ;*****
7392 ;SBTTL CLRPRM - CLEAR DRIVER INPUT PARAMETERS
7393 ;THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
7394 ;PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
7395 ;CALL - JSR PC,CLRPRM
7396 ;*****
7397
7398 032410 010046      CLRPRM: MOV    R0,-(SP)      ;SAVE R0
7399 032412 010546      MOV    R5,-(SP)      ;SAVE R5
7400 032414 010500      MOV    R5,R0        ;GET PARAMETER BLOCK ADDRESS
7401 032416 062705 000016      ADD    #P.CS1,R5      ;COMPUTE LIMIT ADDRESS
7402 032422 005020      1$: CLR    (R0)+      ;CLEAR A WORD IN PARAMETER BLOCK
7403 032424 020005      CMP    R0,R5        ;SEE IF DONE YET
7404 032426 001375      BNE    1$          ;BR IF NOT DONE YET
7405 032430 012605      MOV    (SP)+,R5      ;RESTORE R5
7406 032432 012600      MOV    (SP)+,R0      ;RESTORE R0
7407 032434 000207      RTS     PC          ;RETURN
7408
7409
7410
7411 ;*****
7412 ;SBTTL SCNDRV - SCAN DRIVE FOR STATUS
7413 ;THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
7414 ;THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
7415 ;AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
7416 ;OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
7417 ;MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
7418 ;A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7419 ;AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
7420 ;MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
7421 ;ALSO, IF RUNNING FROM ADRS 200 AND RK06 IS LOAD MEDIUM,
7422 ;DRIVE 0 WILL BE REJECTED FOR USE.
7423 ;CALL - JSR PC,SCNDRV
7424 ;
7425 ;      (ERROR RETURN ADDRESS)
7426 ;
7427 ;*****
7428 032436 105037 003124      SCNDRV: CLRB   TYPFMT      ;ASSUME RK06 TO START
7429 032442 004737 032310      JSR    PC,INITSS      ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
7430 032446 113765 005500      MOVB   DRIVE,P.DRVN(R5) ;GET DRIVE NUMBER
7431 032454 042712 000100      BIC    #IE,(R2)      ;DISABLE RK06 INTERRUPT
7432 032460 113762 005500      MOVB   DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
7433 032466 105737 003124      TSTB   TYPFMT      ;SEE IF RK07
7434 032472 001003      BNE    5$          ;BR IF YES
7435 032474 012712 000001      MOV    #GO,(R2)      ;ELSE SET GO BIT IN CS1 FOR RK06
7436 032500 000402      BR     7$
7437 032502 012712 002001      5$: MOV    #<CDT!GO>,(R2)
7438 032506 005037 005534      7$: CLR    SCRACH      ;CLEAR STALL COUNTER
7439 032512 005237 005534      14$: INC   SCRACH      ;INCREMENT STALL COUNTER
7440 032516 001375      BNE    14$          ;STALL FOR SEVERAL MILLI-SEC

```

```

7441 032520 032762 001000 000010 BIT #MDS,RKCS2(R2) ;SEE IF MDS ERROR
7442 032526 001417 BEQ 18$ ;BR IF NOT
7443 032530 112765 000101 000001 MOVB #SELD RV,P.CMND(R5) ;SET COMMAND FOR ERROR REPORT
7444 032536 011265 000016 MOV (R2),P.CSI(R5) ;GET RKCS1
7445 032542 004737 052226 JSR PC,I.CST1 ;GET OTHER RK611 REGS FOR REPORT
7446 032546 004737 043720 JSR PC,REPSUP ;SET UP ERROR REPORT
7447 032552 1J4052 ERROR 52 ;REPORT MDS ERROR
7448 032554 052737 000200 005474 BIS #ABORT,RECODE ;SET ABORT FLAG
7449 032562 000137 045774 JMP ALLTRM ;GO ABORT TESTING
7450 032566 032762 000040 000014 18$: BIT #DTYPE,RKER(R2) ;SEE IF DRV TYP ERR
7451 032574 001403 BEQ 20$ ;BR IF NO=RK06
7452 032576 152737 000004 003124 BISB #B.CDT,TYPFMT ;ELSE SET FOR RK07
7453 032604 004737 032310 20$: JSR PC,INITSS ;INIT THE SUBSYSTEM
7454 032610 112765 000141 000001 MOVB #R0STAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
7455 032616 012737 030360 003046 MOV #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
7456 032624 012737 000377 005530 MOV #377,NEWON ;INIT. ON-LINE INDICATOR
7457 032632 004737 042542 JSR PC,DRVCAL ;READ ALL DRIVE STATUS
7458 ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
7459 032636 012737 044200 003046 MOV #ERRHDL,A.ABNL ;RESTORE ABNL RETURN ADDRESS
7460 032644 022737 000377 005530 CMP #377,NEWON ;SEE IF NED INDICATION ON THIS DRIVE
7461 032652 001430 BEQ 4$ ;BR IF NED NOT SET
7462 032654 005737 003150 TST AUTOFG ;FORM AUTO MODE ?
7463 032660 001017 BNE 2$
7464 032662 113737 005500 007530 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
7465 032670 152737 000060 007530 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7466 032676 104401 007522 TYPE ,BADDRV ;TYPE "DRIVE X"
7467 032702 104401 007533 TYPE ,NXDRIV ;TYPE "NON-EXISTENT"
7468 032706 132737 000200 001341 BITB #BIT7,SENVN ;SEE IF APT
7469 032714 001401 BEQ 2$ ;BR IF NO
7470 032716 104123 ERROR 123 ;NED UNDER APT SIZING
7471 ;SERVICE ERRORS HERE
7472 032720 042762 000100 000000 2$: BIC #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT
7473 032726 017616 000000 MOV 2(SP),(SP) ;SET UP ERROR RETURN ADDRESS
7474 032732 000207 RTS ;ERROR RETURN
7475 ;SEE IF DRIVE IS READY
7476 032734 032765 000200 000040 4$: BIT #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
7477 032742 001016 BNE 6$ ;BR IF DRIVE IS READY
7478 032744 005737 003150 TST AUTOFG
7479 032750 001363 BNE 2$ ;SKIP IF FROM AUTO MODE
7480 032752 113737 005500 007530 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
7481 032760 152737 000060 007530 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7482 032766 104401 007522 TYPE ,BADDRV ;TYPE "DRIVE X"
7483 032772 104401 007547 TYPE ,NTREDY ;TYPE "NOT READY"
7484 032776 000750 BR 2$ ;TAKE ERROR EXIT
7485 ;SEE IF DRIVE IS WRITE ENABLED
7486 033000 032765 004000 000040 6$: BIT #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
7487 033006 001416 BEQ 8$ ;BR IF WRITE LOCK NOT SET
7488 033010 005737 003150 TST AUTOFG
7489 033014 001013 BNE 8$ ;FROM THE AUTO MODE
7490 033016 113737 005500 007530 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
7491 033024 152737 000060 007530 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7492 033032 104401 007522 TYPE ,BADDRV ;TYPE "DRIVE X"
7493 033036 104401 007561 TYPE ,WRTLOK ;TYPE "WRITE-LOCKED"
7494 033042 000726 BR 2$ ;TAKE ERROR EXIT
7495 ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
7496 033044 112765 000103 000001 8$: MOVB #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND

```

```

7497 033052 004737 042542      JSR    PC,DRVCAL      ;SET VOLUME VALID
7498 033056 112765 000113 000001  MOVB   #RECAL,P.CMND(R5) ;SET RECAL COMMAND
7499 033064 004737 042542      JSR    PC,DRVCAL      ;RECALIBRATE THE DRIVE
7500 033070 112765 000121 000001  MOVB   #RDATA,P.CMND(R5) ;SET READ COMMAND
7501 033076 105737 003124      TSTB   TYPFMT          ;SEE IF RK07
7502 033102 001004                BNE     25$              ;BR IF YES
7503 033104 012765 000632 000002  MOV     #632,P.CYLN(R5)
7504 033112 000403                BR      27$
7505 033114 012765 001456 000002 25$:  MOV     #1456,P.CYLN(R5)
7506 033122 112765 000002 000005 27$:  MOVB   #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
7507 033130 012765 065660 000010  MOV     #RBUF,P.BALO(R5) ;BUS ADDRESS
7508 033136 012765 177774 000012  MOV     #-4,P.WC(R5)      ;READ 4 WORDS
7509 033144 142765 000020 000007  BICB   #B.CFMT,P.CS1H(R5) ;SET 22 SECTOR FORMAT
7510 033152 153765 003124 000007  BISB   TYPFMT,P.CS1H(R5)
7511 033160 005737 003150      TST     AUTOFG          ;SEE IF FROM AUTO MODE
7512 033164 001027                BNE     12$              ;EXIT IF SO
7513 033166 004737 042542      JSR    PC,DRVCAL      ;READ 4 WORDS OF BAD SECTOR FILE
7514 033172 032737 100000 005474  BIT     #ANYDER,RECODE ;SEE IF DATA ERROR
7515 033200 001402                BEQ     10$              ;BR IF OK
7516 033202 104401 013202                TYPE   BAD632      ;TYPE READ ERROR MESSAGE
7517 033206 022737 177777 065666 10$:  CMP     #177777,RBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
7518 033214 001013                BNE     12$              ;BR IF NOT ALL 1'S
7519 033216 113737 005500 007530  MOVB   DRIVE,BADDRV+6 ;GET DRIVE NO.
7520 033224 152737 000060 007530  BISB   #'0,BADDRV+6 ;CONVERT TO ASCII
7521 033232 104401 007522                TYPE   ,BADDRV      ;TYPE "DRIVE X"
7522 033236 104401 007574                TYPE   ,ALNPAK      ;TYPE "LOADED WITH ALIGN PACK"
7523 033242 000626                BR      2$
7524                                ;ERROR FREE RETURN
7525 033244 062716 000002 12$:  ADD     #2,(SP)      ;FIX UP RETURN PC
7526 033250 000207                RTS     PC              ;RETURN
7527
7528
7529
7530 ;*****
7531 ;SBTTL CHKDRV - CHECK STATUS OF DRIVE
7532 ;*THIS SUBROUTINE CHECKS THE DESIRED DRIVE TO DETERMINE
7533 ;*IF THE DRIVE IS ON-LINE,READY,WRITE-PROTECTED. IF NOT, THE
7534 ;*APPROPRIATE ERROR MESSAGE IS TYPED,AND THE CPU HALTS
7535 ;*WHILE MANUAL INTERVENTION TAKES PLACE TO CORRECT THE
7536 ;*PROBLEM. THE OPERATOR THEN DEPRESSES 'CONT' TO PROCEED.
7537 ;*AND A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7538 ;*AFTER THE CALL. IF THERE ARE NO PROBLEMS WITH THIS
7539 ;*DRIVE, A NORMAL RETURN IS MADE.
7540 ;*R5 MUST CONTAIN THE ADDRESS OF THE PARAMETER BLOCK,
7541 ;*AND R2 MUST CONTAIN THE RK06 REGISTER BASE ADDRESS.
7542 ;*
7543 ;* CALL:
7544 ;*      JSR    PC,CHKDRV
7545 ;*      ERROR ADDRESS
7546 ;*****
7547
7548 033252 113765 005500 000000  CHKDRV: MOVB   DRIVE,P.DRVN(R5) ;SET DESIRED DRIVE NUMBER
7549 033260 112765 000141 000001  MOVB   #RSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
7550 033266 012737 030360 003046  MOV     #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
7551 033274 012737 000377 005530  MOV     #377,NEWON      ;INITIALIZE ON-LINE INDICATOR
7552 033302 004737 042542      JSR    PC,DRVCAL      ;READ ALL DRIVE STATUS

```

```

7553 ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
7554 033306 012737 044200 003046 MOV #ERRHDL,A.ABNL ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
7555 033314 022737 000377 005530 CMP #377,NEWON ;SEE IF NED INDICATION ON THIS DRIVE
7556 033322 001411 BEQ 1$ ;BR IF NED NOT SET
7557 033324 113701 005500 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7558 033330 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7559 033334 110137 011754 MOVB R1,DRVNRD ;PUT DRIVE NO. INTO MSG BUFFER
7560 033340 104401 011744 TYPE NONEXD ;TYPE NON-EXISTENT DRIVE MESSAGE
7561 033344 000414 BR 3$ ;SERVICE THE ERROR
7562 ;SEE IF DESIRED DRIVE IS READY
7563 033346 032765 000200 000040 1$: BIT #S.DRY,P.ADD(R5) ;TEST FOR DRIVE READY
7564 033354 001024 BNE 4$ ;BR IF DRIVE READY
7565 033356 113701 005500 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7566 033362 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7567 033366 110137 011622 MOVB R1,DRNRDY ;PUT DRIVE NUMBER IN MESSAGE BUFFER
7568 033372 104401 011612 TYPE NOTRDY ;TYPE DRIVE NOT READY MESSAGE
7569 ;SERVICE ALL ERRORS HERE
7570 033376 042762 000100 000000 3$: BIC #IE,RKCS1(R2) ;CLEAR RK06 INTERRUPT ENABLE BIT
7571 033404 000000 HALT ;HALT IS NECESSARY DURING MANUAL
7572 ; INTERVENTIONS, TO CHANGE PACK,
7573 ; WRITE LOCK, START DRIVE, ETC.
7574 ; PRESS 'CONT' TO PROCEED!
7575 033406 112765 000177 000001 MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
7576 033414 004737 042542 JSR PC,DRVCL ;CLEAR THE SUBSYSTEM
7577 033420 017616 000000 MOV 2(SP),(SP) ;SET UP ERROR RETURN ADDRESS
7578 033424 000207 RTS PC ;ERROR RETURN
7579 ;SEE IF DRIVE IS WRITE-LOCKED
7580 033426 032765 004000 000040 4$: BIT #S.WRL,P.ADD(R5) ;SEE IF WRITE LOCK SET
7581 033434 001011 BNE 5$ ;BR IF WRITE-PROTECTED
7582 033436 113701 005500 MOVB DRIVE,R1 ;GET DRIVE NUMBER
7583 033442 152701 000060 BISB #'0,R1 ;CONVERT TO ASCII
7584 033446 110137 012051 MOVB R1,NOTDLK ;PUT DRIVE NUMBER IN MSG BUFFER
7585 033452 104401 012041 TYPE NOTLOK ;TYPE "DRIVE NOT WRITE-LOCKED"
7586 033456 000747 BR 3$ ;SERVICE THE ERROR
7587 ;RECALIBRATE DESIRED DRIVE, SET VOLUME VALID
7588 033460 112765 000113 000001 5$: MOVB #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
7589 033466 004737 042542 JSR PC,DRVCL ;RECALIBRATE DRIVE
7590 033472 112765 000103 000001 MOVB #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7591 033500 004737 042542 JSR PC,DRVCL ;DO PACK ACK. (SETS VOLUME VALID)
7592 ;RETURN HERE
7593 033504 062716 000002 7$: ADD #2,(SP) ;FIX UP RETURN ADDRESS ON STACK
7594 033510 000207 RTS PC ;ERROR-FREE RETURN
7595
7596
7597
7598 ;*****
7599 ;* ALNSEK - SEEK IN SINGLE INCREMENTS FROM CYL 0 TO ALIGNMENT CYL (365 OCT).
7600 ;*****
7601 033512 112765 000117 000001 ALNSEK: MOVB #SEEK,P.CMND(R5) ;SET SEEK COMMAND
7602 033520 005065 000002 CLR P.CYLN(R5) ;INIT CYL TO 0
7603 033524 004737 042542 JSR PC,DRVCL ;SEEK TO THIS CYL
7604 033530 005265 000002 INC P.CYLN(R5) ;INCR CYL NO.
7605 033534 105737 003124 TSTB TYPFMT ;CURRENT DRIVE IS AN RK07 ?
7606 033540 001005 BNE 3$ ;BRANCH IF IT IS
7607 033542 022765 000366 000002 CMP #ALNCYL+1,P.CYLN(R5) ;SEE IF 365 REACHED YET
7608 033550 001365 BNE 2$ ;BR IF NOT YET

```



```

7609 033552 000207      RTS      PC      ;RETURN
7610 033554 022765 000761 000002 3$:  CMP      #761,P.CYLN(R5) ;REACH THE CYLINDER 760 FOR RK07
7611 033562 001360      BNE      2$      ;CONTINUE
7612 033564 000207      RTS      PC      ;EXIT
7613
7614
7615
7616
7617 ;*****
7618 ;* WAIT4R - THIS SUBROUTINE UNLOADS HEADS ON THE CURRENT DRIVE, TYPES
7619 ;* "TYPE <R> WHEN READY", AND THEN LOADS THE HEADS WHEN <R> IS TYPED.
7620 ;*****
7620 033566 112765 000107 000001 WAIT4R: MOVB    #UNLOAD,P.CMND(R5) ;SET UNLOAD COMMAND
7621 033574 004737 042542      JSR      PC,DRVCAL ;UNLOAD HEADS ON THIS DRIVE
7622 033600 112765 000141 000001      MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
7623 033606 004737 042542 1$:  JSR      PC,DRVCAL ;READ STATUS OF DRIVE
7624 033612 032765 000040 000044      BIT      #S.HDHM,P.A01(R5) ;SEE IF HEADS UNLOADED YET
7625 033620 001772      BEQ      1$      ;BR IF NOT YET
7626 033622 104401 012651      TYPE     RWNRDY ;TYPE "TYPE <R> WHEN READY"
7627 033626 004737 030576      JSR      PC,PREPKB ;PREPARE FOR KBD INPUT
7628 033632 005737 005522 4$:  TST      INTCHR ;SEE IF ANY INPUT YET
7629 033636 001775      BEQ      4$      ;BR IF NOT YET
7630 033640 022737 000122 005522      CMP      #'R,INTCHR ;SEE IF <R> TYPED
7631 033646 001403      BEQ      6$      ;BR IF <R> TYPED
7632 033650 004737 030616      JSR      PC,ECOBAD ;ECHO BAD INPUT
7633 033654 000762      BR       2$      ;GO ASK AGAIN
7634 033656 112765 000111 000001 6$:  MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
7635 033664 004737 042542      JSR      PC,DRVCAL ;START SPINDLE AND LOAD HEADS
7636 033670 000207      RTS      PC      ;RETURN
7637
7638
7639
7640 ;*****
7641 ;*SETUP - SET UP FOR LOOP ON ERROR
7642 ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
7643 ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
7644 ;*****
7645 033672 011637 001076      SETUP:  MOV      (SP),2#STACK-2 ;MOVE RETURN PC ON STACK
7646 033676 012706 001076      MOV      #STACK-2,SP ;RE-INIT THE STACK POINTER
7647 033702 005037 005474      CLR      RECODE ;CLEAR ERROR RECOVERY FLAGS
7648 033706 105037 003126      CLRB    ERRCNT ;CLEAR RETRY ERROR COUNT
7649 033712 012705 002630      MOV      #PARAM,R5 ;SET PARAM BLK ADRS
7650 033716 112765 000177 000001      MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
7651 033724 004737 042542      JSR      PC,DRVCAL ;CLEAR THE SUBSYSTEM
7652 033730 012737 000000 177776      MOV      #PRO,2#PS ;RE-ESTABLISH PRIORITY 0
7653 033736 000207      RTS      PC      ;RETURN
7654
7655
7656
7657 ;*****
7658 ;*BTTL CHKITR - CHECK CURRENT TEST ITERATION NUMBER
7659 ;*THIS SUBROUTINE LOADS THE ITERATION NUMBER FOR THE CURRENT
7660 ;*TEST INTO $TIMES, AND CHECKS IT TO DETERMINE IF THIS TEST
7661 ;*SHOULD BE RUN. IF ITERATION NO. = 0, THE SUBROUTINE RETURNS
7662 ;*TO THE STORED PC (AND THE TEST WILL BE SKIPPED). IF IT IS
7663 ;*NOT 0, THE RETURN IS MADE TO PC+2, AND THE TEST WILL
7664 ;*BE RUN. HOWEVER, ON THE FIRST PASS (QUICK VERIFY) OF A RUN

```

```

7665 ;*WHICH STARTED AT ADRS 200, ALL TESTS WILL BE RUN ONCE.
7666
7667 033740 010146 CHKITR: MOV R1,-(SP) ;SAVE R1
7668 033742 112737 000144 001115 MOVB #100,$SERMAX ;SET MAX ERROR CNT TO 100 FOR $SCOPE
7669 033750 105737 003135 TSTB DULACS ;SEE IF DUAL-ACCESS FLAG SET
7670 033754 001006 BNE 3$ ;BR IF YES
7671 033756 105737 003116 TSTB MDFLAG ;SEE IF 200 START
7672 033762 001007 BNE 4$ ;BR IF NOT
7673 033764 005737 001326 TST $PASS ;SEE IF FIRST PASS
7674 033770 001004 BNE 4$ ;BR IF NOT
7675 033772 012737 000001 001304 3$: MOV #1,$TIMES ;SET UP FOR 1 ITERATION
7676 034000 000410 BR 1$ ;GO RUN DUAL-ACCESS TEST
7677 034002 113701 001102 4$: MOVB $TSTNM,R1 ;GET CURRENT TEST NUMBER
7678 034006 005301 DEC R1 ;DECREMENT BY 1
7679 034010 006301 ASL R1 ;DOUBLE IT, TO GET TEST LIST INDEX
7680 034012 016137 005630 001304 MOV TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
7681 034020 001403 BEQ 2$ ;BR IF TEST SHOULD BE SKIPPED
7682 034022 062766 000004 000002 1$: ADD #4,2(SP) ;ADJUST RETURN PC TO RUN THIS TEST
7683 034030 012601 2$: MOV (SP)+,R1 ;RESTORE R1
7684 034032 000207 RTS PC ;RETURN
7685
7686
7687
7688 ;*****
7689 ;SBTTL RDCHRS - READ A STRING OF KBD INPUT CHARS
7690 ;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
7691 ;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
7692 ;*THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
7693 ;*RUB-OUT AND (↑) FEATURES ARE PROVIDED.
7694 ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
7695 ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
7696 ;*TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
7697 ;*TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
7698 ;*IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
7699 ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
7700 ;*IS TAKEN.
7701 ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
7702 ;*IN R0.
7703 ;*
7704 ; CALL - JSR PC,RDCHRS
7705 ; <CONTROL-C RETURN ADDRESS>
7706 ; <CONTROL-Z RETURN ADDRESS>
7707 ; <CONTROL-U OR ERROR RETURN ADDRESS>
7708 ;*
7709 ;* RETURN
7710 ;*****
7711 034034 RDCHRS:
7712 034034 010146 MOV R1,-(SP) ;SAVE R1
7713 034036 010246 MOV R2,-(SP) ;SAVE R2
7714 034040 005000 CLR R0 ;INITIALIZE CHARACTER COUNT
7715 034042 005001 CLR R1 ;INITIALIZE RUB-OUT INDICATOR
7716 ;READ A CHARACTER
7717 034044 104406 2$: RDCHR ;READ A CHARACTER
7718 034046 112602 MOVB (SP)+,R2 ;GET CHARACTER INTO R2
7719 ;CHECK FOR (↑C)
7720 034050 122702 000003 CMPB #003,R2 ;SEE IF (↑C) TYPED

```

7721	034054	001006			BNE	4\$;BR IF NOT (↑C)
7722	034056	104401	013246		TYPE	CNTRL C		;ECHO (↑C)
7723	034062	017666	000004	000004	MOV	24(SP),4(SP)		;PUT RETURN ADDRESS ON STACK
7724	034070	000523			BR	24\$;BR TO TAKE EXIT
7725					;CHECK FOR (↑Z)			
7726	034072	122702	000032		4\$: CMPB	#032,R2		;SEE IF (↑Z) TYPED
7727	034076	001006			BNE	6\$;BR IF NOT (↑Z)
7728	034100	104401	013253		TYPE	CNTRL Z		;ECHO (↑Z)
7729	034104	062766	000002	000004	ADD	#2,4(SP)		;MAKE OLD PC POINT TO NEXT RETURN ADR.
7730	034112	000763			BR	3\$;BR TO TAKE (↑Z) EXIT
7731					;CHECK FOR (↑U)			
7732	034114	122702	000025		6\$: CMPB	#025,R2		;SEE IF (↑U) TYPED
7733	034120	001006			BNE	8\$;BR IF NOT (↑U)
7734	034122	104401	013265		TYPE	CNTRL U		;ECHO (↑U)
7735	034126	062766	000004	000004	ADD	#4,4(SP)		;MAKE OLD PC POINT TO NEXT RETURN ADDR.
7736	034134	000752			BR	3\$;BR TO TAKE (↑U) EXIT
7737					;CHECK FOR (↑G)			
7738	034136	122702	000007		8\$: CMPB	#007,R2		;SEE IF (↑G) TYPED
7739	034142	001005			BNE	9\$;BR IF NOT (↑G)
7740	034144	104401	013272		TYPE	CNTRL G		;ECHO (↑G)
7741	034150	004737	031126		JSR	PC,GTSWRG		;OPEN SOFTWARE SWITCH REG. FOR CHANGE
7742	034154	000764			BR	7\$;TAKE (↑U) RETURN
7743					;CHECK FOR RUB-OUT (DELETE)			
7744	034156	122702	000177		9\$: CMPB	#177,R2		;SEE IF RUB-OUT (DEL) TYPED
7745	034162	001020			BNE	14\$;BR IF NOT RUB-OUT
7746	034164	005700			TST	R0		;CHECK THE CHARACTER COUNT
7747	034166	001726			BEQ	2\$;BR IF COUNT = 0
7748	034170	005701			TST	R1		;CHECK THE RUB-OUT INDICATOR
7749	034172	001003			BNE	11\$;BR IF WE HAD A PREVIOUS RUB-OUT
7750	034174	005201			INC	R1		;SET RUB-OUT INDICATOR
7751	034176	104401	013303		TYPE	BKSLSH		;TYPE A BACK-SLASH (\)
7752	034202	005037	005534		11\$: CLR	SCRACH		;USE SCRATCH WORD FOR TEMP. BUFFER
7753	034206	005300			DEC	R0		;DECREMENT COUNT
7754	034210	116037	005264	005534	MOVB	BUFFO(R0),SCRACH		;GET LAST CHAR. INTO BUFFER
7755	034216	104401	005534		TYPE	SCRACH		;ECHO CHARACTER TO BE DELETED
7756	034222	000710			BR	2\$;GO READ ANOTHER CHARACTER
7757	034224	005701			14\$: TST	R1		;CHECK THE RUB-OUT INDICATOR
7758	034226	001403			BEQ	15\$;BR IF INDICATOR IS NOT SET
7759	034230	104401	013303		TYPE	BKSLSH		;TYPE A BACK-SLASH
7760	034234	005001			CLR	R1		;CLEAR THE RUB-OUT INDICATOR
7761					;CHECK FOR CARRIAGE RETURN			
7762	034236	122702	000015		16\$: CMPB	#015,R2		;SEE IF <CR> TYPED
7763	034242	001426			BEQ	19\$;BR IF <CR>
7764					;HANDLE POSSIBLE DIGIT			
7765	034244	005037	005534		CLR	SCRACH		;USE SCRATCH WORD FOR TEMP. BUFFER
7766	034250	110237	005534		MOVB	R2,SCRACH		;GET THIS CHARACTER INTO BUFFER
7767	034254	104401	005534		TYPE	SCRACH		;ECHO THE CHARACTER TYPED
7768	034260	110260	005264		MOVB	R2,BUFFO(R0)		;PUT CHARACTER INTO BUFFER
7769	034264	005200			INC	R0		;INCREMENT CHARACTER COUNTER
7770	034266	022700	000120		CMP	#80.,R0		;SEE IF TOO MANY CHARACTERS TYPED
7771	034272	001264			BNE	2\$;BR IF NOT TOO MANY
7772	034274	104401	001315		TYPE	\$CRLF		;TYPE <CR> AND <LF>
7773	034300	112760	000000	005264	MOVB	#0,BUFFO(R0)		;PUT TERMINATING NULL INTO BUFFER
7774	034306	104401	005264		TYPE	\$BUFFO		;ECHO INPUT STRING
7775	034312	104401	001314		TYPE	\$QUES		;TYPE <?> <CR> <LF>
7776	034316	000703			BR	7\$;TAKE ERROR EXIT FROM RDCHRS

7777 034320 104401 001315
7778 034324 112760 000000 005264
7779 034332 062766 000006 000004
7780 034340 012602
7781 034342 012601
7782 034344 000207
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796 034346 104401 013314
7797 034352 010146
7798 034354 006216
7799 034356 005216
7800 034360 104403
7801 034362 002
7802 034363 000
7803 034364 104401 013307
7804 034370 016146 005630
7805 034374 104403
7806 034376 006
7807 034377 000
7808 034400 000207
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821
7822 034402 016137 006010 013316
7823 034410 104401 013316
7824 034414 016137 005660 005470
7825 034422 005037 005472
7826 034426 022761 040515 006010
7827 034434 001003
7828 034436 016137 005662 005472
7829 034444 012746 005470
7830 034450 004737 055436
7831 034454 004737 055752
7832 034460 000207

19\$: TYPE \$CRLF ;TYPE <CR>,<LF>
MOVB #0,BUFF0(R0) ;PUT TERMINATING NULL INTO BUFFER
ADD #6,4(SP) ;FIX UP RETURN ADDRESS
24\$: MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;SUBROUTINE EXIT

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

TYPTST: TYPE SPACE1 ;TYPE A SPACE
MOV R1,-(SP) ;PUT INDEX ONTO STACK
ASR (SP) ;DIVIDE BY 2
INC (SP) ;INCREMENT TO GET TEST NO.
TYPOS ;TYPE TEST NO. IN OCTAL
.BYTE 2 ;TYPE 2 DIGITS
.BYTE 0 ;SUPPRESS LEADING ZEROS
TYPE SPACE6 ;TYPE 6 SPACES
MOV #STLST(R1),-(SP) ;PUT CURRENT ITERATION NO. ONTO STACK
TYPOS ;TYPE ITERATION NO. IN OCTAL
.BYTE 6 ;TYPE 6 DIGITS
.BYTE 0 ;SUPPRESS LEADING ZEROS
RTS PC ;RETURN

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

TYPPRM: MOV PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
TYPE PRMBUF ;TYPE "XX="
MOV PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
CLR HIGOCT ;CLEAR HIGH BINARY BITS
CMP #MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
BNE 1\$;BR IF NOT (MA)
MOV PRMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
1\$: MOV #LOWOCT,-(SP) ;PUT ADDRESS OF BINARY VALUE ON STACK
JSR PC,#\$D820 ;CONVERT BINARY TO OCTAL ASCII
JSR PC,#\$SUPRS ;TYPE YYYYYYYY, SUPPRESSING LEADING 0'S
RTS PC ;RETURN

7833
7834
7835
7836
7837
7838
7839
7840
7841
7842
7843
7844
7845
7846
7847 034462
7848 034462 104401 013322
7849 034466 010146
7850 034470 006216
7851 034472 104403
7852 034474 002
7853 034475 001
7854 034476 104401 013330
7855 034502 016146 006776
7856 034506 104403
7857 034510 006
7858 034511 001
7859 034512 000207
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877 034514
7878 034514 010046
7879 034516 010146
7880 034520 010246
7881
7882 034522 022761 052120 006010
7883 034530 001127
7884
7885 034532 032737 100000 005700
7886 034540 001523
7887
7888 034542 104401 010730

```

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

```

```

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

```

```

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

```

```

MODP15:
        MOV       R0,-(SP)     ;SAVE R0
        MOV       R1,-(SP)     ;SAVE R1
        MOV       R2,-(SP)     ;SAVE R2
;SEE IF PARAMETER IS PT
        CMP       #PT,PRMNM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
        BNE       22$          ;BR IF NOT (PT)
;SEE IF PATTERN 15 IS SPECIFIED
        BIT       #BIT15,PT    ;SEE IF PATTERN 15 SPECIFIED
        BEQ       22$          ;BR IF NOT SPECIFIED
;SEE IF PATTERN 15 SHOULD BE MODIFIED
        TYPE      ,MODY15      ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
4$:

```

7889	034546	004737	034034		JSR	PC, RDCHRS	; READ RESPONSE
7890	034552	035020			24\$; (↑C) RETURN ADDRESS
7891	034554	035030			26\$; (↑Z) RETURN ADDRESS
7892	034556	034614			8\$; (↑U) OR ERROR RETURN ADDRESS
7893	034560	005700			TST	RO	; SEE IF NULL INPUT
7894	034562	001512			BEQ	22\$; BR IF MODIFICATION NOT REQUESTED
7895	034564	022737	000115	005264	CMP	#'M, BUFFO	; SEE IF (M) TYPED
7896	034572	001405			BEQ	6\$; BR IF MODIFICATION REQUESTED
7897	034574	104401	005264		TYPE	, BUFFO	; ECHO BAD INPUT
7898	034600	104401	001314		TYPE	, \$QUES	
7899	034604	000756			BR	4\$; GO ASK AGAIN
7900					; MODIFY PATTERN	15	
7901	034606	104401	010706		6\$: TYPE	A SELP15	; TYPE "MODIFY PATTERN 15"
7902	034612	005001			CLR	R1	; INITIALIZE WORD INDEX
7903	034614	004737	034462		8\$: JSR	PC, TYPPAT	; TYPE CURRENT WORD AND VALUE
7904	034620	104401	013314		TYPE	, SPACE1	; TYPE A SPACE
7905	034624	104401	013334		TYPE	, PROMPT	; TYPE ASTERISK AND SPACE
7906					; READ AND CHECK	INPUT, IF ANY	
7907	034630	004737	034034		JSR	PC, RDCHRS	; READ NEW DATA PATTERN WORD
7908	034634	035020			24\$; (↑C) RETURN ADDRESS FOR RDCHRS
7909	034636	035030			26\$; (↑Z) RETURN ADDRESS FOR RDCHRS
7910	034640	034614			8\$; (↑U) OR ERROR RETURN ADDR. FOR RDCHRS
7911	034642	005700			TST	RO	; SEE IF ANY INPUT
7912	034644	001006			BNE	12\$; BR IF ANY INPUT
7913	034646	062701	000002		10\$: ADD	#2, R1	; INCREMENT WORD INDEX
7914	034652	022701	000040		CMP	#32, R1	; SEE IF ALL DONE
7915	034656	001454			BEQ	22\$; BR IF DONE, TO RETURN
7916	034660	000756			BR	8\$; CONTINUE WITH NEXT WORD
7917	034662	022737	000041	005264	12\$: CMP	#'!, BUFFO	; SEE IF (!) TYPED
7918	034670	001431			BEQ	16\$; BR TO PROPAGATE CURRENT VALUE
7919	034672	005002			CLR	R2	; INIT. (!) INDICATOR
7920	034674	122760	000041	005263	CMPB	#'!, BUFFO-1(R0)	; SEE IF LAST CHAR IN BUF IS (!)
7921	034702	001004			BNE	14\$; BR IF NOT (!)
7922	034704	105060	005263		CLRB	BUFFO-1(R0)	; INSERT TERMINATOR BYTE
7923	034710	005300			DEC	RO	; DECREMENT CHAR COUNT
7924	034712	005202			INC	R2	; SET (!) INDICATOR
7925	034714	022700	000006		14\$: CMP	#6, RO	; SEE HOW MANY CHARS NOW
7926	034720	002426			BLT	20\$; BR IF TOO MANY
7927	034722	012746	005264		MOV	#BUFFO, -(SP)	; GET BUF. ADDR. ON STACK FOR OCTBIN
7928	034726	004737	054500		JSR	PC, OCTBIN	; CHECK DIGITS AND CONVERT TO BINARY
7929	034732	034776			20\$; ERROR RETURN ADDRESS FOR OCTBIN
7930	034734	012600			MOV	(SP)+, RO	; GET BINARY VALUE
7931	034736	005737	054632		TST	\$HI OCT	; SEE IF VALUE EXCEEDS 16 BITS
7932	034742	001015			BNE	20\$; BR TO ECHO BAD INPUT
7933	034744	010061	006776		MOV	RO, PAT15(R1)	; PUT NEW WORD VALUE INTO TABLE
7934	034750	005702			TST	R2	; SEE IF (!) WAS TYPED
7935	034752	001735			BEQ	10\$; BR IF (!) WAS NOT TYPED
7936					; PROPAGATE CURRENT WORD TO END OF	PATTERN 15 TABLE	
7937	034754	022701	000036		16\$: CMP	#30, R1	; SEE IF ALL DONE YET
7938	034760	001413			BEQ	22\$; BR IF DONE
7939	034762	016161	006776	007000	MOV	PAT15(R1), PAT15+2(R1)	; PROPAGATE TO NEXT WORD
7940	034770	062701	000002		ADD	#2, R1	; INCREMENT WORD INDEX
7941	034774	000767			BR	16\$; LOOP UNTIL DONE
7942					; ECHO BAD INPUT		
7943	034776	104401	005264		20\$: TYPE	, BUFFO	; ECHO BAD INPUT
7944	035002	104401	001314		TYPE	, \$QUES	; TYPE <?> AND <CR>, <LF>

```

7945 035006 000702          BR      B$          ;BR TO ASK AGAIN
7946          NORMAL RETURN
7947 035010 012602          22$: MOV      (SP)+,R2      ;RESTORE R2
7948 035012 012601          MOV      (SP)+,R1      ;RESTORE R1
7949 035014 012600          MOV      (SP)+,R0      ;RESTORE R0
7950 035016 000207          RTS      PC          ;RETURN
7951          ;(↑C) RETURN
7952 035020 012766 014434 000006 24$: MOV      #DRVST,6(SP) ;PC FOR (↑C) RETURN
7953 035026 000770          BR      22$          ;BR TO RETURN
7954          ;(↑Z) RETURN
7955 035030 012766 016020 000006 26$: MOV      #ASKMDE,6(SP) ;PC FOR (↑Z) RETURN
7956 035036 000764          BR      22$          ;BR TO RETURN
7957
7958
7959
7960          ;*****
7961          ;SBTTL CHKPRM - CHECK VALUE OF INPUT PARAMETER
7962          ;THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
7963          ;HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
7964          ;LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
7965          ;INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
7966          ;TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
7967          ;IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
7968          ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
7969          ;THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
7970          ;VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
7971          ;CALL - JSR      PC,CHKPRM
7972          ;      <ERROR RETURN ADDRESS>
7973          ;      RETURN
7974          ;*****
7975
7976 035040 104407          CHKPRM: SAVREG          ;SAVE R0-R5
7977 035042 010102          MOV      R1,R2          ;GET COPY OF INDEX
7978 035044 006302          ASL      R2          ;DOUBLE IT
7979 035046 022761 040515 006010 CMP      #MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
7980 035054 001422          BEQ      20$          ;BR IF (MA)
7981 035056 005737 005472 TST      HIGOCT          ;SEE IF HIGH BITS = 0
7982 035062 001403          BEQ      18$          ;BR IF ZERO
7983 035064 017616 000000 16$: MOV      2(SP), (SP) ;GET ERROR RETURN PC
7984 035070 000475          BR      30$          ;GO TO RETURN
7985          ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
7986 035072 023762 005470 005734 18$: CMP      LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
7987 035100 103771          BLO      16$          ;BR IF INPUT VALUE TOO SMALL
7988 035102 023762 005470 005736 CMP      LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
7989 035110 101365          BHI      16$          ;BR IF INPUT VALUE IS TOO LARGE
7990          ;UPDATE 16-BIT PARAMETER VALUE IN LIST
7991 035112 013761 005470 005660 MOV      LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
7992 035120 000457          BR      28$          ;BR TO RETURN
7993          ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
7994 035122 032737 000001 005470 20$: BIT      #BIT0,LOWOCT ;SEE IF MA IS ODD
7995 035130 001355          BNE      16$          ;BR IF MA IS ODD
7996 035132 023762 005472 005736 CMP      HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
7997 035140 103751          BLO      16$          ;BR IF HIGH WORD IS TOO SMALL
7998 035142 001004          BNE      24$          ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
7999 035144 023762 005470 005734 CMP      LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
8000 035152 103744          BLO      16$          ;BR IF LOW WORD IS TOO SMALL

```

```

8001 035154 023762 005472 005742 24$: CMP HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
8002 035162 101340 BHI 16$ ;BR IF HIGH WORD TOO BIG
8003 035164 001004 BNE 26$ ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
8004 035166 023762 005470 005740 CMP LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
8005 035174 101333 BHI 16$ ;BR IF LOW WORD IS TOO LARGE
8006 ;UPDATE 32-BIT PARAMETER VALUE IN LIST
8007 035176 013761 005470 005660 26$: MOV LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
8008 035204 013761 005472 005662 MOV HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
8009 ;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
8010 035212 004737 035270 JSR PC,MXWRDC ;GET WORD COUNT IN R1-RO
8011 035216 104401 010446 TYPE MAWRDC ;TYPE "MAX WORD COUNT ="
8012 035222 010037 003172 MOV RO,SUML01
8013 035226 010137 003174 MOV R1,SUMHI1
8014 035232 012746 003172 MOV #SUML01,-(SP) ;PUT POINTER ON STACK
8015 035236 004737 055436 JSR PC,$DB20 ;CONVERT TO OCTAL
8016 035242 004737 055752 JSR PC,$SUPRS ;TYPE IT
8017 035246 104401 001315 TYPE $CRLF ;TYPE <CR>,<LF>
8018 035252 062766 000002 000014 ADD #2,14(SP) ;INCREMENT R1 INDEX
8019 035260 062716 000002 28$: ADD #2,(SP) ;GET NORMAL RETURN PC
8020 035264 104410 30$: RESREG ;RESTORE RO-R5
8021 035266 000207 RTS PC ;RETURN
8022
8023
8024
8025
8026
8027
8028
8029 035270 013700 005574
8030 035274 013701 005576
8031 035300 163700 005600
8032 035304 005601
8033 035306 163701 005602
8034 035312 100413
8035 035314 000241
8036 035316 006001
8037 035320 006000
8038 035322 062700 000001
8039 035326 005501
8040 035330 005701
8041 035332 001403
8042 035334 005000
8043 035336 012701 000001
8044 035342 000207 27$: RTS PC ;RETURN
8045
8046
8047
8048
8049
8050
8051
8052
8053 035344 104407
8054 035346 004737 032310 000001
8055 035352 112765 000141
8056 035360 004737 042542

```

```

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

```

```

MXWRDC: MOV MAHILM,RO ;GET LO BITS OF MEM LIM
MOV MAHILM+2,R1 ;HI BITS OF MEM LIM
SUB MA,RJ ;SUBTRACT MA FROM MAHILM
SBC R1
SUB MA+2,R1
BMI 27$ ;IF NEGATIVE, RETURN
CLC ;DIVIDE BY 2 TO GET WORDS
ROR R1
ROR RO
ADD #1,RO ;INCREMENT BY 1 WORD
ADC R1
TST R1
BEQ 27$ ;BR IF WORD COUNT < 65,536 DEC
CLR RO ;MAKE WORD COUNT = 65,536
MOV #1,R1
27$: RTS PC ;RETURN

```

```

;*****
;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 RO-R5
JSR PC,INITSS ;CLEAR S.S. AND PARAMETERS
MOVB #ROSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
JSR PC,DRVCL ;READ STATUS OF THIS DRIVE

```


CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 160
DRVSR - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)

SEQ 0159

8057 035364 104401 011135
8058 035370 104401 011151
8059 035374 016501 000054
8060 035400 012704 055540
8061 035404 010446
8062 035406 012703 000003
8063 035412 006101
8064 035414 006101
8065 035416 006101
8066 035420 006101
8067 035422 006101
8068 035424 006101
8069 035426 010100
8070 035430 042700 177760
8071 035434 052700 000060
8072 035440 110024
8073 035442 005303
8074 035444 001364
8075 035446 105014
8076 035450 004737 055752
8077 035454 104401 001315
8078 035460 104410
8079 035462 000207
8080
8081
8082

4\$:

```

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

```

```

8083      ;*****
8084      ;SBTTL CRTSER - TYPE CARTRIDGE SERIAL NUMBER
8085      ;*THIS SUBROUTINE TYPES "CART. SER. NO. XXXXXXXXXX" (IN OCTAL),
8086      ;*WITH LEADING ZEROS SUPPRESSED.
8087      ;*****
8088      CRTSER: JSR      PC,INITSS      ;CLEAR S.S. AND PARAMETERS
8089      BICB      #8,CFMT,P.CS1H(R5)  ;SET 22 SECTOR FORMAT
8090      TSTB      FORMAT              ;CHECK THE ACTUAL FORMAT
8091      BEQ       4$                  ;BR IF 22 SECTORS
8092      INCB      P.SECT(R5)          ;IF 20 SECTORS, READ SECTOR 1
8093      MOVVB     #RDATA,P.CMND(R5)   ;SET READ COMMAND
8094      LSTCYL     P.CYLN(R5)          ;SET CYL = 632(OCT)/1456
8095      LSTTRK     P.TRCK(R5)         ;SET TRACK = 2
8096      MOV       #RWBUF,P.BALO(R5)  ;SET READ BUFFER ADDRESS
8097      MOV       #-2,P.WC(R5)        ;SET WORD COUNT TO READ 2 WORDS
8098      JSR        PC,DRVCAL          ;READ SERIAL NO. IN BSF
8099      TYPE      ,CART              ;TYPE "CART."
8100      TYPE      ,SERNM             ;TYPE "SER. NO."
8101      MOV        #RWBUF,-(SP)        ;GET POINTER FOR $DB20
8102      JSR        PC,@$DB20          ;CONVERT BINARY TO OCTAL
8103      JSR        PC,@$SSUPRS        ;TYPE CART. SERIAL NO. IN OCTAL
8104      TYPE      ,$CRLF              ;TYPE <CR> AND <LF>
8105      RTN      PC                  ;RETURN
8106
8107
8108
8109      ;*****
8110      ;SBTTL STALL - STALL FOR ST UNIT STALL TIMES
8111      ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
8112      ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
8113      ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
8114      ;* CALL - JSR      PC,STALL
8115      ;*****
8116
8117      STALL:  MOV      R0,-(SP)        ;SAVE R0
8118      MOV      R1,-(SP)        ;SAVE R1
8119      BIT      #BIT08,$SWR      ;APPLY RANDOM STALL ?
8120      BEQ      1$              ;BR IF NOT RANDOM
8121      JSR      PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
8122      MOV      $LONUM,R0       ;GET IT INTO R0
8123      ASR      R0              ;SCALE IT DOWN
8124      ASR      R0
8125      BR       2$              ;GO STALL WITH RANDOM NO.
8126      1$:  MOV      STALLS,R0      ;GET REQUESTED NO. OF STALLS
8127      BEQ      6$              ;RETURN IF NO STALL REQUIRED
8128      2$:  MOV      #14.,R1       ;SET CONSTANT FOR 40 US
8129      4$:  DEC      R1            ;INNER LOOP COUNTER
8130      BNE      4$              ;INNER LOOP BR
8131      DEC      R0              ;OUTER LOOP COUNTER
8132      BNE      2$              ;OUTER LOOP BR
8133      6$:  MOV      (SP)+,R1       ;RESTORE R1
8134      MOV      (SP)+,R0         ;RESTORE R0
8135      RTS      PC              ;RETURN
8136
8137
8138

```

8139
8140
8141
8142
8143
8144
8145
8146 035666 004737 054634
8147 035672 013737 054734 005504
8148 035700 042737 177000 005504
8149 035706 023737 020040 005504
8150 035714 002003
8151 035716 043737 020042 005504
8152 035724 000207
8153
8154
8155
8156
8157
8158
8159
8160 035726 104407
8161 035730 012701 065660
8162 035734 010021
8163 035736 020127 066660
8164 035742 001374
8165 035744 104410
8166 035746 000207
8167
8168
8169
8170
8171
8172
8173
8174
8175
8176 035750 104407
8177 035752 016546 000004
8178 035756 105065 000005
8179
8180 035762 116500 000005
8181 035766 062700 000100
8182 035772 004737 035726
8183 035776 112765 000131 000001
8184 036004 004737 042542
8185 036010 105265 000005
8186 036014 122765 000003 000005
8187 036022 001357
8188 036024 012665 000004
8189 036030 104410
8190 036032 000207
8191
8192
8193
8194

```

*****
;SMTL RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
;THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
;ADDRESS, AND LEAVES IT IN "CYLNR". IT REQUIRES THE SYSMAC
;SUBROUTINE $RAND.
;CALL - JSR PC,RNDADR
*****
RNDADR: JSR PC,$RAND ;GENERATE 2 16-BIT RANDOM NUMBERS
        MOV $LONUM,CYLNR ;GET A RANDOM NUMBER
        BIC #177000,CYLNR ;SCALE IT TO 9 BITS
        CMP LSTCYL,CYLNR ;SEE IF CYL IS TOO BIG
        BGE 2$ ;BR IF CYL IS OK
        BIC HOLD1,CYLNR ;SCALE DOWN TO A VALID CYLINDER
        RTS PC ;RETURN
2$:

```

```

*****
;LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF RO INTO ALL
;256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
*****
LODSEC: SAVREG ;SAVE RO-R5
        MOV #RWBUF,R1 ;GET ADDRESS OF DATA BUF INTO R1
2$: MOV RO,(R1)+ ;PUT WORD INTO BUFFER
        CMP R1,#RWBUF+512. ;SEE IF 256 WORDS WRITTEN YET
        BNE 2$ ;BR IF NOT DONE YET
        RESREG ;RESTORE RO-R5
        RTS PC ;RETURN

```

```

*****
;TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
;FC FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
;TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
;THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.
*****
TRKCHK: SAVREG ;SAVE RO-R5
        MOV P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
        CLRB P.TRCK(R5) ;CLEAR THE TRACK NO.
;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
2$: MOVB P.TRCK(R5),RO ;GET TRACK NO. INTO RO
        ADD #100,RO ;ADD 100(OCT) TO TRACK NO.
        JSR PC,LODSEC ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
        MOVB #WATCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
        JSR PC,DRVCAL ;PERFORM THE WRITE CHECK
        INCB P.TRCK(R5) ;INCREMENT THE TRACK NO.
        CMPB #3,P.TRCK(R5) ;SEE IF DONE WITH ALL TRACKS
        BNE 2$ ;BR IF NOT DONE YET
        MOV (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
        RESREG ;RESTORE RO-R5
        RTS PC ;RETURN

```

```

*****

```

```

8195
8196
8197
8198 036034 104407
8199 036036 012701 000010
8200 036042 012700 006736
8201 036046 004737 054634
8202 036052 013720 054734
8203 036056 013720 054732
8204 036062 005301
8205 036064 001370
8206 036066 104410
8207 036070 000207
8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221 036072 104407
8222 036074 013746 005572
8223 036100 005416
8224 036102 005046
8225 036104 116616 000003
8226 036110 005066 000002
8227 036114 116566 000004 000002
8228 036122 066616 000002
8229 036126 005066 000002
8230 036132 012700 000026
8231 036136 105737 003125
8232 036142 001402
8233 036144 012700 000024
8234 036150 020016
8235 036152 101004
8236 036154 160016
8237 036156 005266 000002
8238 036162 000772
8239 036164 112637 005571
8240 036170 005046
8241 036172 116516 000005
8242 036176 066616 000002
8243 036202 005066 000002
8244 036206 122716 000003
8245 036212 101005
8246 036214 162716 000003
8247 036220 005266 000002
8248 036224 000770
8249 036226 112637 005570
8250 036232 066516 000002

```

```

; *LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
; *INTO THE PATTERN 14 TABLE.
; *****
LODP14: SAVREG      ; SAVE R0-R5
        MOV         #8, R1      ; INIT LOOP COUNTER TO 8.
        MOV         #PAT14, R0  ; GET ADDRESS OF PATTERN 14 BUFFER
4$:      JSR         PC, $RAND    ; GENERATE 2 16-BIT RANDOM NUMBERS
        MOV         $LONUM, (R0)+ ; PUT ONE NUMBER INTO PATTERN
        MOV         $SHNUM, (R0)+ ; PUT OTHER NO. INTO PATTERN
        DEC         R1          ; SEE IF 16 WORDS LOADED YET
        BNE         4$         ; BR IF NOT YET
        RESREG      ; RESTORE R0-R5
        RTS         PC         ; RETURN

```

```

; *****
; SBTTL FINADR - COMPUTE FINAL PACK ADDRESS
; *THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
; *COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
; *WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
; *P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
; *PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
; *THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
; *DATA MISCOMPARE.
; *****
FINADR: SAVREG      ; SAVE R0-R5
        MOV         LASTWC, -(SP) ; STORE WORD COUNT
        NEG         (SP)         ; MAKE IT POSITIVE
18$:     CLR         -(SP)        ; MAKE ROOM ON STACK
        MOVB        3(SP), (SP)  ; STORE NO. OF SECTORS TRANSFERRED
        CLR         2(SP)        ; CLEAR LOCATION ON STACK
        MOVB        P.SECT(R5), 2(SP) ; STORE STARTING SECTOR
        ADD         2(SP), (SP)  ; DETERMINE FINAL SECTOR ADDRESS
        CLR         2(SP)        ; CLEAR NO. OF TRACKS TRANSFERRED
        MOV         #22, R0      ; SET FOR 22 SECTORS
        TSTB        FORMAT      ; DETERMINE THE FORMAT
        BEQ         13$         ; BR IF 22 SECTORS
        MOV         #20, R0      ; SET FOR 20 SECTORS
19$:     CMP         R0, (SP)     ; CHECK FOR SECTOR OVERFLOW
        BHI         20$         ; NO, CHECK IF SECTOR CORRECT
        SUB         R0, (SP)     ; DECREMENT SECTOR COUNT BY 20 OR 22
        INC         2(SP)        ; INCREMENT TRACKS TRANSFERRED
        BR          19$         ; CHECK FOR SECTOR OVERFLOW
20$:     MOVB        (SP)+, FINSEC ; STORE FINAL SECTOR
        CLR         -(SP)        ; MAKE ROOM FOR TRACKS TRANSFERRED
        MOVB        P.TRCK(R5), (SP) ; STORE STARTING TRACK
        ADD         2(SP), (SP)  ; DETERMINE FINAL TRACK ADDRESS
        CLR         2(SP)        ; CLEAR FINAL CYLINDER
21$:     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.CYLN(R5), (SP) ; CALCULATE FINAL CYLINDER

```

8251 036236 011637 005566
8252 036242 005726
8253 036244 104410
8254 036246 000207
8255
8256
8257
8258
8259
8260
8261
8262
8263
8264
8265
8266
8267
8268

MOV (SP),FINCYL ;STORE FINAL CYLINDER
TST (SP)+ ;CLEAN OFF STACK
RESREG ;RESTORE RO-R5
RTS PC ;RETURN

```

;*****
;SRTL LOOBUF - 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.
;*****

```

8269 036250 104407
8270 036252 013702 005564
8271 036256 005737 005536
8272 036262 001007
8273 036264 012700 006036
8274 036270 012746 000400
8275 036274 012701 065660
8276 036300 000463
8277 036302 005000
8278 036304 032701 000001
8279 036310 001003
8280 036312 005200
8281 036314 006201
8282 036316 000772
8283 036320 006300
8284 036322 006300
8285 036324 006300
8286 036326 006300
8287 036330 006300
8288 036332 062700 006036
8289 036336 012746 000020
8290 036342 013701 005604
8291 036346 005737 057732
8292 036352 100036
8293
8294 036354 013737 003176 172354
8295 036362 052737 000001 177572
8296 036370 042701 160000
8297 036374 052701 140000
8298 036400 010003
8299 036402 011604
8300 036404 012321
8301 036406 032701 020000
8302 036412 001405
8303 036414 062737 000200 172354
8304 036422 042701 020000
8305 036426 005302
8306 036430 001403

```

LOOBUF: SAVREG ;SAVE RO-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
ADD #PAT00,R0 ;GET ADDRESS OF DESIRED PATTERN
MOV #16,-(SP) ;SET PATTERN WORD COUNT
MOV PMA,R1 ;SET BUFFER ADDRESS
TST $KTI1 ;SEE IF MEM MGT PRESENT
BPL 30$ ;BR IF NOT PRESENT
;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
MOV SAVPAR,2#KIPAR6 ;SET UP WORKING PAR
BIS #BIT0,2#SRO ;TURN ON MEMORY MANAGEMENT
BIC #160000,R1 ;FORCE RELOCATION THRU KIPAR6
BIS #140000,R1
22$: MOV R0,R3 ;GET A COPY OF PATTERN ADDRESS
MOV (SP),R4 ;INIT PATTERN WORD COUNT
24$: MOV (R3)+,(R1)+ ;LOAD A DATA WORD INTO BUFFER
BIT #BIT13,R1 ;SEE IF OVERFLOW TO NEXT PAGE
BEQ 26$ ;BR IF NO OVERFLOW
ADD #200,2#KIPAR6 ;INCREMENT PAR BY 4K FOR NEW PAGE
BIC #BIT13,R1 ;SET PAGE = 6 AGAIN
26$: DEC R2 ;DECREMENT WORD COUNTER
BEQ 28$ ;BR IF ALL DONE

```

8307	036432	005304			DEC	R4	; DECREMENT PATTERN WORD COUNT
8308	036434	001363			BNE	24\$; BR IF NOT DONE WITH PATTERN YET
8309	036436	000760			BR	22\$; BR TO REPEAT THE PATTERN
8310	036440	042737	000001	177572	28\$: BIC	#BIT0, 2#SR0	; DISABLE MEMORY MANAGEMENT
8311	036446	000410			BR	44\$; GO TO RETURN
8312					; BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE		
8313	036450	010003			30\$: MOV	R0, R3	; GET A COPY OF PATTERN ADDRESS
8314	036452	011604				(SP), R4	; INIT PATTERN WORD COUNT
8315	036454	012321			34\$: MOV	(R3)+, (R1)+	; LOAD A DATA WORD INTO BUFFER
8316	036456	005302			DEC	R2	; DECREMENT WORD COUNTER
8317	036460	001403			BEQ	44\$; BR IF ALL DONE
8318	036462	005304			DEC	R4	; DECREMENT PATTERN WORD COUNT
8319	036464	001373			BNE	34\$; BR IF NOT DONE WITH PATTERN YET
8320	036466	000770			BR	30\$; BR TO REPEAT THE PATTERN
8321	036470	005726			44\$: TST	(SP)+	; POP THE STACK
8322	036472	104410			RESREG		; RESTORE R0-R5
8323	036474	000207			RTS	PC	; RETURN
8324							
8325							
8326							
8327							
8328					; *****		
8329					; SBTL CMPBUF - SOFTWARE COMPARE DATA		
8330					; *THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED		
8331					; *TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF		
8332					; *PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS		
8333					; *00-15 (QUICK VERIFY DEFAULT DATA TEST).		
8334					; *IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE		
8335					; *REPEATING DATA PATTERN TO COMPARE AGAINST.		
8336					; *****		
8337	036476	104407			CMPBUF: SAVREG		; SAVE R0-R5
8338	036500	112737	000040	064176	MOVB	#40, DH701+38.	; RESTORE ERROR MSG PARAMS
8339	036506	012737	000007	065552	MOV	#7, DF25+2	
8340	036514	013702	005564		MOV	WDSXFR, R2	; SET NO. OF WORDS
8341	036520	005037	005534		CLR	SCRACH	; CLEAR COMPARE ERROR COUNT
8342	036524	004737	043720		JSR	PC, REPSUP	; STORE PREV CMND FOR POSS. PRINT
8343	036530	005737	005536		TST	PATRN	; SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
8344	036534	001007			BNE	3\$; BR IF NOT
8345	036536	012700	006036		MOV	#PAT00, R0	; GET DATA PATTERN STARTING ADDR
8346	036542	012746	000400		MOV	#256, -(SP)	; SET PATTERN WORD COUNT
8347	036546	012701	065660		MOV	#RWBUFF, R1	; SET BUFFER ADDRESS
8348	036552	000466			BR	30\$; PROCEED
8349	036554	005000			3\$: CLR	R0	; INIT PATTERN NO.
8350	036556	032701	000001		4\$: BIT	#BIT0, R1	; SEE IF THIS BIT IS SET
8351	036562	001003			BNE	6\$; BR IF THIS BIT IS SET
8352	036564	005200			INC	R0	; INCREMENT PATTERN NUMBER
8353	036566	006201			ASR	R1	; SHIFT TO EXAMINE NEXT BIT
8354	036570	000772			BR	4\$; BR TO CHECK NEXT BIT
8355	036572	006300			6\$: ASL	R0	; MULTIPLY PATTERN NO. BY 32(DEC)
8356	036574	006300			ASL	R0	
8357	036576	006300			ASL	R0	
8358	036600	006300			ASL	R0	
8359	036602	006300			ASL	R0	
8360	036604	062700	006036		ADD	#PAT00, R0	; GET ADDRESS OF DESIRED PATTERN
8361	036610	012746	000020		MOV	#16, -(SP)	; PATTERN WORD COUNT
8362	036614	013701	005604		MOV	PMA, R1	; SET BUFFER ADDRESS
8363	036620	005737	057732		TST	\$K11	; SEE IF MEM MGT PRESENT

8363	036624	100041			BPL	30\$;BR IF NOT PRESENT
8364					;COMPARE	LOOP FOR MEM MGT STARTS	HERE
8365	036626	013737	003176	172354	MOV	SAVPAR,2#KIPAR6	;SET UP WORKING PAR
8366	036634	052737	000001	177572	BIS	#BIT0,2#SRO	;TURN ON MEM MGT
8367	036642	042701	160000		BIC	#160000,R1	;FORCE RELOCATION THRU KIPAR6
8368	036646	052701	140000		BIS	#140000,R1	
8369	036652	010003			22\$: MOV	RO,R3	;GET A COPY OF PATTERN ADDRESS
8370	036654	011604			MOV	(SP),R4	;INIT PATTERN WORD COUNT
8371	036656	022321			24\$: CMP	(R3)+,(R1)+	;COMPARE DATA WORD TO PATTERN WORD
8372	036660	001404			BEQ	25\$;BR IF NO ERROR
8373	036662	013737	172354	001216	MOV	2#KIPAR6,\$REG16	;SET PAR FOR PRINTOUT
8374	036670	000432			BR	35\$;GO HANDLE ERROR
8375	036672	032701	020000		25\$: BIT	#BIT13,R1	;SEE IF OVERFLOW TO NEXT PAGE
8376	036676	001405			BEQ	26\$;BR IF NO OVERFLOW
8377	036700	062737	000200	172354	ADD	#200,2#KIPAR6	;INCR PAR BY 4K FOR NEW PAGE
8378	036706	042701	020000		BIC	#BIT13,R1	;SET PAGE = 6 AGAIN
8379	036712	005302			26\$: DEC	R2	;DECREMENT WORD COUNTER
8380	036714	001002			BNE	27\$;BR IF NOT ALL DONE YET
8381	036716	000137	037362		JMP	54\$;ALL DONE - GET OUT
8382	036722	005304			27\$: DEC	R4	;DECREMENT PATTERN WORD COUNT
8383	036724	001354			BNE	24\$;BR IF NOT DONE WITH PATTERN YET
8384	036726	000751			BR	22\$;BR TO REPEAT THE PATTERN
8385					;COMPARE	LOOP FOR NO MEM MGT STARTS	HERE
8386	036730	010003			30\$: MOV	RO,R3	;GET COPY OF PATTERN ADDRESS
8387	036732	011604			MOV	(SP),R4	;INIT PATTERN WORD COUNT
8388	036734	022321			34\$: CMP	(R3)+,(R1)+	;COMPARE DATA WORD TO PATTERN WORD
8389	036736	001002			BNE	31\$;BR IF COMPARE ERROR
8390	036740	000137	037342		JMP	40\$;JUMP IF DATA COMPARES OK
8391	036744	105037	064176		31\$: CLRB	DH701+38.	;ADJUST DATA HEADER FOR MSG
8392	036750	012737	000005	065552	MOV	#5,DF25+2	;ADJUST ERROR DATA WORD COUNT
8393					;COMMON	COMPARE	ERROR HANDLER
8394	036756	010237	001202		35\$: MOV	R2,\$REG10	;GET WORD NO.
8395	036762	163737	005564	001202	SUB	WDSXFR,\$REG10	
8396	036770	013737	001202	005572	MOV	\$REG10, LASTWC	;GET 2'S COMP OF WORD NO.
8397	036776	004737	036072		JSR	PC,FINADR	;COMPUTE ACTUAL PACK ADRS
8398	037002	104407			SAVREG		;SAVE RO-R5
8399	037004	013700	005566		MOV	FINCYL,RO	;GET CYL
8400	037010	113701	005570		MOVB	FINTRK,R1	;GET TRACK
8401	037014	113702	005571		MOVB	FINSEC,R2	;GET SECTOR
8402	037020	004737	042436		JSR	PC,BDSACK	;SEE IF THIS SECTOR LISTED BAD
8403	037024	104410			RESREG		;RESTORE RO-R5
8404	037026	032737	001000	005474	BIT	#BADSEC,RECODE	
8405	037034	001132			BNE	50\$;BR IF LISTED- DON'T REPORT ERROR
8406	037036	005737	005534		TST	SCRACH	;CHECK THE ERROR COUNT
8407	037042	001024			BNE	36\$;BR IF THIS IS NOT FIRST ERROR
8408	037044	105737	003143		TSTB	UBMPRS	;SEE IF UNIBUS MAP FPRESENT
8409	037050	001411			BEQ	46\$;BR IF NOT
8410	037052	013737	005262	001174	MOV	CRMPHD,\$REG5	;GET CURRENT MAP REG 0
8411	037060	013737	005260	001176	MOV	CRMPLO,\$REG6	
8412	037066	104116			ERROR	116	;DATA MISCOMPARE (11/70)
8413	037070	104117			ERROR	117	
8414	037072	000401			BR	48\$	
8415	037074	104034			46\$: ERROR	34	;TYPE HEADING FOR ERROR MSG
8416	037076	012737	177777	001174	48\$: MOV	#-1,\$REG5	;INIT CYL NO.
8417	037104	005037	001176		CLR	\$REG6	
8418	037110	005037	001200		CLR	\$REG7	

8419	037114	005237	005534	36\$:	INC	SCRACH	; INCREMENT THE ERROR COUNT
8420	037120	032777	000001	142012	BIT	#BIT0,ASWR	; SEE IF ALL ERRORS SHOULD BE REPORTED
8421	037126	001004			BNE	38\$; BR TO REPORT ALL ERRORS
8422	037130	022737	000012	005534	CMP	#10.,SCRACH	; SEE IF 10(DEC) ERRORS YET
8423	037136	002511			BLT	54\$; BR IF ERROR LIMIT EXCEEDED
8424	037140	023737	001174	005566	38\$:	CMP	\$REG5,FINCYL
8425	037146	001010			BNE	42\$; SEE IF DIFFERENT CYL
8426	037150	123737	001176	005570	CMPB	\$REG6,FINTRK	; BR IF YES
8427	037156	001004			BNE	42\$; SEE IF DIFFERENT TRACK
8428	037160	123737	001200	005571	CMPB	\$REG7,FINSEC	; BR IF YES
8429	037166	001412			BEQ	44\$; SEE IF DIFFERENT SECTOR
8430	037170	013737	005566	001174	42\$:	MOV	FINCYL,\$REG5
8431	037176	113737	005570	001176	MOV	FINTRK,\$REG6	; BR IF SAME PACK ADDRESS
8432	037204	113737	005571	001200	MOV	FINSEC,\$REG7	; SET NEW PACK ADRS FOR PRINTOUT
8433	037212	104115			ERROR	115	; TYPE NEW PACK ADDRESS
8434	037214	005437	001202		44\$:	NEG	\$REG10
8435	037220	016337	177776	001204	MOV	-2(R3),\$REG11	; GET WORD NO.
8436	037226	016137	177776	001206	MOV	-2(R1),\$REG12	; GET GOOD DATA
8437	037234	013737	001202	001212	MOV	\$REG10,\$REG14	; GET BAD DATA
8438	037242	005037	001210		CLR	\$REG13	; COMPUTE PHYSICAL ADDRESS
8439	037246	006137	001212		ROL	\$REG14	
8440	037252	006137	001210		ROL	\$REG13	
8441	037256	063737	005604	001212	ADD	PMA,\$REG14	
8442	037264	005537	001210		ADC	\$REG13	
8443	037270	063737	005606	001210	ADD	PMA+2,\$REG13	
8444	037276	010137	001214		MOV	R1,\$REG15	; GET VIRT. ADRS FOR PRINTOUT
8445	037302	162737	000002	001214	SUB	#2,\$REG15	
8446	037310	104063			ERROR	63	; TYPE GOOD AND BAD DATA, MEM. ADRS.
8447	037312	032777	000100	141620	BIT	#BIT6,ASWR	; SEE IF JUST 1 ERROR SHOULD BE REPORTED
8448	037320	001020			BNE	54\$; BR IF JUST 1 ERROR SHOULD BE REPORTED
8449	037322	005737	005536		50\$:	TST	PATRN
8450	037326	001405			BEQ	40\$; SEE IF DEFAULT DATA TEST
8451	037330	005737	057732		TST	\$KT11	; BR IF YES
8452	037334	100002			BPL	40\$; SEE IF MEM MGT PRESENT
8453	037336	000137	036672		JMP	25\$; BR IF NOT PRESENT
8454	037342	005302			40\$:	DEC	R2
8455	037344	001406			BEQ	54\$; GO HANDLE MEM MGT
8456	037346	005304			DEC	R4	; DECREMENT WORD COUNTER
8457	037350	001002			BNE	53\$; BR IF ALL DONE
8458	037352	000137	036730		JMP	30\$; DECREMENT PATTERN WORD COUNT
8459	037356	000137	036734		53\$:	JMP	34\$
8460	037362	005737	057732		54\$:	TST	\$KT11
8461	037366	100003			BPL	56\$; SEE IF MEM MGT PRESENT
8462	037370	042737	000001	177572	BIC	#BIT0,ASRO	; BR IF NOT PRESENT
8463	037376	005726			56\$:	TST	(SP)+
8464	037400	104410			RESREG		; DISABLE MEM MGT
8465	037402	000207			RTS	PC	; POP THE STACK
8466							; RESTORE R0-R5
8467							; RETURN
8468							
8469							
8470							
8471							
8472							
8473							
8474							

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



```
8475      ;*NO ACTION IS TAKEN.
8476      ;*  CALL -   JSR      PC,CTLOUT
8477      ;*  OR  -   CKEXIT
8478      ;*****
8479 037404 122737 000001 003120 CTLOUT: CMPB    #1,TSTING      ;SEE IF CURRENTLY RUNNING TESTS
8480 037412 001077          BNE     10$          ;BR IF NOT RUNNING TESTS
8481 037414 005737 005522      TST     INTCHR      ;SEE IF ANY TTY INPUT
8482 037420 001474          BEQ     10$          ;BR IF NO INPUT
8483 037422 105737 003116      TSTB    MDFLAG      ;SEE IF DEFAULT MODE RUN
8484 037426 001446          BEQ     12$          ;BR IF YES
8485 037430 122737 000003 005522 CMPB    #003,INTCHR  ;SEE IF (↑C) TYPED
8486 037436 001033          BNE     4$          ;BR IF NOT (↑C)
8487 037440 012700 014434      MOV     #DRVST,RO    ;SET RETURN ADDR = DRVST
8488 037444 105737 003133      TSTB    XOVLAD      ;SEE IF XXDP CURRENTLY OVERLAID
8489 037450 001402          BEQ     6$          ;BR IF NOT
8490 037452 004737 032050      JSR     PC,GETXDP    ;RESTORE SAVED XXDP, IF NECESSARY
8491 037456 000005          6$:  RESET          ;RESET ALL DEVICES
8492 037460 005037 005522      CLR     INTCHR      ;CLEAR TTY CHAR BUFFER WORD
8493 037464 005037 001304      CLR     $TIMES      ;CLEAR THE ITER. COUNT
8494 037470 105037 003134      CLRB    XDPSVD      ;CLEAR THE XXDP SAVED FLAG
8495 037474 012737 044200 003046 MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADDRESS
8496 037502 012706 001100      MOV     #STACK,SP    ;RESET THE STACK
8497 037506 112765 000113 000001 MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
8498 037514 004737 042542      JSR     PC,DRVCL     ;DO CLEANUP RECALIBRATE
8499 037520 005037 001102      CLR     $STNM      ;CLEAR THE TEST NO.
8500 037524 000110          JMP     $RO          ;EXIT FROM TESTS
8501 037526 122737 000032 005522 4$:  CMPB    #032,INTCHR  ;SEE IF (↑Z) TYPED
8502 037534 001016          BNE     7$          ;BR IF NOT (↑Z)
8503 037536 012700 016014      MOV     #INPUTP,RO   ;SET RETURN ADDR = INPUTP
8504 037542 000740          BR      2$          ;TAKE EXIT
8505 037544 122737 000003 005522 12$: CMPB    #003,INTCHR  ;SEE IF (↑C) TYPED
8506 037552 001007          BNE     7$          ;BR IF NOT
8507 037554 104401 007704      TYPE    ,HLTRQD     ;TYPE "HALT REQUESTED"
8508 037560 104401 007654      TYPE    ,CNTRDY     ;TYPE "PRESS CONT WHEN RDY"
8509 037564 012700 046150      MOV     #HLTPRG,RO   ;SET HALT ADDRESS
8510 037570 000725          BR      2$          ;TAKE EXIT
8511 037572 122737 000007 005522 7$:  CMPB    #007,INTCHR  ;SEE IF (↑G) TYPED
8512 037600 001002          BNE     8$          ;BR IF NOT (↑G)
8513 037602 004737 031126      JSR     PC,GTSWRG    ;OPEN SOFTWARE SWR FOR MODIFICATION
8514 037606 004737 030576      8$:  JSR     PC,PREPKB   ;ENABLE KBD INPUT AGAIN
8515 037612 000207          10$:  RTS      PC      ;RETURN
8516
8517
8518
8519
8520      ;*****
8521      ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO DISK AND PERFORM
8522      ;*A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
8523      ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
8524      ;*USED.
8525      ;*****
8526 037614 104407      WRTSEC: SAVREG          ;SAVE RO-R5
8527      ;LOAD THE R/W BUFFER WITH DATA
8528      MOV     #RWBUF,RO    ;BUFFER ADDRESS
8529      MOV     #400,R1      ;400(OCT) WORDS
8529 037626 010320      4$:  MOV     R3,(RO)+    ;LOAD A BUFFER WORD
```

```

        DEC     R1                ;DECR COUNTER
        BNE     4$                ;BR IF NOT DONE YET
;SET UP PARAMETERS
        MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
        MOV     #-400,P.WC(R5)   ;SET WORD COUNT
;WRITE THE DATA
        MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
        BIT     #BIT1,2$WR       ;SEE IF WRITES INHIBITED
        BNE     6$                ;BR IF YES
        JSR     PC,DRVCL         ;WRITE THE DATA
;PERFORM WRITE CHECK
6$:      MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
        JSR     PC,DRVCL         ;PERFORM WRITE CHECK
        RESREG                ;RESTORE R0-R5
        RTS     PC                ;RETURN

```

```

*****
*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       LDMM0                      ;PROCEED
LDMEM2: SAVREG                                ;SAVE R0-R5
        MOV      #-1,R4                     ;R4=177777 INDICATES LDMEM2
LDMM0:  MOV      WDSXFR,R2                  ;GET NO. OF WORDS
        MOV      PMA,R1                    ;GET MEM ADRS
        TST      $KT11                     ;SEE IF MEM MGT
        BPL      6$                        ;BR IF NOT
        BIC      #160000,R1                ;FORCE RELOCATION THRU PAR6
        BIS      #140000,R1
        MOV      SAVPAR,2*#KIPAR6          ;INIT PAR6
        BIS      #BIT0,2*#SRO              ;TURN ON MEM MGT
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$:     TST      $KT11                     ;SEE IF MEM MGT
        BPL      10$                       ;BR IF NOT
        BIT      #BIT13,R1                 ;SEE IF OVERFLOW TO NEXT PAGE
        BEQ      10$                       ;BR IF NO OVERFLOW
        ADD      #200,2*#KIPAR6            ;INCR PAR BY 4K FOR NEW PAGE
        BIC      #BIT13,R1                 ;SET PAGE = 6 AGAIN
10$:    DEC      R2                        ;DECREMENT COUNTER
        BNE      6$                        ;BR IF NOT DONE LOADING YET
        TST      $KT11                     ;SEE IF MEM MGT
        BPL      12$                       ;BR IF NOT
        BIC      #BIT0,2*#SRO              ;TURN OFF MEM MGT
12$:    RESREG                                ;RESTORE R0-R5
        RTS      PC                        ;RETURN

```

```
8586
8587
8588
8589
8590
8591
8592
8593
8594
8595
8596 040054 104407
8597 040056 005004
8598 040060 012703 000001
8599 040064 000403
8600 040066 104407
8601 040070 012704 177777
8602 040074 013702 005564
8603 040100 013701 005604
8604 040104 005037 005534
8605 040110 112737 000040 064176
8606 040116 012737 000007 065552
8607 040124 004737 043720
8608 040130 005737 057732
8609 040134 100012
8610 040136 042701 160000
8611 040142 052701 140000
8612 040146 013737 003176 172354
8613 040154 052737 000001 177572
8614 040162 020321
8615 040164 001575
8616
8617 040166 010237 001202
8618 040172 163737 005564 001202
8619 040200 013737 001202 005572
8620 040206 004737 036072
8621 040212 104407
8622 040214 013700 005566
8623 040220 113701 005570
8624 040224 113702 005571
8625 040230 004737 042436
8626 040234 104410
8627 040236 032737 001000 005474
8628 040244 001145
8629 040246 005737 057732
8630 040252 100406
8631 040254 105037 064176
8632 040260 012737 000005 065552
8633 040266 000403
8634 040270 013737 172354 001216
8635 040276 005737 005534
8636 040302 001024
8637 040304 105737 003143
8638 040310 001411
8639 040312 013737 005262 001174
8640 040320 013737 005260 001176
8641 040326 104116

*****
*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.
*BOTHSUBROUTINES 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           CKM0        ;PROCEED
          CKMEM2: SAVREG          ;SAVE R0-R5
          MOV           #-1,R4      ;R4=177777 INDICATES CKMEM2
          CKM0: 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,DF25+2
          JSR        PC,REPSUP     ;STORE PREV CMND FOR POSS. PRINT
          TST        $KT11        ;SEE IF MEM MGT
          BPL        6$           ;BR IF NOT
          BIC        #160000,R1    ;FORCE RELOCATION THRU PAR6
          BIS        #140000,R1
          MOV        SAVPAR,0#KIPAR6 ;INIT PAR6
          BIS        #BIT0,0#SRO   ;TURN ON MEM MGT
          CMP        R3,(R1)+      ;COMPARE A DATA WORD
          BEQ        16$          ;BR IF DATA COMPARES OK
          ;COMPARE ERROR HANDLER
          MOV        R2,$REG10     ;GET WORD NO.
          SUB        WDSXFR,$REG10
          MOV        $REG10, LASTWC ;GET 2'S COMP OF WORD NO.
          JSR        PC,FINADR     ;COMPUTE ACTUAL PACK ADRS
          SAVREG          ;SAVE R0-R5
          MOV        FINCYL,R0     ;GET CYL
          MOV        FINTRK,R1     ;GET TRACK
          MOV        FINSEC,R2     ;GET SECTOR
          JSP        PC,BDSACK     ;SEE IF THIS SECTOR LISTED BAD
          RESREG          ;RESTORE R0-R5
          BIT        #BADSEC,RECODE ;BR IF LISTED- DON'T REPORT ERROR
          BNE        16$
          TST        $KT11        ;SEE IF MEM MGT
          BMI        8$           ;BR IF YES
          CLRB       DH701+38.    ;ADJUST DATA HEADER FOR MSG
          MOV        #5,DF25+2    ;ADJ. ERROR DATA WORD COUNT
          BR        9$
          8$: MOV        0#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
          9$: TST        SCRACH    ;SEE IF FIRST ERROR IN THIS BLOCK
          BNE        10$          ;BR IF NOT FIRST ERROR
          TSTB       UBMPRS       ;SEE IF UNIBUS MAP PRESENT
          BEQ        17$          ;BR IF NOT
          MOV        CRMPHO,$REG5  ;CURRENT UB MAP REG 0
          MOV        CRMPLO,$REG6
          ERROR      116          ;DATA MISCOMPARE (11/70)
```

8642	040330	104117			ERROR	117	
8643	040332	000401			BR	115	
8644	040334	104034			ERROR	34	;TYPE HEADING FOR ERROR MSG
8645	040336	012737	177777	001174	115: MOV	#-1,\$REG5	;INIT CYL NO.
8646	040344	005037	001176		CLR	\$REG6	
8647	040350	005037	001200		CLR	\$REG7	
8648	040354	005237	005534		105: INC	SCRACH	;INCREMENT THE ERROR COUNT
8649	040360	032777	000001	140552	BIT	#BIT0,\$SWR	;SEE IF ALL ERRORS SHOULD BE REPORTED
8650	040366	001004			BNE	125	;BR TO REPORT ALL ERRORS
8651	040370	022737	000012	005534	CMP	#10.,SCRACH	;SEE IF 10(DEC) ERRORS YET
8652	040376	002512			BLT	215	;BR IF ERROR LIMIT EXCEEDED
8653	040400	023737	001174	005566	125: CMP	\$REG5,FINCYL	;SEE IF DIFFERENT CYL
8654	040406	001010			BNE	145	;BR IF YES
8655	040410	123737	001176	005570	CMPR	\$REG6,FINTRK	;SEE IF DIFFERENT TRACK
8656	040416	001004			BNE	145	;BR IF YES
8657	040420	123737	001200	005571	CMPB	\$REG7,FINSEC	;SEE IF DIFFERENT SECTOR
8658	040426	001412			BEQ	155	;BR IF SAME PACK ADDRESS
8659	040430	013737	005566	001174	145: MOV	FINCYL,\$REG5	;SET NEW PACK ADR FOR PRINTOUT
8660	040436	113737	005570	001176	MOV	FINTRK,\$REG6	
8661	040444	113737	005571	001200	MOV	FINSEC,\$REG7	
8662	040452	104115			ERROR	115	;TYPE NEW PACK ADRS
8663	040454	005437	001202		155: NEG	\$REG10	;GET WORD NO.
8664	040460	010337	001204		MOV	R3,\$REG11	;GOOD DATA
8665	040464	016137	177776	001206	MOV	-2(R1),\$REG12	;BAD DATA
8666	040472	013737	001202	001212	MOV	\$REG10,\$REG14	;COMPUTE PHYSICAL ADDRESS
8667	040500	005037	001210		CLR	\$REG13	
8668	040504	006137	001212		ROL	\$REG14	
8669	040510	006137	001210		ROL	\$REG13	
8670	040514	063737	005604	001212	ADD	PMA,\$REG14	
8671	040522	005537	001210		ADC	\$REG13	
8672	040526	063737	005606	001210	ADD	PMA+2,\$REG13	
8673	040534	010137	001214		MOV	R1,\$REG15	;GET VIRT ADRS FOR PRINTOUT
8674	040540	162737	000002	001214	SUB	#2,\$REG15	
8675	040546	104063			ERROR	63	;TYPE GOOD AND BAD DATA
8676	040550	032777	000100	140362	BIT	#BIT6,\$SWR	;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8677	040556	001022			BNE	215	;BR IF JUST 1
8678					;PROCEED WITH COMPARISONS		
8679	040560	005704			165: TST	R4	;SEE WHICH DATA DESIRED
8680	040562	001001			BNE	185	;BR IF NOT INCREASING DATA
8681	040564	005203			INC	R3	;INCREMENT THE DATA WORD
8682	040566	005737	057732		185: TST	\$KT11	;SEE IF MEM MGT
8683	040572	100010			BPL	205	;BR IF NOT
8684	040574	032701	020000		BIT	#BIT13,R1	;SEE IF OVERFLOW TO NEXT PAGE
8685	040600	001405			BEQ	205	;BR IF NO OVERFLOW
8686	040602	062737	000200	172354	ADD	#200,\$KIPAR6	;INCR. PAR BY 4K FOR NEW PAGE
8687	040610	042701	020000		BIC	#BIT13,R1	;SET PAGE = 6 AGAIN
8688	040614	005302			205: DEC	R2	;DECR COUNTER
8689	040616	001402			BEQ	215	;BR IF DONE COMPARING
8690	040620	000137	040162		JMP	65	;CONTINUE COMPARING WORDS
8691	040624	005737	057732		215: TST	\$KT11	;SEE IF MEM MGT PRESENT
8692	040630	100003			BPL	225	;BR IF NOT
8693	040632	042737	000001	177572	BIC	#BIT0,\$SRO	;TURN OFF MEM MGT
8694	040640	104410			225: RESREG		;RESTORE R0-R5
8695	040642	000207			RTS	PC	;RETURN
8696							
8697							

8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737
8738
8739
8740
8741
8742
8743
8744
8745
8746
8747
8748
8749
8750
8751
8752
8753

040644	013765	005660	000002
040652	113765	005506	000074
040660	113765	005664	000035
040666	023737	005660	020036
040674	003403		
040676	013765	020036	000002
040704	012737	065660	005600
040712	005737	000170	
040716	001003		
040720	062737	006000	005600
040726	042737	003777	005600
040734	062737	004000	005600
040742	005037	005602	
040746	000207		
040750			
040750	004737	037404	
040754	105037	003123	
040760	004737	042622	
040764	010537	041002	
040770	012737	041036	000004
040776	004737	052604	
041002	000000		
041004	004737	047154	
041010	105737	003123	
041014	001012		
041016	010304		
041020	001402		
041022	005304		
041024	001376		
041026	005737	160000	
041032	104111		
041034	000401		
041036	022626		
041040	000761		
041042	012737	000006	000004
041050	000207		

```

*****
;SETUPM - SUBROUTINE TO INIT PARAMS FOR NPR/MEMORY TESTS.
*****
SETUPM: MOV     FC,P.CYLN(R5) ;SET CYL
        MOV     FS,P.SECT(R5) ;SET SECTOR
        MOV     FT,P.TRCK(R5) ;SET TRACK
        CMP     FC,LCM6       ;SEE IF CYL SHOULD BE SCALED DOWN
        BLE     2$           ;BR IF NOT
        MOV     LCM6,P.CYLN(R5) ;SCALE DOWN THE CYLINDER
        MOV     #RWBUFF,MA    ;GET MEMORY ADDRESS
        TST     KILLDR        ;SEE IF LOADER CAN BE OVERLAYED
        BNE     4$           ;BR IF YES
        ADD     #6000,MA      ;LEAVE ROOM TO SAVE LOADER
        BIC     #3777,MA
        ADD     #4000,MA
        CLR     MA+2
        RTS     PC           ;RETURN

```

```

*****
; REFNEM - THIS SUBROUTINE PERFORMS A SUBSYSTEM COMMAND, WHILE
; REPEATEDLY REFERENCING A NON-EXISTENT MEMORY LOCATION (760000). THIS
; IS DONE FOR THE PURPOSE OF CAUSING UNIBUS CONTENTION DURING DISK
; NPR'S, SINCE THE BUS IS SIEZED FOR 5-20 USEC AT EACH NEM TIME-OUT.
; ALL DRIVER PARAMETERS (INCLUDING THE COMMAND) MUST BE SET IN THE
; PARAMETER BLOCK ON ENTRY, AND R3 MUST CONTAIN THE TIMING CONSTANT
; WHICH REGULATES THE FREQUENCY OF NEM REFERENCES, WHILE WAITING FOR
; COMMAND COMPLETION.
*****
REFNEM: JSR     PC,CTLOUT    ;CHECK FOR (1C) OR (1Z) KBD INPUT
        CLRB    DONE        ;CLEAR THE DONE FLAG
        JSR     PC,STRCMD    ;STORE PREV AND CURRENT COMMANDS
        MOV     R5,4$        ;SET PARAM BLK ADDRESS
        MOV     #20$,2$ERRVEC ;SET TIME-OUT VECTOR ADDRESS
        JSR     PC,C.INIT    ;CALL DRIVER
        .WORD
4$:      JSR     PC,W.WTCH    ;CALL WATCH DOG
6$:      TST     DONE        ;DONE SET ?
        BNE     26$         ;BR IF YES
        MOV     R3,R4        ;GET COPY OF R3
        BEQ     12$         ;BR IF TIMER = 0
        DEC     R4           ;DECREMENT TIMER
        BNE     10$         ;BR IF NOT DONE TIMING
10$:     TST     2$160000    ;REFERENCE NON-EXISTENT MEMORY
        ERROR   111         ;"NO NEM WHEN EXPECTED"
        BR      22$
20$:     CMP     (SP)+,(SP)+ ;RESTORE THE STACK
22$:     BR      6$         ;KEEP WAITING FOR COMMAND COMPLETION
26$:     MOV     #6,2$ERRVEC ;RESTORE TIME-OUT VECTOR
        RTS     PC         ;RETURN

```

```

*****

```

```

8754      ;*INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
8755      ;*****
8756      INCRMA: SAVREG          ;SAVE R0-R5
8757      CLR          R1
8758      MOV          P.WC(R5),R0 ;GET WORD COUNT
8759      BNE          4$          ;BR IF NOT 65,536(DEC)
8760      INC          R1          ;SET HI BIT
8761      BR           6$          ;CONTINUE
8762      4$: NEG          R0          ;GET TRUE WORD COUNT
8763      6$: CLC              ;DOUBLE THE WORD COUNT TO GET BYTES
8764      ROL          R0
8765      ROL          R1
8766      ADD          R0,MA          ;ADD IT TO COMPUTE NEW MA
8767      ADC          MA+2
8768      ADD          R1,MA+2
8769      RESREG
8770      RTS          PC          ;RESTORE R0-R5
8771      ;RETURN
8772
8773      ;*****
8774      ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
8775      ;*INTO BSSOFT.
8776      ;*****
8777      REDBSF:
8778      JSR          PC,INITSS      ;INIT THE S.S.
8779      BICB          #8,CFMT,P.CS1H(R5) ;SET 22 SECTOR FMT
8780      MOV          #RDATA,P.CMND(R5) ;SET READ DATA COMMAND
8781      MOV          LSTCYL,P.CYLN(R5) ;SET CYL = 632/1456
8782      MOV          #2,P.TRCK(R5) ;SET TRACK = 2
8783      MOV          #10,R3          ;SET FACTORY BSF SECTOR LIMIT
8784      TSTB          FORMAT
8785      BEQ          6$          ;BR IF 22 SECTORS
8786      INCB          P.SECT(R5) ;SET STARTING FACTORY BSF SECTOR NO.
8787      INC          R3
8788      6$: MOV          #-400,P.WC(R5) ;SET WORD COUNT FOR 1 SECTOR
8789      MOV          #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
8790      MOV          #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
8791      CLRB          WCEFLG          ;INIT THE ERROR FLAG
8792      JSR          PC,DRVCAL          ;READ THE FACTORY BSF
8793      TSTB          WCEFLG          ;SEE IF ANY ERRORS
8794      BEQ          12$          ;BR IF NOT
8795      ADD          #2,P.SECT(R5) ;CHECK NEXT SECTOR
8796      CMPB          P.SECT(R5),R3 ;SEE IF LIMIT EXCEEDED YET
8797      BNE          8$          ;BR IF NOT YET
8798      JSR          PC,REPSUP          ;GATHER STATUS FOR PRINTOUT
8799      ERROR          120          ;ABORTING- BAD BSF READ
8800      JMP          HLTPRG          ;HALT- CAN'T PROCEED
8801      MOV          #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
8802      MOV          R3,P.SECT(R5) ;SET STARTING SECTOR NO.
8803      MOV          #22,R3          ;SET 22 SECTOR LIMIT
8804      TSTB          FORMAT          ;SEE IF 22 SECTOR FORMAT
8805      BEQ          14$          ;BR IF YES
8806      MOV          #19,R3          ;SET 20 SECTOR LIMIT
8807      MOV          #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
8808      CLRB          WCEFLG          ;INIT THE ERROR FLAG
8809

```

```

8810 041324 004737 042542      JSR      PC,DRVCAL      ;READ THE SOFTWARE BSF
8811 041330 105737 003140      TSTB     WCEFLG      ;SEE IF ANY ERRORS
8812 041334 001407              BEQ       16$          ;BR IF NOT
8813 041336 062765 000002 000004 ADD      #2,P,SECT(R5) ;CHECK NEXT SECTOR
8814 041344 126503 000004      CMPB     P,SECT(R5),R3 ;SEE IF LIMIT EXCEEDED YET
8815 041350 003760              BLE      14$          ;BR IF NOT YET
8816 041352 000736              BR       10$          ;REPORT ERROR AND ABORT
8817 041354 012737 044200 003046 16$: MOV      #ERRHDL,A,ABNL ;RESTORE ERROR HANDLER ADRS
8818 041362 000207              RTS       PC          ;RETURN
8819
8820 ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
8821 ;*POSSIBLE ERRORS IN READING BAD SECTORS.
8822 041364 032765 130600 000034 BDSCHD: BIT      #BSE!HVR!DTE!OPI!DCK,P,ER(R5) ;SEE IF ANY READ ERRORS
8823 041372 001002              BNE      4$           ;BR IF A READ ERROR OCCURRED
8824 041374 000137 044200      JMP      ERRHDL      ;GO HANDLE OTHER ERROR
8825 041400 105237 003140      4$: INCB     WCEFLG      ;SET ERROR FLAG
8826 041404 000137 046356      JMP      RETNML     ;TAKE NORMAL DRIVER RETURN
8827
8828
8829
8830 ;*****
8831 ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
8832 ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
8833 ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
8834 ;*2 BSF'S.
8835 ;*****
8836 041410 104407              TYPBSF: SAVREG      ;SAVE R0-R5
8837 041412 005004              CLR      R4          ;INIT BSF FILE INDICATOR
8838 041414 012700 004234      MOV      #BSFACT+10,R0 ;ADRS OF DATA IN FACTORY BSF
8839 041420 104401 011164      TYPE     ,FACTBS    ;TYPE "FACTORY"
8840 041424 104401 011211      TYPE     ,BDSECT    ;TYPE "BAD SECTORS"
8841 041430 104401 063743 3$: TYPE     ,DH6041    ;TYPE "CYLNDR TRACK SECTOR"
8842 041434 104401 001315      TYPE     ,SCRLF
8843 041440 005001              CLR      R1          ;INIT LIMIT COUNTER
8844 041442 005710 4$: TST      (R0)          ;SEE IF A SECTOR LIS ED HERE
8845 041444 100007              BPL      8$          ;BR IF YES
8846 041446 005701              TST      R1          ;SEE IF FILE IS EMPTY
8847 041450 001032              BNE      14$          ;BR IF NOT EMPTY
8848 041452 104401 011231      TYPE     ,NOFALS    ;TYPE "NONE"
8849 041456 104401 001315      TYPE     ,SCRLF      ;TYPE <CR>,<LF>
8850 041462 000425              BR       14$
8851 041464 012046 8$: MOV      (R0)+,-(SP) ;GET A CYL NO.
8852 041466 104402              TYPOC     ;TYPE IT
8853 041470 104401 013313      TYPE     ,SPACE2    ;TYPE SEPARATORS
8854 041474 011003              MOV      (R0),R3    ;GET TRACK AND SECTOR
8855 041476 105003              CLRB     R3          ;GET THE TRACK NO.
8856 041500 000303              SWAB     R3
8857 041502 010346              MOV      R3,-(SP)
8858 041504 104402              TYPOC     ;TYPE IT
8859 041506 104401 013313      TYPE     ,SPACE2    ;TYPE SEPARATORS
8860 041512 111003              MOV      (R0),R3    ;GET SECTOR NO.
8861 041514 010346              MOV      R3,-(SP)
8862 041516 104402              TYPOC     ;TYPE IT
8863 041520 104401 001315      TYPE     ,SCRLF
8864 041524 005720              TST      (R0)+      ;INCR THE POINTER
8865 041526 005201              INC       R1          ;INCR THE COUNTER

```

8866 041530 020127 000176
8867 041534 002742
8868 041536 005704
8869 041540 001006
8870 041542 005204
8871 041544 012700 003234
8872 041550 104401 011177
8873 041554 000725
8874 041556 104410
8875 041560 000207

14\$: CMP R1, #126. ; SEE IF LIMIT REACHED IN THIS FILE
BLT 4\$; BR IF NOT YET
TST R4 ; SEE IF BOTH FILES TYPED YET
BNE 18\$; BR IF YES
INC R4 ; SET INDICATOR FOR SOFT. FILE
MOV #BSSOFT+10, R0 ; ADRS OF DATA IN SOFTWARE BSF
TYPE SOFTBS ; TYPE "SOFTWARE BAD SECTORS :"
BR 3\$; GO TYPE SOFT. BSF
18\$: RESREG ; RESTORE RO-R5
RTS PC ; RETURN

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

8883 041562 104407
8884 041564 016500 000002
8885 041570 116501 000005
8886 041574 116502 000004
8887 041600 005003
8888 041602 026500 000030
8889 041606 001006
8890 041610 126501 000027
8891 041614 001003
8892 041616 126502 000026
8893 041622 001404
8894 041624 005203
8895 041626 004737 041646
8896 041632 000763
8897 041634 000303
8898 041636 010337 005564
8899 041642 104410
8900 041644 000207

FINMEM: SAVREG ; SAVE RO-R5
MOV P.CYLN(R5), R0 ; GET ORIG. CYL NO.
MOVB P.TRCK(R5), R1 ; GET TRACK NO.
MOVB P.SECT(R5), R2 ; SECTOR NO.
CLR R3 ; INIT SECTOR COUNT TO 0
6\$: CMP P.DCYL(R5), R0 ; COMPARE ORIG. AND CURRENT CYLS
BNE 8\$; BR IF NOT EQUAL
CMPB P.DTS+1(R5), R1 ; COMPARE TRACKS
BNE 8\$; 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 6\$; GO COMPARE ADDRESSES
12\$: SWAB R3 ; COMPUTE NO. OF WORDS XFERRED
MOV R3, WDSXFR ; STORE IT
RESREG ; RESTORE RO-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.

8909 041646 012704 000025
8910 041652 105737 003125
8911 041656 001402
8912 041660 012704 000023
8913 041664 005202
8914 041666 020204
8915 041670 003407
8916 041672 005002
8917 041674 005201
8918 041676 020127 000002
8919 041702 003402
8920 041704 005001
8921 041706 005200

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.

8922 041710 000207

6\$: RTS PC ;RETURN

8923
8924
8925
8926
8927
8928
8929
8930
8931
8932 041712 104407
8933 041714 004737 041562
8934 041720 016500 000030
8935 041724 116501 000027
8936 041730 116502 000026
8937 041734 004737 041646
8938 041740 010065 000002
8939 041744 110165 000005
8940 041750 110265 000004
8941 041754 013703 005564
8942 041760 062703 000400
8943 041764 060365 000012
8944 041770 005004
8945 041772 006103
8946 041774 006104
8947 041776 060337 005604
8948 042002 005537 005606
8949 042006 060437 005606
8950 042012 013765 005604 000010
8951 042020 013700 005606
8952 042024 042700 177774
8953 042030 150065 000007
8954 042034 005737 057732
8955 042040 100002
8956 042042 004737 031646
8957 042046 104410
8958 042050 000207

```

*****
MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
*THIS SUBROUTINE UPDATES PMA,PMA+2 MEM MGT REGS UNIBUS MAP P.BALO(R5),
P.BAHI(R5), AND P.WC(R5) IN PREPARATION TO RESUME A TRANSFER
*TERMINATED BY A BAD SECTOR ERROR (BSE).
*****
MIDXFR: SAVREG          ;SAVE R0-R5
        JSR          PC,FINMEM      ;COMPUTE NO. OF WORDS XFERRED
        MOV          P.DCYL(R5),R0  ;GET CYL NO.
        MOV          P.DTS+1(R5),R1 ;GET TRACK NO.
        MOV          P.DTS(R5),R2   ;GET SECTOR NO.
        JSR          PC,INCRSC      ;INCREMENT PACK ADRS TO NEXT SECTOR
        MOV          R0,P.CYLN(R5)  ;UPDATE CYLINDER
        MOV          R1,P.TRCK(R5)  ;UPDATE TRACK
        MOV          R2,P.SECT(R5)  ;UPDATE SECTOR
        MOV          WDSXFR,R3      ;GET NO. OF WORDS XFERRED
        ADD          #400,R3        ;SKIP BAD SECTOR
        ADD          R3,P.WC(R5)    ;UPDATE P.WC(R5)
        CLR          R4             ;GET BYTES XFERRED
        ROL          R3
        ROL          R4
        ADD          R3,PMA         ;UPDATE PMA,PMA+2
        ADC          PMA+2
        ADD          R4,PMA+2
        MOV          PMA,P.BALO(R5) ;UPDATE P.BALO(R5)
        MOV          PMA+2,R0
        BIC          #177774,R0
        BISB         R0,P.BAHI(R5) ;UPDATE P.BAHI(R5)
        TST          $K11          ;SEE IF MEM MGT
        BPL          16$
        JSR          PC,PREPAR      ;UPDATE MEM MGT AND UNIBUS MAP
        RESREG         ;RESTORE R0-R5
        RTS            PC           ;RETURN

```

8959
8960
8961
8962
8963
8964
8965
8966 042052 104407
8967 042054 012700 002632
8968 042060 012701 005552
8969 042064 016537 000012 005564
8970 042072 005437 005564
8971 042076 012737 065660 005604
8972 042104 005037 005606
8973 042110 005737 005536
8974 042114 001414
8975 042116 013737 005600 005604
8976 042124 013737 005602 005606
8977 042132 000405

```

*****
SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
*****
SVPRMS: SAVREG          ;SAVE R0-R5
        MOV          #PARMO+2,R0    ;ADRS OF PARAMS
        MOV          #SAVPRS,R1     ;ADRS OF SAVE AREA
        MOV          P.WC(R5),WDSXFR ;GET WORD COUNT
        NEG          WDSXFR         ;MAKE IT POSITIVE
        MOV          #RWBUF,PMA     ;INIT PMA TO RWBUF
        CLR          PMA+2          ;INIT PMA+2 TO 0
        TST          PATRN          ;SEE IF DEFAULT DATA TEST
        BEQ          GTO            ;BR IF YES
        MOV          MA,PMA         ;SET PMA=MA
        MOV          MA+2,PMA+2     ;SET PMA+2=MA+2
        BR           GTO

```

H14

8978 042134 104407
8979 042136 012700 005552
8980 042142 012701 002632
8981 042146 012702 000005
8982 042152 012021
8983 042154 005302
8984 042156 001375
8985 042160 104410
8986 042162 000207

GTPRMS: SAVREG
MOV #SAVPRS,R0 ;SAVE R0-R5
MOV #PARMO+2,R1 ;ADRS OF SAVE AREA
GTO: MOV #5,R2 ;ADRS OF PARAMS
6\$: MOV (R0)+,(R1)+ ;SET FOR 5 WORDS
DEC R2 ;MOVE A WORD
BNE 6\$;SEE IF DONE YET
RESREG ;BR IF NOT YET
RTS PC ;RESTORE R0-R5
;RETURN

8989
8990
8991
8992
8993
8994
8995 042164 010446
8996 042166 005004
8997 042170 105037 003140
8998 042174 004737 042542
8999 042200 032737 000100 005474
9000 042206 001403
9001 042210 112737 000001 003140
9002 042216 032737 000002 005474
9003 042224 001406
9004 042226 005204
9005 042230 004737 041712
9006 042234 005765 000012
9007 042240 001355
9008 042242 005704
9009 042244 001411
9010 042246 004737 042134
9011 042252 004737 042052
9012 042256 005737 057732
9013 042262 100002
9014 042264 004737 031646
9015 042270 012604
9016 042272 000207

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

TRANSFR: MOV R4,-(SP) ;SAVE R4 ON STACK
CLR R4 ;CLEAR BSE ERROR INDICATOR
CLRB WCEFLG ;INIT WCE ERROR FLAG TO 0
4\$: JSR PC,DRVCAL ;PERFORM THE S.S. FUNCTION
BIT #WCERR,RECODE ;SEE IF WCE ERROR OCCURRED
BEQ 5\$;BR IF NOT
MOVB #1,WCEFLG ;SET WCE ERROR FLAG
5\$: BIT #BSEERR,RECODE ;SEE IF BSE ERROR OCCURRED
BEQ 6\$;BR IF NOT
INC R4 ;INCR BSE ERROR INDICATOR
JSR PC,MIDXFR ;UPDATE PARAMS TO RESUME XFER
TST P,WC(R5) ;SEE IF ENTIRE XFER IS COMPLETED
BNE 4\$;BR IF NOT
6\$: TST R4 ;SEE IF ANY BSE ERRORS OCCURRED
BEQ 8\$;BR IF NOT
JSR PC,GTPRMS ;RESTORE ORIGINAL PARAMS OF XFER
JSR PC,SVPRMS ;RESTORE WDS,FR,PMA,PMA+2
TST \$K11 ;SEE IF MEM MGT PRESENT
BPL 8\$;BR IF NOT
JSR PC,PREPAR ;PREPARE MEM MGT AND U.M.
8\$: MOV (SP)+,R4 ;RESTORE R4
RTS PC ;RETURN

9017
9018
9019
9020
9021
9022
9023
9024
9025
9026
9027
9028
9029
9030
9031 042274 010237 001266
9032 042300 010337 001270
9033 042304 012637 001272

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

SRHTBS: MOV R2,\$TMP2
MOV R3,\$TMP3
MOV (SP)+,\$TMP4 ;STORE RETURN CONTENTS OF R4

```

9034 042310 011402      MOV      (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
9035 042312 012437 001264 MOV      (R4)+,$TMP1 ;SETUP FOR LENGTH OF BAD SECTOR TABLE
9036 042316 062737 001000 001264 ADD      #1000,$TMP1
9037 042324 005003      CLR      R3 ;CLEAR R3 FOR COUNT
9038 042326 062702 000010 1$: ADD      #10,R2 ;SET R2 FOR POINTER TO CYLINDER ENTRY
9039 042332 005712      TST      (R2) ;TEST IF ALL ONES
9040 042334 100430      BMI      5$ ;YES-DONE
9041 042336 020012      CMP      R0,(R2) ;TEST IF BAD SECTOR IN PRESENT CYL
9042 042340 001406      BEQ      3$ ;YES-GO CHECK TRACK
9043 042342 062702 000004      ADD      #4,R2 ;ELSE BUMP POINTER
9044 042346 020237 001264      CMP      R2,$TMP1 ;TEST IF OUT OF TABLE
9045 042352 002021      BGE      5$ ;YES-EXIT
9046 042354 000766      BR      1$ ;LOOP
9047 042356 005722 3$: TST      (R2)+ ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
9048 042360 011237 001302      MOV      (R2),$TMP10 ;GET TRK/SEC WORD
9049 042364 042737 174377 001302 BIC      #174377,$TMP10 ;CLEAR ALL BUT TRACK
9050 042372 123701 001303      CMPB     $TMP10+1,R1 ;CHECK IF BAD SECTOR IN THIS TRACK
9051 042376 001402      BEQ      4$ ;YES - GO PUT SECTOR NUMBER ON STACK
9052 042400 005722      TST      (R2)+ ;ELSE BUMP POINTER TO NEXT CYL WORD
9053 042402 000753      BR      1$ ;GO TEST NEXT CYL
9054 042404 005203 4$: INC      R3 ;BUMP BAD SECTOR COUNT
9055 042406 012246      MOV      (R2)+,-(SP) ;PUT TRK/SEC WORD ON STACK
9056 042410 042716 177700      BIC      #177700,(SP) ;CLEAR ALL BUT SECTOR NUMBER
9057 042414 000746      BR      1$ ;GO CHECK REST OF FILE
9058 042416 010346 5$: MOV      R3,-(SP) ;EXIT - PUT NUMBER OF BAD
9059 042420 013702 001266      MOV      $TMP2,R2 ;SECTORS ON STACK-RESTORE
9060 042424 013703 001270      MOV      $TMP3,R3 ;REGISTERS
9061 042430 013746 001272      MOV      $TMP4,-(SP) ;PUT RETURN ON STACK
9062 042434 000204      RTS      R4 ;RETURN
9063 *****
9064 ;SHTTL BAD SECTOR CHECK ROUTINE
9065 ;THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
9066 ;SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
9067 ;TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
9068 ;IS NOT LISTED THE FLAG IS RESET.
9069 *****
9070 BDSRCK: SAVREG
9071 042440 012737 004224 042452 MOV      #BSFACT,1$ ;SET TABLE TO SEARCH
9072 042446 004437 042274 2$: JSR      R4,SRHTBS ;GO SEARCH IT
9073 042452 000000 1$: .WORD ;TABLE ADDRESS GOES HERE
9074 042454 012603      MOV      (SP)+,R3 ;GET NUMBER OF BAD SECTORS, IF ANY
9075 042456 001015      BNE      6$ ;IF ANY, GO TEST WHICH ONES
9076 042460 023727 042452 003224 7$: CMP      1$,#BSSOFT ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
9077 042466 001404      BEQ      3$ ;IF YES-EXIT
9078 042470 012737 003224 042452 MOV      #BSSOFT,1$ ;SET OTHER TABLE FOR SEARCH
9079 042476 000763      BR      2$ ;GO SEARCH IT
9080 042500 042737 001000 005474 3$: BIC      #BADSEC,RECODE ;CLEAR BAD SECTOR BIT
9081 042506 104410 4$: RESREG
9082 042510 000207      RTS      PC ;RETURN
9083 042512 022602 6$: CMP      (SP)+,R2 ;THIS SECTOR IN TABLE?
9084 042514 001403      BEQ      8$ ;YES-GO SET BIT & EXIT
9085 042516 005303      DEC      R3 ;DECREMENT BAD SECTOR COUNT
9086 042520 001374      BNE      6$ ;IF NOT ZERO-CHECK NEXT ENTRY
9087 042522 000756      BR      7$ ;ELSE GO SEARCH OTHER TABLE
9088 042524 052737 001000 005474 8$: BIS      #BADSEC,RECODE ;SET BAD SECTOR BIT
9089 042532 005303 9$: DEC      R3 ;CLEAR STACK OF OTHER BAD SECTOR

```

9090 042534 001764
9091 042536 005726
9092 042540 000774

BEQ 4\$;NUMBER
TST (SP)+
BR 9\$;EXIT

9093
9094
9095
9096
9097
9098
9099
9100
9101
9102
9103
9104
9105
9106
9107
9108
9109
9110
9111
9112
9113
9114
9115
9116
9117
9118
9119
9120
9121
9122
9123
9124
9125
9126
9127
9128
9129
9130
9131
9132
9133
9134
9135
9136
9137
9138
9139
9140
9141
9142
9143
9144
9145

042542
042542 104407
042544 004737 035604
042550 004737 037404
042554 105037 003123
042560 105037 003136
042564 004737 042622

042570 000240
042572 010537 042602
042576 004737 052604
042602 000000
042604 004737 047154
042610 105737 003123
042614 001773
042616 104410
042620 000207

```

*****
;SBTTL CALL DRIVER ROUTINE
;ENTRY JSR PC,DRVCAL
;      WITH R5 POINTING TO PARAMETER BLOCK
;RETURN RTS PC
;
;THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
;CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
;BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
;BLOCK WHEN THE ROUTINE IS CALLED.
;
;THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
;WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
;SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
;INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
;FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
*****

```

```

DRVCAL:
    SAVREG                ;SAVE R0-R5
    JSR PC,STALL          ;PERFORM A STALL IF REQUIRED
    JSR PC,CTLOUT         ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
    CLRB DONE            ;CLEAR DONE FLAG
    CLRB DRNAFG           ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
    PC,STRCMD             ;STORE PREV AND CURRENT COMMANDS
    BICB #B.CDT,P.CS1H(R5) ;CLEAR THE RK07 DRIVE TYPE BIT
    TSTB TYPFMT           ;CURRENT DRIVE IS AN RK07 ?
    BEQ 9$                ;BRANCH IF NOT
    BISB #B.CDT,P.CS1H(R5) ;SET THE RK07 DRIVE TYPE BIT
9$:
    NOP
    MOV R5,4$            ;GET PARAM BLOCK ADDRESS
    JSR PC,C.INIT        ;CALL DRIVER
4$:
    WORD P.B. ADDRESS GOES HERE
6$:
    JSR PC,W.WTCH        ;CALL WATCH DOG
    TSTB DONE            ;DONE SET?
    BEQ 6$               ;NO-LOOP
12$:
    RESREG               ;RESTORE R0-R5
    RTS PC               ;YES-RETURN

```

```

*****
;STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
*****
STRCMD: SAVREG                ;SAVE R0-R5
        MOV #PRVCMD,R1        ;ADDR OF PREV COMMAND STORAGE
        MOV #COMSTR,R0        ;ADDR OF CURRENT COMMAND STORAGE
;STORE PREVIOUS COMMAND
        MOV (R0)+,(R1)+
        MOV (R0)+,(R1)+
        MOV (R0)+,(R1)+

```

9146 042642 012021
9147 042644 012021
9148 042646 012021
9149 042650 105737 003143
9150 042654 001414
9151 042656 013737 005262 005256
9152 042664 013737 005260 005254
9153 042672 013737 170202 005262
9154 042700 013737 170200 005260
9155
9156 042706 012701 005240
9157 042712 012521
9158 042714 012521
9159 042716 012521
9160 042720 012521
9161 042722 012521
9162 042724 012521
9163 042726 104410
9164 042730 000207
9165
9166
9167
9168
9169
9170
9171
9172
9173
9174
9175 042732 152737 000377 003123
9176 042740 032737 100000 005474
9177 042746 001403
9178 042750 105037 003126
9179 042754 000413
9180 042756 105737 003126
9181 042762 001410
9182 042764 005037 001174
9183 042770 113737 003126 001174
9184 042776 104101
9185 043000 105037 003126
9186 043004 005037 005474
9187 043010 000207
9188
9189
9190
9191
9192
9193
9194
9195
9196
9197
9198 043012 104407
9199 043014 105237 003130
9200 043020 032737 000040 005702
9201

```
MOV (R0)+, (R1)+  
MOV (R0)+, (R1)+  
MOV (R0)+, (P1)+  
TSTB UBMPRS ;SEE IF UNIBUS MAP PRESENT  
BEQ 4$ ;BR IF NOT  
MOV CRMPHO, PRMPHO ;STORE PREV U.B. MAP REG 0  
MOV CRMPLO, PRMPLO  
MOV 2#MAPH00, CRMPHO ;STORE CURRENT U.B. MAP REG 0  
MOV 2#MAPL00, CRMPLO  
;STORE CURRENT COMMAND  
4$: MOV #COMSTR, R1  
MOV (R5)+, (R1)+ ;R5 POINTS TO DRIVER PARAM BLK  
MOV (R5)+, (R1)+  
MOV (R5)+, (R1)+  
MOV (R5)+, (R1)+  
MOV (R5)+, (R1)+  
MOV (R5)+, (R1)+  
RESREG ;RESTORE R0-R5  
RTS PC ;RETURN  
  
;*****  
;SBTTL DRIVE ERROR FREE RETURN ROUTINE  
;THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR  
;HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE  
;DONE FLAG THAT IS TESTED IN THE DRIVER CALLING  
;ROUTINE.  
;*****  
ERRFRE: BISB #377, DONE ;SET THE DONE FLAG  
BIT #ANYDER, RECODE ;TEST IF ANY DATA ERROR  
BEQ 2$ ;IF NO - DO ERROR RECOVERY PRINT TEST  
CLRB ERRCNT ;CLEAR ERROR COUNT  
BR 1$ ;EXIT  
2$: TSTB ERRCNT ;CHECK IF ANY ERRORS HAVE OCCURRED  
BEQ 1$ ;NO - SKIP TO EXIT  
CLR $REG5  
MOVB ERRCNT, $REG5 ;GET RETRY COUNT  
ERROR 101 ;PRINT RETRY SUCCESSFUL MESSAGE  
CLRB ERRCNT ;CLEAR ERROR COUNT  
1$: CLR RECODE ;CLEAR RECOVERY FLAGS  
RTS PC ;RETURN  
;*****  
;SBTTL TYPE ERROR ROUTINE  
;ENTRY - JSR PC, TYPERR  
;RETURN - RTS PC  
;  
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH  
;ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)  
;ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING  
;THE ERROR.  
;*****  
TYPERR: SAVREG  
INCB DRVERS ;INCR ERROR COUNT FOR THIS DRIVE  
BIT #BITS, CS ;SEE IF DRIVE SHOULD BE DROPPED  
; IF ERROR LIMIT EXCEEDED
```

9202	043026	001412		BEQ	9\$;BR IF NOT
9203	043030	123727	003130	CMPB	DRVERS, #20.	;SEE IF 20(DEC) ERRORS EXCEEDED
9204	043036	002406		BLT	9\$;BR IF NOT
9205	043040	105037	003130	CLRB	DRVERS	;CLEAR DRIVE ERROR COUNT
9206	043044	104401	011051	TYPE	, DROPDR	;TYPE "DROPPING DRIVE"
9207	043050	000137	017332	JMP	NEWDRV	;PROCEED TO TEST NEXT DRIVE
9208	043054	032777	020000	BIT	#SW13, #SWR	;INHIBIT ERROR TYPEOUTS?
9209	043062	001402		BEQ	6\$;BR IF NO
9210	043064	000137	043470	JMP	20\$	
9211	043070	005000		CLR	RO	;CLR RO FOR ERROR NUMBER
9212	043072	005005		CLR	R5	;INIT INDENT INDICATOR
9213	043074	005105		COM	R5	
9214	043076	113700	001114	MOVB	\$ITEMB, RO	;ENTER ERROR NUMBER
9215	043102	005300		DEC	RO	;FORM INDEX FOR ERROR TABLE
9216	043104	006300		RO	RO	
9217	043106	006300		ASL	RO	
9218	043110	006300		ASL	RO	
9219	043112	062700	001400	ADD	#ERRTB, RO	;FORM ADDRESS OF ERROR ENTRY
9220	043116	012037	043136	MOV	(RO)+, 2\$;GET EM POINTER
9221	043122	001406		BEQ	3\$;BRANCH IF THERE ISN'T ONE
9222	043124	104401	013277	TYPE	, CR2LF	
9223	043130	104401	060370	TYPE	, AS2SP2	;TYPE "*** "
9224	043134	104401		TYPE		;TYPE ERROR MESSAGE (EM)
9225	043136	000000		.WORD	0	;EM POINTER GOES HERE
9226	043140	012037	043314	MOV	(RO)+, 4\$;GET OH POINTER
9227	043144	001467		BEQ	5\$;BR IF THERE ISN'T ONE
9228	043146	104401	001315	TYPE	, \$CRLF	
9229	043152	104401	060356	TYPE	, TSTMSG	;TYPE " TEST "
9230	043156	013746	001102	MOV	\$TSTNM, -(SP)	;GET TEST NO. ON STACK
9231	043162	104403		TYPOS		;TYPE IT
9232	043164	002		.BYTE	2	;2 DIGITS
9233	043165	000		.BYTE	0	;SUPPRESS LEADING ZEROS
9234	043166	104401	013277	TYPE	, CR2LF	
9235	043172	032777	010000	BIT	#BIT12, #SWR	;REPORT DESCRIPTION ONLY ?
9236	043200	001133		BNE	20\$;BR IF YES
9237	043202	104401	063110	TYPE	, DH105	;TYPE "PREVIOUS COMMAND :"
9238	043206	104401	001315	TYPE	, \$CRLF	
9239	043212	104401	063253	TYPE	, DH101+10	;TYPE PREV COMMAND HEADER
9240	043216	104401	001315	TYPE	, \$CRLF	
9241	043222	012701	000006	MOV	#6, R1	;SIX COMMAND VALUES
9242	043226	012702	001236	MOV	#\$REG26, R2	;STARTING ADDR OF PREV CMND VALUES
9243	043232	012246		MOV	(R2)+, -(SP)	;PUT A WORD ON STACK
9244	043234	104402		TYPOC		;TYPE IT
9245	043236	104401	013313	TYPE	, SPACE2	;TYPE SEPARATORS
9246	043242	005301		DEC	R1	;SEE IF 7 VALUES TYPED YET
9247	043244	001372		BNE	30\$;BR IF NOT
9248	043246	104401	001315	TYPE	, \$CRLF	
9249	043252	104401	013313	TYPE	, SPACE2	;INDENT
9250	043256	104401	063332	TYPE	, DH102	;TYPE HDR FOR BA DATA
9251	043262	104401	001315	TYPE	, \$CRLF	
9252	043266	104401	013313	TYPE	, SPACE2	;INDENT
9253	043272	012246		MOV	(R2)+, -(SP)	
9254	043274	104402		TYPOC		;TYPE PREV. HI BA BITS
9255	043276	104401	013313	TYPE	, SPACE2	
9256	043302	011246		MOV	(R2), -(SP)	
9257	043304	104402		TYPOC		;TYPE PREV. LO BA BITS

```

9258 043306 104401 013277      TYPE      ,CR2LF
9259 043312 104401      TYPE
9260 043314 000000      4$:      .WORD      0      ;TYPE DATA HEADER
9261 043316 104401 001315      TYPE      $CRLF      ;DH POINTER GOES HERE
9262 043322 005005      CLR      R5      ;INIT INDENT INDICATOR
9263 043324 032777 010000 135606 5$:      BIT      #BIT12,2SWR      ;REPORT DESCRIPTION ONLY ?
9264 043332 001056      BNE      20$      ;BR IF YES
9265 043334 012001      MOV      (R0)+,R1      ;GET DT POINTER
9266 043336 001454      BEQ      20$      ;BRANCH IF THERE ARE NONE
9267 043340 012000      MOV      (R0)+,R0      ;GET DF POINTER
9268 043342 012002      MOV      (R0)+,R2      ;STORE NUMBER OF DH'S
9269 043344 112003      10$:      MOV8      (R0)+,R3      ;GET & STORE NUMBER OF DATA WORDS
9270 043346 105720      TSTB      (R0)+      ;BUMP PAST FORMAT WORD
9271 043350 005703      TST      R3      ;TEST IF ANY DATA FOR THIS HEADER
9272 043352 001417      BEQ      14$      ;NO - SKIP DATA PRINT
9273 043354 005705      TST      R5      ;SEE IF SHOULD INDENT
9274 043356 001002      BNE      11$      ;BR IF NOT
9275 043360 104401 013313      TYPE      SPACE2      ;INDENT
9276 043364 013146      11$:      MOV      2(R1)+,-(SP)      ;PUT FIRST DATA WORD ON STACK
9277 043366 104402      TYP0C
9278 043370 005303      DEC      R3      ;TYPE IT
9279 043372 001403      BEQ      12$      ;MORE DATA WORDS
9280 043374 104401 013313      TYPE      SPACE2      ;NO-BRANCH
9281 043400 000771      BR      11$      ;TYPE SEPARATORS
9282 043402 005702      12$:      TST      R2      ;LOOP
9283 043404 001402      BEQ      14$      ;SEE IF <CR>,<LF> NEEDED
9284 043406 104401 001315      TYPE      $CRLF      ;BR IF NOT
9285 043412 005302      14$:      DEC      R2      ;TYPE IT
9286 043414 003425      BLE      20$      ;MORE DH'S?
9287 043416 012037 043450      15$:      MOV      (R0)+,16$      ;NO-BRANCH
9288 043422 105710      TSTB      (R0)      ;GET NEXT DH POINTER
9289 043424 001004      BNE      34$      ;SEE IF ANY DATA FOR HDR
9290 043426 104401 001315      TYPE      $CRLF      ;BR IF YES
9291 043432 005005      CLR      R5      ;SKIP EXTRA LINE
9292 043434 000404      BR      36$      ;RE-INIT INDENT INDICATOR
9293 043436 005105      34$:      COM      R5      ;COMPLEMENT INDENT INDICATOR
9294 043440 001002      BNE      36$      ;BR IF NO INDENT REQUIRED
9295 043442 104401 013313      TYPE      SPACE2      ;INDENT
9296 043446 104401      36$:      TYPE
9297 043450 000000      16$:      .WORD      0      ;TYPE DH
9298 043452 104401 001315      TYPE      $CRLF      ;DH POINTER GOES HERE
9299 043456 105710      TSTB      (R0)      ;TYPE A DT?
9300 043460 001331      BNE      10$      ;YES-BRANCH
9301 043462 062700 000002      ADD      #2,R0      ;INCREMENT OF POINTER
9302 043466 000751      BR      14$      ;SEE IF END OF DF BLOCK
9303 043470 104410      20$:      RESREG
9304 043472 000207      RTS      PC
9305
9306 ;*****
9307 ;SBTTL CONTROLLER ERROR REPORTER ROUTINE
9308 ;*ENTRY:      JSR PC, CONERR
9309 ;*RETURN:      RTS PC
9310 ;*
9311 ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
9312 ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
9313 ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID

```

```
9314 ;*AT THIS TIME.
9315 ;*****
9316 043474 104407 CONERR: SAVREG ;SAVE R0-R5
9317 043476 152737 000377 003123 BISB #377,DONE ;SET DONE FLAG
9318 043504 105237 003126 INCB ERRCNT ;INCREMENT ERROR COUNT
9319 043510 004737 046366 JSR PC, TOPROC ;LOAD RK REGS INTO $REGS
9320 043514 032737 000001 003052 BIT #BIT0,E.CONT ;ERROR 0?
9321 043522 001402 BEQ 1$ ;NO-BRANCH
9322 043524 104064 ERROR 64 ;CLEAR CONT DID NOT CLEAR ERROR
9323 043526 000470 BR 7$
9324 043530 032737 000002 003052 1$: BIT #BIT1,E.CONT ;ERROR 1?
9325 043536 001402 BEQ 2$ ;NO-BRANCH
9326 043540 104065 ERROR 65 ;NO ATTENTION IN ATTENTION SUM REG
9327 043542 000462 BR 7$
9328 043544 032737 000004 003052 2$: BIT #BIT2,E.CONT ;ERROR 2?
9329 043552 001407 BEQ 3$ ;NO-BRANCH
9330 043554 105737 003136 TSTB DRNAFG ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
9331 043560 001402 BEQ 15$ ;BR IF NOT
9332 043562 105237 003137 INCB REISSU ;SET FLAG TO RE-ISSUE COMMAND
9333 043566 104066 15$: ERROR 66 ;UNSOLICITED ATTENTION
9334 043570 000447 BR 7$
9335 043572 032737 000010 003052 3$: BIT #BIT3,E.CONT ;ERROR 3?
9336 043600 001402 BEQ 4$ ;NO-BRANCH
9337 043602 104067 ERROR 67 ;UNEXPECTED DATA TYPE ERROR
9338 043604 000441 BR 7$
9339 043606 032737 000020 003052 4$: BIT #BIT4,E.CONT ;ERROR 4?
9340 043614 001402 BEQ 5$ ;NO-BRANCH
9341 043616 104070 ERROR 70 ;ATTENTION DID NOT RESET WITH CLEAR
9342 043620 000433 BR 7$
9343 043622 032737 000040 003052 5$: BIT #BIT5,E.CONT ;ERROR 5?
9344 043630 001402 BEQ 6$ ;NO-BRANCH
9345 043632 104071 ERROR 71 ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
9346 043634 000425 BR 7$
9347 043636 032737 000400 003052 6$: BIT #BIT8,E.CONT
9348 043644 001401 BEQ 8$
9349 043646 104072 ERROR 72 ;DATA LATE WHEN UNLOADING HEADER
9350 043650 032737 001000 003052 8$: BIT #BIT9,E.CONT
9351 043656 001401 BEQ 9$
9352 043660 104073 ERROR 73 ;CONTROLLER ERROR DURING DRIVER SERVICE
9353 043662 032737 002000 003052 9$: BIT #BIT10,E.CONT
9354 043670 001401 BEQ 10$
9355 043672 104074 ERROR 74 ;DRIVE DETECTED PARITY ERROR
9356 043674 032737 100000 003052 10$: BIT #BIT11,E.CONT
9357 043702 001401 BEQ 11$
9358 043704 104052 ERROR 52 ;MULTIPLE DRIVE SELECT
9359 043706 104075 11$: ERROR 75 ;UNDEFINED ERROR
9360 043710 005037 003052 7$: CLR E.CONT ;CLEAR CONTROLLER ERROR WORD
9361 043714 000137 046166 JMP BGNRTY ;GO DO RETRY
9362 ;*****
9363 ;SBTTL REPORT SUPPORT ROUTINE
9364 ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
9365 ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
9366 ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
9367 RCPSUP: SAVREG
9368 043720 104407 CLR ERRCOM ;CLEAR ERROR COMMAND STORE
9369 043722 005037 005476
```



```

9370 043726 116537 000001 005476      MOV      P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
9371 043734 012700 001162      MOV      #SREG0,R0 ;FOR REPORTING
9372 043740 016520 000002      MOV      P.CYLN(R5),(R0)+
9373 043744 116520 000005      MOV      P.TRCK(R5),(R0)+
9374 043750 105020      CLRB      (R0)+
9375 043752 116520 000004      MOV      P.SECT(R5),(R0)+
9376 043756 105020      CLRB      (R0)+
9377 043760 016520 000012      MOV      P.WC(R5),(R0)+
9378 043764 012700 001174      MOV      #SREG5,R0
9379 043770 116503 000007      MOV      P.BAHI(R5),R3
9380 043774 042703 177774      BIC      #177774,R3
9381 044000 010337 001256      MOV      R3,SREG36 ;HI BA BITS
9382 044004 016537 000010 001260      MOV      P.BALO(R5),SREG37 ;LO BA BITS
9383 044012 016520 000016      MOV      P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
9384 044016 016520 000020      MOV      P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
9385 044022 016520 000030      MOV      P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
9386 044026 016520 000026      MOV      P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
9387 044032 016520 000022      MOV      P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
9388                                ;FOR ALL REPORTS (TO BE
9389                                ;DETERMINED LATER) BUT IT IS
9390 044036 016520 000024      MOV      P.BAR(R5),(R0)+ ;STORED ANY WAY.
9391 044042 016520 000032      MOV      P.ASOF(R5),(R0)+
9392 044046 016520 000036      MOV      P.DS(R5),(R0)+
9393 044052 016520 000034      MOV      P.ER(R5),(R0)+
9394 044056 016520 000040      MOV      P.A00(R5),(R0)+
9395 044062 016520 000042      MOV      P.B00(R5),(R0)+
9396 044066 016520 000044      MOV      P.A01(R5),(R0)+
9397 044072 016520 000046      MOV      P.B01(R5),(R0)+
9398 044076 016520 000050      MOV      P.A10(R5),(R0)+
9399 044102 016520 700052      MOV      P.B10(R5),(R0)+
9400 044106 016520 000054      MOV      P.A11(R5),(R0)+
9401 044112 016520 000056      MOV      P.B11(R5),(R0)+
9402                                ;STORE PREVIOUS COMMAND FOR PRINTOUT
9403 044116 012701 001236      MOV      #SREG26,R1 ;ADRS OF PRINT BUF AREA
9404 044122 012700 005224      MOV      #PRVCMD,R0 ;ADRS OF PREV CMND STORAGE
9405 044126 112021      MOV      (R0)+,(R1)+ ;DRIVE NO.
9406 044130 105021      CLRB      (R1)+
9407 044132 112021      MOV      (R0)+,(R1)+ ;COMMAND
9408 044134 105021      CLRB      (R1)+
9409 044136 012021      MOV      (R0)+,(R1)+ ;CYL ADDRESS
9410 044140 116021 000001      MOV      1(R0),(R1)+ ;TRACK
9411 044144 105021      CLRB      (R1)+
9412 044146 111021      MOV      (R0),(R1)+ ;SECTOR
9413 044150 105021      CLRB      (R1)+
9414 044152 016021 000006      MOV      6(R0),(R1)+ ;WORD COUNT
9415 044156 116003 000003      MOV      3(R0),R3 ;HI BA BITS
9416 044162 042703 177774      BIC      #177774,R3
9417 044166 010321      MOV      R3,(R1)+
9418 044170 016011 000004      MOV      4(R0),(R1) ;LO BA BITS
9419 044174 104410      RESREG
9420 044176 000207      RTS      PC
9421                                ;*****
9422                                ;SBTTL REPORT ERROR ROUTINE
9423                                ;* ENTRY      JSR      PC,ERRHDL
9424                                ;*RETURN      RTS      PC
9425                                ;*
9426                                ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 185
REPORT ERROR ROUTINE

SEQ 0184

```

9426 ;*IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
9427 ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
9428 ;:*****
9429
9430 044200 104407 ERRHDL: SAVREG
9431 044202 152737 000377 003123 BISB #377,DONE ;SET DONE FLAG
9432 044210 105237 003126 INCB ERRCNT ;INCREMENT ERROR COUNT
9433 044214 005037 005474 CLR RECODE ;CLEAR RECOVERY CODE WORD
9434 044220 032737 000400 005474 ER2ENT: BIT #LEV2ER,RECODE ;TEST IF 2ND LEVEL ERROR
9435 044226 001402 BEQ 52$ ;NO - SKIP PARAM BLOCK CHANGE
9436 044230 012705 002714 MOV #PARM1,R5 ;ELSE SET R5 TO PARAMETER LOCK 1
9437 044234 012737 046340 003046 52$: MOV #RETANL,A.ABNL ;SET NOW ABNORMAL AND NORMAL RETURN FOR
9438 044242 012737 046356 003044 MOV #RETNML,A.NORM ;DRIVER OPERATIONS IN ERROR PROCESSING
9439 044250 004737 043720 JSR PC,REPSUP ;GO SET UP REGISTERS FOR REPORT
9440 ;NOW BEGIN TESTING THE ERROR
9441 ;BITS. THE SEQUENCE IN
9442 ;WHICH THEY ARE TESTED IS
9443 ;CONSIDERED SIGNIFICANT IN
9444 ;THAT ERRORS OF A MORE
9445 ;CATASTROPHIC NATURE ARE FIRST
9446 ;TESTED.
9447
9448 ;IF AN ERROR IS FOUND SET
9449 ;THAT ERROR IS REPORTED AND
9450 ;THE REPORTING IS TERMINATED.
9451 ;IF ADDITIONAL ERRORS ARE SET,
9452 ;THE RK611 REGISTER PRINTOUTS
9453 ;WILL SHOW THIS BUT THE
9454 ;REGISTER CONTENTS MUST BE
9455 ;MANUALLY DECODED TO LOCATE THE
9456 ;SECOND ERROR
9457 044254 016504 000034 MOV P.ER(R5),R4 ;SET R4 TO ERROR REGISTER
9458 044260 032765 020000 000020 BIT #UPE,P.CS2(R5) ;TEST UPE. IF UES-SET

```

9459	044266	001406			BEQ	1\$;REPORT ERROR
9460	044270	104001			ERROR	1		
9461	044272	052737	000200	005474	BIS	#ABORT,RECODE		
9462	044300	000137	045674		JMP	37\$		
9463	044304	032765	004000	000020	1\$:	BIT	#NEM,P.CS2(R5)	;TEST NON-EXISTANT MEMORY
9464	044312	001406			BEQ	2\$		
9465	044314	104002			ERROR	2		
9466	044316	052737	000200	005474	BIS	#ABORT,RECODE		
9467	044324	000137	045674		JMP	37\$		
9468	044330	032765	010000	000020	2\$:	BIT	#NED,P.CS2(R5)	;TEST NON-EXISTANT DRIVE
9469	044336	001412			BEQ	3\$		
9470	044340	032765	000400	000020	BIT	#UFE,P.CS2(R5)		;TEST IF NED & UFE BOTH SET
9471	044346	001403			BEQ	38\$		
9472	044350	104035			ERROR	35		;NED & UFE BOTH SET ERROR
9473	044352	000137	045674		JMP	37\$		
9474	044356	104003			38\$:	ERROR	3	;NED ONLY
9475	044360	000137	045674		JMP	37\$		
9476	044364	032765	000400	000020	3\$:	BIT	#UFE,P.CS2(R5)	;TEST UNIT FIELD ERROR
9477	044372	001412			BEQ	4\$		
9478	044374	032765	010000	000020	BIT	#NED,P.CS2(R5)		;TEST IF UFE & NED BOTH SET
9479	044402	001403			BEQ	39\$;NO-SKIP
9480	044404	104035			ERROR	35		;REPORT NED & UFE BOTH SET
9481	044406	000137	045674		JMP	37\$		
9482	044412	104004			39\$:	ERROR	4	;REPORT UFE ONLY
9483	044414	000137	045674		JMP	37\$		
9484	044420	032765	000100	000014	4\$:	BIT	#CMDTO,P.PRST(R5)	
9485	044426	001423			BEQ	5\$		
9486	044430	004737	046366		JSR	PC, TOPROC		;GO PROCESS TIMEOUT
9487	044434	032737	002000	005474	BIT	#TWOTOS,RECODE		;2ND TIMEOUT IN TIMEOUT PROC?
9488	044442	001007			BNE	40\$;YES - SKIP TO ERROR 56
9489	044444	032737	000400	005474	BIT	#LEV2ER,RECODE		;TEST IF LEVEL 2 ERROR
9490	044452	001006			BNE	41\$;YES - SKIP TO ERROR 57
9491	044454	104005			ERROR	5		;ELSE MAKE FULL TIMEOUT REPORT
9492	044456	000137	045674		JMP	37\$		
9493	044462	104056			40\$:	ERROR	56	;TWO TIMEOUTS ERROR REPORT
9494	044464	000137	045674		JMP	37\$		
9495	044470	104057			41\$:	ERROR	57	;2ND LEVEL ERROR REPORT
9496	044472	000137	044220		JMP	ER2ENT		;GO BUILD AND MAKE 2ND REPORT
9497	044476	032765	010000	000014	5\$:	BIT	#DRVSZD,P.PRST(R5)	;SEE IF DRIVE SIEZED BY OTHER PORT
9498	044504	001403			BEQ	65\$;OR IF DRIVE IS AVAILABLE
9499	044506	104107			ERROR	107		;DRIVE SIEZED BY OTHER PORT
9500	044510	000137	045674		JMP	37\$		
9501	044514	032765	020000	000016	65\$:	BIT	#SPAR,P.CS1(R5)	;TEST D TO C PARITY ERROR
9502	044522	001406			BEQ	6\$		
9503	044524	104006			ERROR	6		
9504	044526	052737	004000	005474	BIS	#RCLREQ,RECODE		
9505	044534	000137	045674		JMP	37\$		
9506	044540	032704	000010		6\$:	BIT	#DRPAR,R4	;TEST DRIVE DETECTED PARITY ERROR
9507	044544	001406			BEQ	7\$		
9508	044546	104007			ERROR	7		
9509	044550	052737	004000	005474	BIS	#RCLREQ,RECODE		
9510	044556	000137	045674		JMP	37\$		
9511	044562	032765	000010	000036	7\$:	BIT	#ACLO,P.DS(R5)	;TEST AC LOW
9512	044570	001403			BEQ	8\$		
9513	044572	104010			ERROR	10		
9514	044574	000137	045674		JMP	37\$		

9515	044600	032765	000020	000036	8\$:	BIT	#SPDLSS,P.DS(R5)	;TEST SPEED LOSS
9516	044606	001403				BEQ	24\$	
9517	044610	104011				ERROR	11	
9518	044612	000137	045674			JMP	37\$	
9519	044616	032765	004000	000016	24\$:	BIT	#CTO,P.CS1(R5)	;TEST FOR CONTROLLER TIMEOUT
9520	044624	001401				BEQ	25\$	
9521	044626	104027				ERROR	27	
9522	044630	032704	000001		25\$:	BIT	#ILC,R4	;TEST ILLEGAL FUNCTION CODE
9523	044634	001403				BEQ	10\$	
9524	044636	104012				ERROR	12	
9525	044640	000137	045674			JMP	37\$	
9526	044644	032765	002000	000020	10\$:	BIT	#PGE,P.CS2(R5)	;TEST PROGRAMMING ERROR
9527	044652	001403				BEQ	11\$	
9528	044654	104013				ERROR	13	
9529	044656	000137	045674			JMP	37\$	
9530	044662	032704	000004		11\$:	BIT	#ILF,R4	;TEST ILLEGAL DRIVE FUNCTION
9531	044666	001403				BEQ	12\$	
9532	044670	104014				ERROR	14	
9533	044672	000137	045674			JMP	37\$	
9534	044676	032704	000040		12\$:	BIT	#DTYPE,R4	;TEST DRIVE TYPE ERROR
9535	044702	001403				BEQ	13\$	
9536	044704	104015				ERROR	15	
9537	044706	000137	045674			JMP	37\$	
9538	044712	032704	000020		13\$:	BIT	#FMTE,R4	;TEST FORMAT ERROR
9539	044716	001403				BEQ	14\$	
9540	044720	104016				ERROR	16	
9541	044722	000137	045674			JMP	37\$	
9542	044726	032704	004000		14\$:	BIT	#WLE,R4	;TEST WRITE LOCK ERROR
9543	044732	001403				BEQ	15\$	
9544	044734	104017				ERROR	17	
9545	044736	000137	045674			JMP	37\$	
9546	044742	032704	040000		15\$:	BIT	#UNS,R4	;TEST DRIVE UNSAFE
9547	044746	001406				BEQ	16\$	
9548	044750	104020				ERROR	20	
9549	044752	052737	000200	005474		BIS	#ABORT,RECODE	
9550	044760	000137	045774			JMP	ALLTRM	
9551	044764	032704	000002		16\$:	BIT	#SKI,R4	;TEST SEEK INCOMPLETE
9552	044770	001406				BEQ	17\$	
9553	044772	104021				ERROR	21	
9554	044774	052737	004000	005474		BIS	#RCLREQ,RECODE	
9555	045002	000137	045674			JMP	37\$	
9556	045006	032704	001000		17\$:	BIT	#COE,R4	;TEST CYLINDER OVERFLOW
9557	045012	001406				BEQ	18\$	
9558	045014	104022				ERROR	22	
9559	045016	052737	004000	005474		BIS	#RCLREQ,RECODE	
9560	045024	000137	045674			JMP	37\$	
9561	045030	032704	002000		18\$:	BIT	#IDAE,R4	;TEST ILLEGAL CYLINDER
9562	045034	001406				BEQ	19\$	
9563	045036	104023				ERROR	23	
9564	045040	052737	004000	005474		BIS	#RCLREQ,RECODE	
9565	045046	000137	045674			JMP	37\$	
9566	045052	032765	000040	000036	19\$:	BIT	#DROT,P.DS(R5)	;TEST DRIVE OFF TRACK
9567	045060	001406				BEQ	20\$	
9568	045062	104024				ERROR	24	
9569	045064	052737	004000	005474		BIS	#RCLREQ,RECODE	
9570	045072	000137	045674			JMP	37\$	

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 188
REPORT ERROR ROUTINE

SEQ 0187

9571	045076	032704	010000		20\$:	BIT	#DTE,R4		;TEST DRIVE TIMING ERROR
9572	045102	001406				BEQ	21\$		
9573	045104	104025				ERROR	25		
9574	045106	052737	000200	005474		BIS	#ABORT,RECODE		
9575	045114	000137	045674			JMP	37\$		
9576	045120	032755	100000	000020	21\$:	BIT	#DLT,P.CS2(R5)		;TEST DATA LATE
9577	045126	001403				BEQ	22\$		
9578	045130	104026				ERROR	26		
9579	045132	000137	045674			JMP	37\$		
9580	045136	032704	020000		22\$:	BIT	#OPI,R4		;TEST IF OPI ERROR
9581	045142	001457				BEQ	29\$		
9582	045144	052737	000010	005474		BIS	#OPIERR,RECODE		
9583	045152	105737	003121			TSTB	DERCNT		;TEST IF FIRST ERROR
9584	045156	001402				BEQ	50\$		
9585	045160	000137	045674			JMP	37\$;NO - SKIP REPORT
9586	045164	032737	000400	005474	50\$:	BIT	#LEV2ER,RECODE		;TEST IF A SECOND LEVEL 2 ERROR
9587	045172	001403				BEQ	26\$;HAS ALREADY OCCURRED
9588	045174	104054			27\$:	E ROR	54		
9589	045176	000137	045674			JMP	37\$;GET OUT OF ERROR REPORT
9590	045202	004737	047026		26\$:	JSR	PC,BLDEXH		;GO BUILD EXPECTED HEADER
9591	045206	004737	046536			JSR	PC,ROHDO		;GET HEADER OF SECTOR 0
9592	045212	032737	000400	005474		BIT	#LEV2ER,RECODE		;TEST IF ERROR GETTING HDR
9593	045220	001025				BNE	28\$;IF YES-GO MAKE ABBREVIATED REPORT
9594	045222	013702	003036			MOV	RKBAS,R2		;STORE HEADER 0 INTO REGISTERS
9595	045226	042762	000100	000000		BIC	#IE,RKCS1(R2)		;RESET INTERRUPT ENABLE
9596	045234	016237	000024	001202		MOV	RKDB(R2),SREG10		;FOR REPORTING
9597	045242	016237	000024	001204		MOV	RKDB(R2),SREG11		
9598	045250	016237	000024	001206		MOV	RKDB(R2),SREG12		
9599	045256	032762	100000	000000		BIT	#CERR,RKCS1(R2)		;TEST IF ERROR DURING STORAGE
9600	045264	001343				BNE	27\$		
9601	045266	104030				ERROR	30		;MAKE OPI REPORT
9602	045270	000137	045674			JMP	37\$		
9603	045274	104054			28\$:	ERROR	54		
9604	045276	000137	044220			JMP	ER2ENT		;GO MAKE 2ND LEVEL REPORT
9605	045302	032704	000400		29\$:	BIT	#HVRC,R4		;TEST IF HVRC ERROR
9606	045306	001457				BEQ	23\$		
9607	045310	052737	000004	005474		BIS	#HVRCER,RECODE		
9608	045316	105737	003121			TSTB	DERCNT		;TEST IF FIRST ERROR
9609	045322	001402				BEQ	30\$;YES - REPORT
9610	045324	000137	045674			JMP	37\$;JUMP TO RETURN
9611	045330	032737	000400	005474	30\$:	BIT	#LEV2ER,RECODE		;TEST IF A 2ND LEVEL ERROR HAS ALREADY
9612	045336	001403				BEQ	31\$;OCCURRED. NO-SKIP EXIT
9613	045340	104055			51\$:	ERROR	55		
9614	045342	000137	045674			JMP	37\$;GET OUT OF ERROR REPORT
9615	045346	004737	047026		31\$:	JSR	PC,BLDEXH		;GO BUILD EXPECTED HEADER
9616	045352	004737	046536			JSR	PC,ROHDO		;GO GET HDR 0
9617	045356	032737	000400	005474		BIT	#LEV2ER,RECODE		;TEST IF ERROR IN GETTING HDR
9618	045364	001025				BNE	32\$;IF YES-GO MAKE ABBREVIATED REPORT
9619	045366	013702	003036			MOV	RKBAS,R2		;GET RK611 BASE ADDRESS
9620	045372	042762	000100	000000		BIC	#IE,RKCS1(R2)		;CLEAR INTERRUPT ENABLE
9621	045400	016237	000024	001202		MOV	RKDB(R2),SREG10		;STORE OFF HEADER
9622	045406	016237	000024	001204		MOV	RKDB(R2),SREG11		
9623	045414	016237	000024	001206		MOV	RKDB(R2),SREG12		
9624	045422	032762	100000	000000		BIT	#CERR,RKCS1(R2)		;TEST IN ANY ERROR IN UNLOAD
9625	045430	001343				BNE	51\$;IF YES - GO MAKE SHORT REPORT
9626	045432	104031				ERROR	31		;MAKE FULL REPORT

9627	045434	000137	045674		JMP	37\$	
9628	045440	104055		32\$:	ERROR	55	; ABREVIATED HVRC ERROR RPORT
9629	045442	000137	044220		JMP	ER2ENT	; GO REPORT 2ND LEVEL ERROR
9630	045446	032704	000200	23\$:	BIT	#BSE,R4	; TEST FOR BAD SECTOR ERROR
9631	045452	001430			BEQ	33\$; NO - SKIP
9632	045454	052737	000002	005474	BIS	#BSERR,RECODE	; SET ERROR FLAG
9633	045462	016500	000030		MOV	P.DCYL(R5),R0	; GET CYL NO.
9634	045466	116501	000027		MOVB	P.DTS+1(R5),R1	; GET TRACK NO.
9635	045472	042701	177774		BIC	#177774,R1	; CLEAR ALL BITS EXCEPT TRACK
9636	045476	016502	000026		MOV	P.DTS(R5),R2	; GET SECTOR IN ERROR
9637	045502	042702	177740		BIC	#177740,R2	; CLEAR ALL BITS EXCEPT SECTOR
9638	045506	004737	042436		JSR	PC,BDSRCK	; GO SEE IF THIS SECTOR LISTED
9639	045512	032737	001000	005474	BIT	#BADSEC,RECODE	; TEST RESULT
9640	045520	001065			BNE	37\$; YES - EXIT, NO ERROR OR REPORT
9641	045522	104104			ERROR	104	; ELSE REPORT BAD BSE
9642	045524	042737	000002	005474	BIC	#BSERR,RECODE	; RESET BSE ERROR FLAG
9643	045532	000460			BR	37\$; GO EXIT
9644	045534	032704	100000	33\$:	BIT	#DCK,R4	; TEST IF DATA CHECK
9645	045540	001435			BEQ	36\$	
9646	045542	052737	000020	005474	BIS	#DCKERR,RECODE	; SET DATA CHECK ERROR IN RECOVERY CODE
9647	045550	032704	000100		BIT	#ECH,R4	; TEST IF ECC IS HARD. IF
9648	045554	001406			BEQ	34\$; YES SET UNCORRECTABLE IN
9649	045556	052737	000040	005474	BIS	#ECCNC,RECODE	; RECOVERY FLAG AND A 0 IN
9650	045564	005037	001206		CLR	\$REG12	; REG12 TO INDICATE UNCORRECTABLE,
9651	045570	000403			BR	35\$; A 1 IN REG 12 FOR CORRECTABLE
9652	045572	012737	000001	001206	MOV	#1,\$REG12	
9653	045600	105737	003121	34\$:	TSTB	DERCNT	; TEST IF FIRST ERROR
9654	045604	001033		35\$:	BNE	37\$; NO SKIP REPORT
9655	045606	004737	046702		JSR	PC,GTPKAD	; GO GET PACK ADDRESS OF ERROR
9656	045612	016537	000060	001202	MOV	P.EPOS(R5),\$REG10	; STORE ECC POSITION &
9657	045620	016537	000062	001204	MOV	P.EPAT(R5),\$REG11	; PATTERN
9658	045626	104032			ERROR	32	; REPORT DCK ERROR
9659	045630	000137	045674		JMP	37\$	
9660	045634	032765	040000	000020	36\$:	BIT	#WCE,P.CS2(R5) ; TEST WRITE CHECK ERROR
9661	045642	001414			BEQ	37\$	
9662	045644	042737	000200	005474	BIC	#ABORT,RECODE	; CLEAR ABORT & SET WRITE
9663	045652	052737	000100	005474	BIS	#WCERR,RECODE	; CHECK ERROR IN RECODE
9664	045660	105737	003121		TSTB	DERCNT	; TEST IF FIRST ERROR
9665	045664	001003			BNE	37\$; NO - SKIP
9666	045666	004737	046702		JSR	PC,GTPKAD	; GO GET ADDRESS OF ERROR
9667	045672	104033			ERROR	33	; REPORT WCE
9668							
9669	045674	032765	000020	000014	37\$:	BIT	#DRVHRD,P.PRST(R5) ; TEST HARD ERROR
9670	045702	001404			BEQ	43\$	
9671	045704	104036			ERROR	36	
9672	045706	052737	000200	005474	BIS	#ABORT,RECODE	
9673							
9674	045714	032765	000040	000014	43\$:	BIT	#DRVDSK,P.PRST(R5) ; TEST STATUS CHANGE NOT CLEARED
9675	045722	001404			BEQ	44\$	
9676	045724	104037			ERROR	37	
9677	045726	052737	000200	005474	BIS	#ABORT,RECODE	
9678							
9679	045734	032765	004000	000014	44\$:	BIT	#NODSC,P.PRST(R5) ; IFST ATTENTION BUT NO FAULT OR DSC
9680	045742	001404			BEQ	46\$	
9681	045744	104040			ERROR	40	
9682	045746	052737	000200	005474	BIS	#ABORT,RECODE	

```

9683
9684 045754 032765 000010 000014 46$: BIT #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
9685 045762 001404 BEQ ALLTRM
9686 045764 104042 ERROR 42
9687 045766 052737 000200 005474 BIS #ABORT,RECODE
9688
9689
9690 ;ALL ERRORS MUST EXIT THROUGH THIS POINT
9691
9692 045774 012705 002630 ALLTRM: MOV #PARMO,R5 ;RESTORE PARAMETER BLOCK SELECTION
9693 046000 012737 044200 003046 MOV #ERRHDL,A.ABNL ;RESTORE INTERRUPT VECTORS FOR RETRY
9694 046006 012737 042732 003044 MOV #ERRFRE,A.NORM ;DRIVER OPERATIONS, IF ANY
9695 046014 032737 000200 005474 BIT #ABORT,RECODE ;IF ABORT IS NOT SET AND
9696 046022 001043 BNE 48$ ;THE DRIVE IS READY (HAS NOT
9697 ;CYCLED DOWN)
9698 046024 012702 003036 MOV RKBAS,R2 ;GET BASE ADDRESS
9699 046030 032762 000200 000012 BIT #RDY,RKDS(R2) ;TEST IF DRIVE READY SET
9700 046036 001004 BNE 47$ ;RECALIBRATE REQUIRED BIT IS SET
9701 046040 052737 000200 005474 BIS #ABORT,RECODE ;ELSE ABORT WITH ABORT MESSAGE
9702 046046 000431 BR 48$
9703 046050 032737 004000 005474 47$: BIT #RCLREQ,RECODE ;IF RECALIBRATE IS REQUIRED
9704 046056 001443 BEQ BGNRTY ;FOR RETRY SET UP PARAM
9705 046060 112737 000113 000001 MOVB #RECAL,P.CMND ;BLOCK TO DO IT.
9706 046066 012737 046340 003046 MOV #RETANL,A.ABNL
9707 046074 004737 042542 JSR PC,DRVCAL
9708 046100 012737 044200 003046 MOV #ERRHDL,A.ABNL ;RESTORE ERROR RETURN
9709 046106 032737 000400 005474 BIT #LEV2ER,RECODE ;IF AN ERROR OCCURRED IN THE
9710 046114 001424 BEQ BGNRTY ;RECAL ATTEMP SET ABORT,
9711 046116 052737 000200 005474 BIS #ABORT,RECODE ;PRINT THE RECAL ERROR MESSAGE, AND
9712 046124 104060 ERROR 60 ;GO REPORT DETAILS
9713 046126 000137 044220 JMP ERZENT
9714 046132 104061 48$: ERROR 61 ;REPORT ABORT-RETRY FAILED
9715 046134 105037 003130 CLRB DRVERS ;CLR DRV ERR CNT
9716 046140 105037 003126 CLRB ERRCNT
9717 046144 000137 017332 JMP NEWDRV ;TEST NEXT DRV
9718
9719 ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
9720 ;IF THE DRIVE READY BIT IS RESET.
9721
9722 046150 000005 HLTPRG: RESET ;DISABLE ALL DEVICES
9723 046152 000000 HALT ;HALT THE CPU
9724 046154 105037 003126 CLRB ERRCNT ;CLEAR ERROR COUNT
9725 046160 000005 RESET ;RESET ALL DEVICES
9726 046162 000137 013410 JMP CMSTRT
9727
9728
9729 ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
9730 ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
9731 ;THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
9732 ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
9733 ;FLAG IS SET AND PROGRAM HALTS.
9734
9735 046166 032737 000136 005474 BGNRTY: BIT #BSERR!HVR CER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
9736 046174 001404 BEQ 3$
9737 046176 052737 100000 005474 9$: BIS #ANYDER,RECODE
9738 046204 000453 BR 2$

```

```

9739 046206 105737 003142 3$: TSTB NORTRY ;SEE IF "NO-RETRY" FLAG SET
9740 046206 001371 003142 BNE 9$ ;BR IF YES
9741 046212 001371 003142 BIT #BADSEC,RECODE ;TEST IF BAD SECTOR FLAG SET
9742 046214 032737 001000 005474 BNE 2$ ;IF YES-EXIT TO CALLER
9743 046222 001044 003126 003127 CMPB EPRCNT,ERRLMT ;BEGIN RETRY IF ERROR COUNT HAS
9744 046224 123737 003126 003127 BNE 1$ ;NOT BEEN EXCEEDED
9745 046232 001012 003126 003127 CLR $REG5 ;GET ERROR RETRY COUNT
9746 046234 005037 001174 001174 MOVB ERRCNT,$REG5 ;REPORT RETRY UNSUCCESSFUL
9747 046240 113737 003126 001174 ERROR 102 ;SET ABORT & QUIT
9748 046246 104102 000200 005474 BIS #ABORT,RECODE ;GET RK BASE ADDRESS
9749 046250 052737 000200 005474 BR HLTPRG ;SET UP TO CLEAR SUBSYSTEM
9750 046256 000734 003036 000001 1$: MOV RKBAS,R2 ;GO DO IT
9751 046260 013702 003036 000001 MOVB #SUBCLR,P.CMND(R5) ;GO AND REESTABLISH THE COMMAND
9752 046264 112765 000177 000001 JSR PC,DRVCAL ;AS IT WAS ENTERED INTO THE
9753 046272 004737 042542 005240 MOV #COMSTR,R0 ;PARAMETER BLOCK
9754 046276 012700 005240 MOV (R0)+,(R5)+
9755 046302 012025 MOV (R0)+,(R5)+
9756 046304 012025 MOV (R0)+,(R5)+
9757 046306 012025 MOV (R0)+,(R5)+
9758 046310 012025 MOV (R0)+,(R5)+
9759 046312 012025 MOV (R0)+,(R5)+
9760 046314 012025 MOV (R0)+,(R5)+
9761 046316 012705 002630 MOV #PARMO,R5 ;LOWER PRIORITY TO ALLOW INTERRUPT
9762 046322 012737 000000 177776 MOV #PRO,PSW ;CALL DRIVER
9763 046330 004737 042542 JSR PC,DRVCAL ;IF RETURN GETS HERE, NO ERROR
9764 046334 104410 042542 2$: RETREG ;OCCURRED, RECOVERY WAS SUCCESSFUL
9765 046336 000207 RTS PC
9766
9767
9768 046340 152737 000377 003123 RETANL: BISB #377,DONE ;SET DONE
9769 046346 052737 000400 005474 BIS #LEV2ER,RECODE ;SET LEVEL TWO ERROR
9770 046354 000207 003123 RTS PC
9771 046356 152737 000377 003123 RETNML: BISB #377,DONE ;SET DONE
9772 046364 000207 003123 RTS PC
9773
9774
9775 ;*****
9776 ;SBTTL TIME OUT PROCESSOR ROUTINE
9777 ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
9778 ;*GATHERING DUTIES.
9779 ;*****
9780
9781 046366 104407 003036 000000 177776 TOPROC: SAVREG
9782 046366 013702 003036 000000 MOV RKBAS,R2
9783 046370 012701 001174 000000 MOV #SREG5,R1 ;SET UP R1 FOR RK REGISTER STORAGE
9784 046400 016221 000000 000000 MOV RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
9785 046404 016221 000010 000010 MOV RKCS2(R2),(R1)+ ;REGISTERS
9786 046410 016221 000020 000020 MOV RKDC(R2),(R1)+
9787 046414 016221 000006 000006 MOV RKDA(R2),(R1)+
9788 046420 016221 000002 000002 MOV RKWC(R2),(R1)+
9789 046424 016221 000004 000004 MOV RKBA(R2),(R1)+
9790 046430 016221 000016 000016 MOV RKASOF(R2),(R1)+
9791 046434 016221 000012 000012 MOV RKDS(R2),(R1)+
9792 046440 016221 000014 000014 MOV RKER(R2),(R1)+
9793 046444 005000 CLR R0 ;THIS CODE WILL ATTEMPT TO
9794 ;RETRIEVE THE STATUS FROM THE

```



```
9795                                     ;DRIVE.
9796 046446 012705 002714             MOV  #PARI, R5             ;SET UP TO USE PARI
9797 046452 112765 000141 000001     MOVB #RDSTAT, P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
9798 046460 004737 042542             JSR  PC, DRVCAL           ;CALL DRIVER
9799 046464 032765 000100 000014     BIT  #CMDTO, P.PRST(R5) ;TEST FOR TIMEOUT
9800 046472 001403                     BEQ  1$                  ;NO - SKIP
9801 046474 052737 002000 005474     BIS  #TWCTOS, RECODE       ;SET TWO TIMEOUTS FLAG
9802 046502 062705 000040             ADD  #P.AOD, R5          ;BUMP R5 TO POINT TO DRIVE STATUS
9803 046506 012521                     MOV  (R5)+, (R1)+       ;MOVE ALL THE DRIVE STATUS INTO THE
9804 046510 012521                     MOV  (R5)+, (R1)+       ;TEMP REGS FOR REPORTING.
9805 046512 012521                     MOV  (R5)+, (R1)+
9806 046514 012521                     MOV  (R5)+, (R1)+
9807 046516 012521                     MOV  (R5)+, (R1)+
9808 046520 012521                     MOV  (R5)+, (R1)+
9809 046522 012521                     MOV  (R5)+, (R1)+
9810 046524 012521                     MOV  (R5)+, (R1)+
9811
9812 046526 012705 002630             MOV  #PARMO, R5           ;RESTORE PARM 0
9813 046532 104410                     RESREG
9814 046534 000207                     RTS  PC
9815
9816
9817
9818                                     ;*****
9819 :SBTTL READ HEADER 0 ROUTINE
9820 :*****
9821 RDHDO:
9822 046536 104407                     SAVREG
9823 046540 016501 000026             MOV  P.DTS(R5), R1         ;STORE TRACK AND SECTOR
9824 046544 016500 000052             MOV  P.B10(R5), R0        ;GET THE CYLINDER ADRS
9825 046550 042700 160017             BIC  #160017, R0         ;FROM THE DRIVE STATUS. CLEAR
9826 046554 006200                     ASR  R0                 ;OFF UNUSED BITS AND POSITION
9827 046556 006200                     ASR  R0                 ;FOR USE AS THE DESIRED
9828 046560 006200                     ASR  R0                 ;CYLINDER IN THE READ
9829 046562 006200                     ASR  R0                 ;HEADER COMMAND.
9830 046564 012705 002714             MOV  #PARI, R5           ;SET UP TO USE P.B. 1
9831 046570 010165 000004             MOV  R1, P.SECT(R5)       ;INSERT TRK, SECT FOR READ HDR
9832 046574 010065 007002             MOV  R0, P.CYLN(R5)       ;SET CYL NO.
9833 046600 112765 000177 000001     MOVB #SUBCLR, P.CMND(R5) ;SET S.S. CLEAR CMND
9834 046606 012737 000000 177776     MOV  #PRO, #PSW         ;ALLOW INTERRUPTS
9835 046614 004737 042542             JSR  PC, DRVCAL           ;CLEAR THE S.S.
9836 046620 112765 000125 000001     MOVB #RDHEAD, P.CMND(R5) ;SET READ HEADER COMMAND
9837 046626 004737 042542             JSR  PC, DRVCAL           ;DO READ HEADER
9838 046632 012712 000025             MOV  #25, (R2)          ;ISSUE RD HDR WITH FMT BIT = 0
9839 046636 032712 000200             BIT  #BIT7, (R2)         ;CHECK FOR C. READY IN CSI
9840 046642 001775                     BEQ  4$                  ;BR IF NOT RDY YET
9841 046644 012712 010025             MOV  #10025, (R2)        ;ISSUE RD HDR WITH FMT BIT = 1
9842 046650 032712 000200             BIT  #BIT7, (R2)         ;CHECK FOR C. READY IN CSI
9843 046654 001775                     BEQ  6$                  ;BR IF NOT RDY YET
9844 046656 142765 000020 000007     BICB #B.CFMT, P.CSIH(R5) ;CLEAR THE FORMAT BIT
9845 046664 153765 003125 000007     BISB FORMAT, P.CSIH(R5) ;RESTORE TYPE AND FMT IN USE
9846 046672 012705 002630             MOV  #PARMO, R5           ;RESTORE P.B. 0 ADDRESS
9847 046676 104410                     RESREG
9848 046700 000207                     RTS  PC
9849
9850                                     ;RETURN
```

```
9851
9852
9853
9854
9855 046702 016537 000030 001174
9856 046710 005037 001176
9857 046714 005037 001200
9858 046720 116537 000026 001200
9859 046726 116537 000027 001176
9860 046734 005737 001200
9861
9862 046740 001403
9863 046742 005337 001200
9864 046746 000426
9865 046750 032765 010000 000016 1$:
9866 046756 001404
9867 046760 012737 000023 001200
9868 046766 000403
9869 046770 012737 000025 001200 2$:
9870 046776 005737 001176 3$:
9871 047002 001403
9872 047004 005337 001176
9873 047010 000405
9874 047012 012737 000002 001176 4$:
9875 047020 005337 001174
9876 047024 000207 5$:
9877
9878
9879
9880
9881
9882
9883
9884
9885
9886 047026 104407
9887 047030 016537 000030 001174
9888 047036 016501 000026
9889 047042 042701 174377
9890 047046 006201
9891 047050 006201
9892 047052 006201
9893
9894 047054 016537 000026 001176
9895 047062 042737 177740 001176
9896 047070 060137 001176
9897 047074 052737 140000 001176
9898 047102 032765 010000 000016
9899 047110 001403
9900 047112 052737 001000 001176
9901 047120 013737 001174 001200 23$:
9902 047126 013701 001176
9903 047132 043737 001176 001200
9904 047140 043701 001174
9905 047144 050137 001200
9906 047150 104410

;*****
;SBTTL GET PACK ADDRESS ROUTINE
;*****
GTPKAD: MOV P.DCYL(R5), $REG5 ;GET CYLINDER NUMBER
        CLR $REG6 ;CLEAR REGISTERS FOR
        CLR $REG7 ;TRACK & SECTOR STORAGE
        MOV8 P.DTS(R5), $REG7 ;STORE THE TRACK AND SECTOR
        MOV8 P.DTS+1(R5), $REG6
        TST $REG7 ;ADJUST THE ADDRESS CONTAINED IN
                    ;THE RK REGISTERS FOR THE AUTOMATIC
                    ;INCREMENT
        BEQ 1$
        DEC $REG7
        BR 5$
        BIT #CFMT, P.CS1(R5)
        BEQ 2$
        MOV #19., $REG7
        BR 3$
        MOV #21., $REG7
        TST $REG6
        BEQ 4$
        DEC $REG6
        BR 5$
        MOV #2, $REG6
        DEC $REG5
        RTS PC

;*****
;SBTTL BUILD EXPECTED HEADER
;USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
;WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
;7 FOR REPORTING.
;*****
BLDEXH: SAVREG
        MOV P.DCYL(R5), $REG5 ;CONSTRUCT EXPECTED HDR
        MOV P.DTS(R5), R1 ;DESIRED CYLINDER & DESIRED TRACK
        BIC #174377, R1 ;CLEAR ALL BUT TRACK BITS
        ASR R1 ;AND SECTOR. SHIFT THE TRACK
        ASR R1 ;OVER TO CONFORM TO HEADER FORMAT
        ASR R1 ;CHECK THE FORMAT BIT AND
        ;IF SET, SET THE HEADER FORMAT
        MOV P.DTS(R5), $REG6 ;BIT.
        BIC #177740, $REG6 ;CLEAR ALL BUT SECTOR
        ADD R1, $REG6 ;ADD TRACK AND SECTOR TOGETHER
        BIS #140000, $REG6 ;INSERT BSE BITS
        BIT #CFMT, P.CS1(R5)
        BEQ 23$
        BIS #1000, $REG6
        MOV $REG5, $REG7 ;COMPUTE THE HEADER VRC
        MOV $REG6, R1
        BIC $REG6, $REG7
        BIC $REG5, R1
        BIS R1, $REG7
        RESREG
```

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 194
BUILD EXPECTED HEADER

L15

SEQ 0193

9907 047152 000207
9908

RTS PC

M15

9909
9910
9911
9912
9913
9914
9915
9916
9917
9918
9919
9920
9921
9922
9923
9924
9925
9926
9927
9928
9929
9930
9931
9932
9933
9934
9935
9936
9937
9938
9939
9940
9941
9942
9943
9944
9945
9946
9947
9948
9949
9950
9951
9952
9953
9954
9955
9956
9957
9958
9959
9960
9961
9962
9963
9964

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

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

.SBTTL *WATCH-DOG TIMER

```

*****
*
*   THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06-RK07 UNIBUS
*   SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
*   REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
*   THE RK06-RK07 DRIVER WILL USE THE LOCATION W.MTIM FOR
*   MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
*   WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
*   DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
*   (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
*   IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
*   TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
*   REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
*
*   THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
*   OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
*   THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
*   LIMIT FOR ALL OTHER COMMANDS.
*
*   FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
*   WATCH UP TO 8 OPERATIONS SIMULTANEOUSLY. FOR SEQUENTIAL
*   OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
*
*CALL JSR PC,W.WTCH
*   RETURN IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
*
*   OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
*   BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
*   IN THE PROGRAM DEVICE STATUS REGISTER OF THE
*   APPROPRIATE PARAMETER BLOCK WILL BE SET.
*****

```

```

W.WTCH: MOV R5,-(SP) ;SAVE R5 ON THE STACK
        MOV R4,-(SP) ;SAVE R4 ON THE STACK
        MOV R3,-(SP) ;SAVE R3 ON THE STACK
        MOV R2,-(SP) ;SAVE R2 ON STACK
        MOV PS,-(SP) ;SAVE PROGRAM STATUS WORD ON STACK
        DEC W.MTIM ;DECREMENT MILLISECOND TIMER
        BNE 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

```

047154	010546		
047156	010446		
047160	010346		
047162	010246		
047164	013746	177776	
047170	005337	003056	
047174	001034		
047176	013737	003060	003056
047204	105737	003100	
047210	001426		
047212	013737	003042	177776
047220	013702	003036	
047224	005337	003114	
047230	001016		

CZR6N00 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 196
*WATCH-DOG TIMER

SEQ 0195

9965	047232	105037	003100		CLRB	W.TIME	;RESET TIMING INDICATOR
9966	047236	013705	003112		MOV	PBLKT,R5	;LOAD ADDRESS OF PARAMETER BLOCK
9967							TABLE FOR INDEXING
9968	047242	052765	000100	000014	BIS	#CMDTO,P.PRST(R5)	;SET COMMAND TIME OUT
9969	047250	020537	003054		CMP	R5,0.WAIT	;CHECK IF DRIVER IS WAITING FOR
9970							COMMAND COMPLETION
9971	047254	001002			BNE	SS	;NO, DO NOT ALTER WAITING FOR
9972							COMMAND COMPLETION
9973	047256	005037	003054		CLR	0.WAIT	;CLEAR WAIT FOR COMMAND COMPLETION
9974	047262	004737	052536	SS:	JSR	PC,R.ABNL	;BRANCH TO ERROR ROUTINE
9975	047266	012637	177776	20\$:	MOV	(SP)+,PS	;RESTORE PSW
9976	047272	012602			MOV	(SP)+,R2	;RESTORE R2
9977	047274	012603			MOV	(SP)+,R3	;RESTORE R3
9978	047276	012604			MOV	(SP)+,R4	;RESTORE R4
9979	047300	012605			MOV	(SP)+,R5	;RESTORE R5
9980	047302	000207			RTS	PC	;RETURN

.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

:*****

```

9981
9982
9983
9984
9985
9986
9987
9988
9989
9990
9991
9992
9993
9994
9995
9996
9997
9998
9999
10000
10001
10002
10003
10004
10005
10006
10007
10008
10009
10010
10011
10012
10013
10014
10015
10016
10017
10018
10019
10020
10021
10022
10023
10024
10025 047304 010546
10026 047306 010446
10027 047310 010346
10028 047312 010246
10029 047314 010146
10030 047316 010046
10031 047320 013702 003036
10032 047324 016237 000010 003002
10033 047332 032737 001000 003002
10034 047340 001407
10035 047342 052737 100000 003052
10036 047350 004737 052562

```

```

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

```

```

10037 047354 000137 051534      JMP      I.RTRN      ;RETURN
10038
10039 047360 105737 003074      1$:  TSTB      I.ISRL      ;CHECK IF INTERRUPT OR RELEASE
10040 047364 001410              2EQ      6$          ;NO, CHECK IF DRIVE AVAILABLE
10041 047366 100403              BMI      5$          ;CHECK IF RELEASE COMMAND
10042 047370 105037 003074      CLRB      I.ISRL      ;YES, CLEAR FLAG
10043 047374 000473              BR       I.ICO      ;CONTINUE PROCESSING INTERRUPT
10044
10045 047376 105037 003074      5$:  CLRB      I.ISRL      ;CLEAR FLAG
10046 047402 000137 050516      JMP      I.ATTN      ;GO PROCESS DRIVE ATTENTIONS
10047
10048 047406 032737 010400 003002 6$:  BIT      #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
10049                                UNIT FIELD ERROR
10050 047414 001413              BEQ      7$          ;NO, WAIT FOR DUAL ACCESS INTERRUPT
10051 047416 013704 003002      MOV      T.CS2,R4      ;LOAD R4 FOR DRIVE NUMBER
10052 047422 042704 177770      BIC      #1C<DRVMSK>,R4 ;KEEP DRIVE BITS
10053 047426 013705 003112      MOV      PBLKT,R5      ;STORE PARAMETER BLOCK ADDRESS
10054 047432 016237 000000 003000  MOV      RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
10055 047440 000137 047726      JMP      I.ERRC      ;REPORT ERROR
10056
10057 047444 016237 000012 003020 7$:  MOV      RKDS(R2),T.DS ;STORE STATUS REGISTER FOR COMPARISON
10058 047452 032737 000001 003020  BIT      #DRA,T.DS      ;CHECK IF DRIVE SEIZED BY OTHER
10059                                PORT
10060 047460 001041              BNE      I.IDD      ;NO, CONTINUE PROCESSING INTERRUPT
10061
10062                                ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
10063 047462 032737 164000 003002  BIT      #DLT!WCE!UPE!NEM,T.CS2
10064
10065 047470 001007              BNE      10$         ;INDICATE ERROR
10066 047472 016237 000014 003016  MOV      RKER(R2),T.ER ;STORE ERROR REGISTER
10067
10068                                ;CHECK FOR DATA TRANSFER ERROR TYPE ERROR
10069 047500 032737 125700 003016  BIT      #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10070
10071 047506 001407              BEQ      11$         ;NO, WAIT FOR RELEASE OF RK06 DRIVE
10072
10073 047510 052737 000010 003052 10$:  BIS      #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
10074 047516 004737 052562      JSR      PC,R.CONT ;REPORT ERROR
10075 047522 000137 051534      JMP      I.RTRN      ;RESTORE REGISTERS
10076
10077 047526 105037 003100      11$:  CLRB      W.TIME      ;RESET TIMING ON THIS DRIVE
10078 047532 005037 003114      CLR      W.DRV      ;CLEAR TIMING COUNT FOR THIS DRIVE
10079 047536 013705 003112      MOV      PBLKT,R5      ;LOAD R5 WITH PARAMETER BLOCK
10080                                ADDRESS
10081 047542 052765 010000 000014  BIS      #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
10082                                PROGRAM DRIVE STATUS REGISTER
10083 047550 005037 003054      CLR      0.WAIT      ;CLEAR WAIT FOR COMMAND COMPLETION
10084 047554 004737 052536      JSR      PC,R.ABNL ;INDICATE ABNORMAL TERMINATION
10085 047560 000137 051534      JMP      I.RTRN      ;GO RESTORE REGISTERS
10086
10087 047564 013705 003054      I.IDD:  MOV      0.WAIT,R5 ;LOAD PARAMETER BLOCK ADDRESS INTO R5
10088 047570 001002              BNE      2$          ;IS COMMAND WAITING PROCESSING
10089                                YES, DO PROCESSING
10090 047572 000137 050516      JMP      I.ATTN      ;NO, PROCESS ATTENTION
10091
10092 047576 013704 003002      2$:  MOV      T.CS2,R4      ;STORE RKCS2 FOR DRIVE NUMBER

```

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 199
*RK06 INTERRUPT SERVICE ROUTINE

SEQ 0198

10093	047602	042704	177770			BIC	#IC<DRVMSK>,R4	;MASK OUT UNNECESSARY BITS
10094								
10095								
10096	047606	126504	000000			CMPB	P.DRVN(R5),R4	;CHECK IF DRIVE NUMBER IS EXPECTED
10097	047612	001401				BEQ	3\$;YES, CONTINUE
10098	047614	000000				HALT		;NO, DRIVER ERROR
10099	047616	122765	000164	000001	3\$:	CMPB	#RDALHD,P.CMND(R5)	;CHECK IF READ ALL HEADERS
10100	047624	001002				BNE	10\$;NO, EXECUTE NORMAL DATA TRANSFER
10101	047626	000137	050164			JMP	I.HDAL	;GO EXECUTE SPECIAL HEADER SEQUENCE
10102								
10103	047632	005037	003054		10\$:	CLR	O.WAIT	;CLEAR WAIT FOR COMMAND COMPLETION
10104	047636	005037	003114			CLR	W.DRV	;CLEAR WATCH-DOG TIME
10105	047642	105037	003100			CLRB	W.TIME	;RESET TIMING ON THIS DRIVE
10106	047646	016237	000000	003000		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REGISTER 1
10107	047654	032737	100000	003000		BIT	#CERR,T.CS1	;CHECK IF CONTROLLER ERROR
10108	047662	001021				BNE	I.ERRC	;YES, PROCESS ERROR
10109	047664	016237	000016	003014		MOV	RKASOF(R2),T.ASOF	;STORE ATTENTION SUMMARY
10110	047672	133737	003101	003015		BITB	INTMSK,T.ASOF+1	;CHECK IF DRIVE ATTENTION SET
10111	047700	001004				BNE	15\$;YES, REPORT ERROR
10112	047702	004737	052550			JSR	PC,R.NORM	;INDICATE NORMAL RETURN
10113	047706	000137	051534			JMP	I.ATRN	;RESTORE REGISTERS
10114								
10115	047712	052745	000010	000014	15\$:	BIS	#UEXATT,P.PRST(R5)	;SET UNEXPECTED ATTENTION
10116								
10117	047720	004737	052204		I.ERRA:	JSR	PC,I.CSTS	;STORE CONTROLLER STATUS
10118	047724	000405				BR	I.ERR	;STORE PATTERN AND POSITION INFORMATION
10119								
10120	047726	013765	003000	000016	I.ERRC:	MOV	T.CS1,P.CS1(R5)	;GET ERROR RACS1
10121	047734	004737	052226			JSR	PC,I.CST1	;GET REST OF CONTROLLER STATUS
10122	047740	016265	000032	000062	I.ERR:	MOV	RKECPT(R2),P.EPAT(R5)	;STORE ECC PATTERN
10123	047746	016265	000030	000060		MOV	RKECPS(R2),P.EPOS(R5)	;STORE ECC POSITION
10124	047754	004037	051552			JSR	RD,I.CCLR	;CLEAR CONTROLLER
10125	047760	051534				I.ATRN		;ERROR RETURN
10126	047762	032765	010400	000020		BIT	#NED!UFE,P.CS2(R5)	;CHECK IF IT WAS NON-EXISTENT DRIVE OR
10127								;UNIT FIELD ERROR
10128	047770	001046				BNE	5\$;YES, REPORT ERROR
10129	047772	004037	052310			JSR	RD,I.STAT	;GATHER DRIVE STATUS
10130	047776	051534				I.ATRN		;ERROR RETURN
10131	050000	112737	000005	003000		MOVB	#DR.CLR,T.CS1	;LOAD COMMAND
10132	050006	004037	051634			JSR	RD,I.ISSU	;ISSUE DRIVE CLEAR
10133	050012	051534				I.ATRN		;ERROR RETURN
10134	050014	133737	003101	003015		BITB	INTMSK,T.ASOF+1	;CHECK IF ATTENTION RESET
10135	050022	001407				BEQ	2\$;NO, INDICATE DRIVE ERROR
10136	050024	052737	000020	003052		BIS	#E.CLAT,E.CONT	;SET ATTENTION DID NOT RESET
10137								;WITH CLEAR
10138	050032	004737	052562			JSR	PC,R.CONT	;REPORT CONTROLLER ERROR
10139	050036	000137	051534			JMP	I.ATRN	;GO RESTORE REGISTERS
10140								
10141	050042	032737	040000	003024	2\$:	BIT	#S.DSC,T.MR2	;CHECK IF DRIVE STATUS CHANGE CLEARED
10142	050050	001403				BEQ	3\$;YES, CHECK FAULT
10143	050052	052765	000040	000014		BIS	#DRVDSC,P.PRST(R5)	;SET DSC DID NOT CLEAR
10144	050060	032737	001000	003026	3\$:	BIT	#S.PAR,T.MR3	;CHECK IF DRIVE PARITY ERROR
10145	050066	001407				BEQ	5\$;NO, INDICATE ABNORMAL TERMINATION
10146	050070	052737	002000	003052		BIS	#E.DPAR,E.CONT	;SET DRIVE PARITY ERROR
10147	050076	004737	052562			JSR	PC,R.CONT	;INDICATE CONTROLLER ERROR
10148	050102	000137	051534			JMP	I.ATRN	;RETURN


```

10149
10150 050106 032765 000020 000014 5$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
10151 050114 001017 BNE 10$ ;YES, GO REPORT ERROR
10152 050116 032737 020000 003024 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
10153 050124 001413 BEQ 10$ ;NO, REPORT ERROR
10154 050126 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
10155 050134 113737 003101 003100 MOVB INTMSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
10156 050142 013737 003064 003114 MOV W.8SEC,W.DRV
10157 050150 000137 051534 JMP I.RTRN ;GO RESTORE REGISTERS
10158
10159 050154 004737 052536 10$: JSR PC,R.ABNL ;GO REPORT ERROR
10160 050160 000137 051534 JMP I.RTRN ;GO RESTORE REGISTERS
10161
10162 .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
10163
10164 050164 016237 000000 003000 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
10165 ;ERROR
10166 050172 032737 100000 003000 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
10167 050200 001422 BEQ 5$ ;NO, CHECK FOR ATTENTION
10168
10169 050202 005037 003054 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
10170 050206 105037 003100 CLRB W.TIME ;RESET TIMING ON DRIVE
10171 050212 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT
10172 050216 013765 003000 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
10173 050224 004737 052226 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
10174 050230 004037 051552 JSR RO,I.CCLR ;CLEAR CONTROLLER
10175 050234 051534 I.RTRN ;ERROR RETURN
10176 050236 004737 052536 JSR PC,R.ABNL ;INDICATE ERROR RETURN
10177 050242 000137 051534 JMP I.RTRN ;RESTORE REGISTERS
10178
10179 050246 016237 000016 003014 5$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10180 050254 133737 003101 003015 BITB INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
10181 050262 001410 BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
10182 050264 005037 003054 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
10183 050270 105037 003100 CLRB W.TIME ;RESET TIMING ON DRIVE
10184 050274 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT
10185 050300 000137 047720 JMP I.ERRA ;GO REPORT ERROR
10186
10187 050304 013701 003070 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
10188 050310 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
10189 050314 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
10190 050320 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
10191 050324 010137 003070 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
10192 050330 016237 000010 003002 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
10193 050336 032737 100000 003002 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
10194 050344 001055 BNE 35$ ;YES, REPORT ERROR
10195 050346 005337 003072 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
10196 050352 001026 BNE 25$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
10197 050354 005037 003054 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
10198 050360 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
10199 050364 105037 003100 CLRB W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
10200 050370 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COL IT
10201 050376 112737 000001 003000 MOVB #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
10202 050404 004037 051634 JSR RO,I.ISSU ;GET SECTOR COUNT
10203 050410 051534 I.RTRN ;ERROR RETURN
10204 050412 013765 003026 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 201
*READ ALL HEADERS INTERRUPT SEQUENCE

SEQ 0200

```

10205 050420 004737 052550      JSR    PC.R.NORM      ;INDICATE NORMAL TERMINATION
10206 050424 000137 051534      JMP    I.ATRN        ;RESTORE REGISTERS
10207
10208 050430 016562 000002 000020 25$: MOV    P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
10209 050436 016562 000004 000006      MOV    P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10210 050444 116565 000007 000017      MOVVB  P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10211 050452 042765 165777 000016      BIC    #1C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
10212                                     ; DRIVE TYPE
10213 050460 112765 000125 000016      MOVVB  #RDHEAD,P.CS1(R5) ;STORE COMMAND ISSUED
10214 050466 016562 000016 000000      MOV    P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
10215 050474 000137 051534      JMP    I.ATRN        ;RESTORE REGISTERS
10216
10217 050500 052737 000400 003052 35$: BIS    #E.DLT,E.CONT ;SET DATA LATE WHILE UNLOADING HEADER
10218 050506 004737 052562      JSR    PC.R.CONT      ;REPORT ERROR
10219 050512 000137 051534      JMP    I.ATRN        ;RESTORE REGISTERS
10220
10221      .SBTTL  *DRIVE ATTENTION SCANNER
10222
10223 050516 016237 000000 003000 I.ATTN: MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
10224                                     ; REGISTER 1 FOR COMPARISON
10225 050524 032737 100000 003000      BIT    #CERR,T.CS1      ;CHECK IF CONTROLLER ERROR OCCUPED
10226 050532 001441                                     BEQ    SS              ;NO, CHECK IF ATTENTION
10227
10228                                     ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
10229 050534 032737 164000 003002      BIT    #DLT!WCE!UPE!NEM,T.CS2
10230
10231 050542 001007                                     BNE    1$             ;INDICATE ERROR
10232 050544 016237 000014 003016      MOV    RKER(R2),T.ER      ;STORE ERROR REGISTER
10233
10234                                     ; CHECK FOR DATA TRANSFER ERROR TYPE
10235 050552 032737 125700 003016      BIT    #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10236
10237 050560 001407                                     BEQ    2$             ;NO DATA TRANSFER ERROR
10238
10239 050562 052737 000010 003052 1$: BIS    #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
10240 050570 004737 052562      JSR    PC.R.CONT      ;REPORT ERROR
10241 050574 000137 051534      JMP    I.ATRN        ;RESTORE REGISTERS
10242
10243 050600 013704 003002 2$: MOV    T.CS2,R4      ;SAVE CS2 FOR REGISTER NUMBER
10244 050604 042704 177770      BIC    #1C<DRVMSK>,R4 ;STRIP OFF JUNK
10245 050610 105037 003100      CLRVB  W.TIME         ;CLEAR WATCH DOG TIMER
10246 050614 005037 003114      CLR    W.DRV         ;RESET TIMER VALUE
10247 050620 013705 003112      MOV    PBLKT,R5      ;STORE PARAMETER BLOCK ADDRESS IN R5
10248
10249                                     ;
10250                                     ; CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
10251 050624 042765 000006 000014      BIC    #DRVPOS!DRVPDT,P.PRST(R5)
10252
10253 050632 000137 047726      JMP    I.ERRC        ;GO REPORT ERROR
10254
10255 050636 032737 040000 003000 5$: BIT    #DI,T.CS1      ;CHECK IF ANY DRIVE ATTENTION
10256 050644 001002                                     BNE    6$             ;YES, PROCESS INTERRUPT
10257 050646 000137 051534      JMP    I.ATRN        ;RESTORE REGISTERS
10258
10259 050652 016237 000016 003014 6$: MOV    RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10260 050660 105737 003015      TSTB   T.ASOF+1      ;CHECK IF ANY ATTENTIONS SET

```

10261	050664	001007			BNE	7\$;YES GO PROCESS INTERRUPT
10262	050666	052737	000002	003052	BIS	#E.NOAT,E.CONT	;SET NO ATTENTION IN ATTENTION SUMMARY
10263	050674	004737	052562		JSR	PC.R.CONT	;GO REPORT ERROR
10264	050700	000137	051534		JMP	I.ATRNL	;GO RESTORE REGISTERS
10265							
10266	050704	133737	003101	003015	7\$: BITB	INTMSK,T.ASOF+1	;CHECK IF DESIRED INTERRUPT
10267	050712	001007			BNE	8\$;YES, GO PROCESS IT
10268	050714	052737	000004	003052	BIS	#E.UATT,E.CONT	;SET UNSOLICATED ATTENTION
10269	050722	004737	052562		JSR	PC.R.CONT	;GO REPORT ERROR
10270	050726	000137	051534		JMP	I.ATRNL	;GO RESTORE REGISTERS
10271							
10272	050732	013705	003112		8\$: MOV	PBLKT,R5	;STORE PARAMETER BLOCK TABLE
10273	050736	116504	000000		MOVB	P.DRVN(R5),R4	;STORE DRIVE NUMBER
10274	050742	032765	020000	000014	BIT	#E.UNLD,P.PRST(R5)	;CHECK IF DRIVE UNLOADING
10275	050750	001402			BEQ	11\$;NO CONTINUE
10276	050752	000137	051434		JMP	I.UNLD	;SERVICE DRIVE IN POSITION AFTER ERROR
10277							
10278	050756	042765	000002	000014	11\$: BIC	#DRVPOS,P.PRST(R5)	;RESET DRIVE POSITIONING
10279	050764	005062	000026		CLR	RKMRI(R2)	;CLEAR MAINTENANCE REGISTER 1
10280	050770	112737	000001	003000	MOVB	#DR.SEL,T.CS1	;LOAD COMMAND
10281	050776	004037	051634		JSR	RO,I.ISSU	;SELECT DRIVE WITH ATTENTION HIGH
10282	051002	051534			I.ATRNL		;ERROR RETURN
10283	051004	013765	003026	000042	MOV	T.MR3,P.B00(R5)	;STORE STATUS BYTE OO MESS B
10284	051012	032765	000200	000042	BIT	#S.FLT,P.B00(R5)	;CHECK IF DRIVE FAULT
10285	051020	001401			BEQ	12\$;NO CHECK FOR DRIVE STATUS CHANGE
10286	051022	000461			BR	I.AERR	;PROCESS ERROR
10287							
10288	051024	013765	003024	000040	12\$: MOV	T.MR2,P.A00(R5)	;STORE MAINTENANCE REGISTER 2
10289	051032	032765	040000	000040	BIT	#S.DSC,P.A00(R5)	;CHECK FOR DRIVE STATUS CHANGE
10290	051040	001004			BNE	13\$;YES, PROCESS DRIVE STATUS CHANGE
10291	051042	052765	004000	000014	BIS	#NODSC,P.PRST(R5)	;SET NO DRIVE STATUS CHANGE
10292	051050	000446			BR	I.AERR	;PROCESS ERROR
10293							
10294	051052	112737	000005	003000	13\$: MOVB	#DR.CLR,T.CS1	;LOAD COMMAND
10295	051060	004037	051634		JSR	RO,I.ISSU	;CLEAR DRIVE STATUS CHANGE
10296	051064	051534			I.ATRNL		;ERROR RETUN
10297	051066	013765	003014	000032	MOV	T.ASOF,P.ASOF(R5)	;STORE ATTENTION SUMMARY
10298	051074	133765	003101	000033	BITB	INTMSK,P.ASOF+1(R5)	;CHECK IF ATTENTION RESET
10299	051102	001407			BEQ	15\$;YES, CONTINUE INTERRUPT PROCESSING
10300	051104	052737	000020	003052	BIS	#E.CLAT,E.CONT	;SET ATTENTION DID NOT RESET
10301							;WITH DRIVE CLEAR
10302	051112	004737	052562		JSR	PC.R.CONT	;FLAG ERROR
10303	051116	000137	051534		JMP	I.ATRNL	;RESTORE REGISTERS
10304							
10305	051122	013765	003024	000040	15\$: MOV	T.MR2,P.A00(R5)	;STORE MAINTENANCE REGISTER 2
10306	051130	032765	040000	000040	BIT	#S.DSC,P.A00(R5)	;CHECK IF DRIVE STATUS CHANGE
10307							;RESET
10308	051136	001404			BEQ	16\$;YES, CONTINUE INTERRUPT PROCESSING
10309	051140	052765	000040	000014	BIS	#DRVDSC,P.PRST(R5)	;SET DRIVE STATUS CHANGE DID NOT CLEAR
10310	051146	000407			BR	I.AERR	;GO PROCESS ERROR
10311							
10312	051150	105037	003100		16\$: CLRB	W.TIME	;RESET TIMING ON THIS DRIVE
10313	051154	005037	003114		CLR	W.DRV	;CLEAR DRIVE TIMING COUNT
10314	051160	004737	052550		JSR	PC.R.NORM	;REPORT SUCCESSFUL COMMAND COMPLETION
10315	051164	000563			BR	I.ATRNL	;RESTORE REGISTERS
10316							

H16

.SBTTL *ATTENTION ERROR HANDLER

```

10317
10318
10319 051166 042765 000004 000014 I.AERR: BIC #DRV PDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
10320                                     OF DATA TRANSFER
10321 051174 105037 003100 CLR B W.TIME ;CLEAR TIMING FOR THIS DRIVE
10322 051200 005037 003114 CLR W.DRV ;RESET WATCH-DOG TIME
10323 051204 042765 177741 000016 BIC #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
10324 051212 042737 000036 003000 BIC #36,T.CS1 ;KEEP CURRENT CONTROLLER STATUS
10325 051220 053765 003000 000016 BIS T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
10326 051226 013765 003002 000020 MOV T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
10327 051234 013765 003004 000022 MOV T.WCR,P.WCR(R5)
10328 051242 013765 003006 000024 MOV T.BA,P.BAR(R5)
10329 051250 013765 003010 000026 MOV T.DA,P.DTS(R5)
10330 051256 013765 003012 000030 MOV T.DC,P.DCYL(R5)
10331 051264 013765 003014 000032 MOV T.ASOF,P.ASOF(R5)
10332 051272 013765 003016 000034 MOV T.ER,P.ER(R5)
10333 051300 013765 003020 000036 MOV T.DS,P.DS(R5)
10334 051306 004037 052310 JSR RO,I.STAT ;GATHER DRIVE STATUS
10335 051312 051534 I.RTRN ;ERROR RETURN
10336 051314 112737 000005 003000 MOVB #DR.CLR,T.CS1 ;LOAD COMMAND
10337 051322 004037 051634 JSR RO,I.ISSU ;CLEAR DRIVE ERRORS
10338 051326 051534 I.RTRN ;ERROR RETURN
10339 051330 133737 003101 003015 BITB INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
10340 051336 001407 BEQ 2$ ;YES, FLAG DRIVE ERROR
10341 051340 052737 000020 003052 BIS #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
10342 051346 004737 052562 JSR PC,R.CONT ;REPORT ERROR
10343 051352 000137 051534 JMP I.RTRN ;RESTORE REGISTERS
10344
10345 051356 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF A HARD DRIVE ERROR
10346 051364 001017 BNE 10$ ;YES, REPORT ERROR
10347 051366 032737 020000 003024 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS UNLOADING
10348 051374 001413 BEQ 10$ ;NO, REPORT ERROR
10349 051376 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
10350 051404 113737 003101 003100 MOVB INTMSK,W.TIME ;SET TIMING ON THIS DRIVE
10351 051412 013737 003064 003114 MOV W.BSEC,W.DRV ;LOAD 8 SECONDS FOR CYCLE UP TIME
10352 051420 000137 051534 JMP I.RTRN ;RESTORE REGISTERS
10353
10354 051424 004737 052536 10$: JSR PC,R.ABNL ;REPORT ERROR
10355 051430 000137 051534 JMP I.RTRN ;RESTORE REGISTERS
10356
10357 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
10358
10359 051434 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
10360 051442 112737 000005 003000 MOVB #DR.CLR,T.CS1 ;LOAD IN DRIVE CLEAR
10361 051450 004037 051634 JSR RO,I.ISSU ;GO ISSUE DRIVE CLEAR
10362 051454 051534 I.RTRN ;ERROR RETURN
10363 051456 136437 003101 003015 BITB INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
10364 051464 001406 BEQ 15$ ;YES, CONTINUE
10365 051466 012737 000020 003052 MOV #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
10366 051474 004737 052562 JSR PC,R.CONT ;REPORT ERROR
10367 051500 000415 BR I.RTRN ;RESTORE REGISTERS
10368
10369 051502 032737 040000 003024 15$: BIT #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE RESET
10370 051510 001403 BEQ 20$ ;YES, CONTINUE
10371 051512 052765 000040 000014 BIS #DRV DSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
10372 051520 105037 003100 20$: CLRB W.TIME ;RESET TIMING ON THIS DRIVE

```

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 204
*ERROR CAUSING DRIVE TO UNLOAD

SEQ 0203

10373	051524	005037	003114	CLR	W.DRV	;CLEAR TIME COUNT
10374	051530	004737	052536	JSR	PC,R.ABNL	;REPORT ERROR
10375						
10376	051534	012600		I.RTRN: MOV	(SP)+,R0	;RESTORE R0
10377	051536	012601		MOV	(SP)+,R1	;RESTORE R1
10378	051540	012602		MOV	(SP)+,R2	;RESTORE R2
10379	051542	012603		MOV	(SP)+,R3	;RESTORE R3
10380	051544	012604		MOV	(SP)+,R4	;RESTORE R4
10381	051546	012605		MOV	(SP)+,R5	;RESTORE R5
10382	051550	000002		RTI		;RETURN
10383						

.SBTTL *CONTROLLER CLEAR ROUTINE

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

10384									
10385									
10386									
10387									
10388									
10389									
10390									
10391									
10392									
10393									
10394									
10395									
10396									
10397									
10398									
10399									
10400									
10401									
10402									
10403									
10404									
10405	051552	012762	100000	000000	I.CCLR:	MOV	#CCLR,RKCS1(R2)	; CLEAR CONTROLLER	
10406	051560	016237	000000	003000		MOV	RKCS1(R2),T.CS1	; STORE COMMAND AND STATUS REGISTER 1	
10407	051566	032737	100000	003000		BIT	#CERR,T.CS1	; CHECK IF CONTROLLER CLEAR DID	
10408								CLEAR ERROR	
10409	051574	001407				BEQ	SS	; YES, RETURN TO DRIVER PROCESSING	
10410	051576	052737	000001	003052		BIS	#E.CCLR,E.CONT	; SET CLEAR CONTROLLER DID NOT CLEAR ERROR	
10411	051604	004737	052562			JSR	PC,R.CONT	; REPORT CONTROLLER ERROR	
10412	051610	011000				MOV	(R0),R0	; SET UP ERROR RETURN	
10413	051612	000200				RTS	R0	; RETURN	
10414									
10415	051614	012762	000100	000000	SS:	MOV	#IE,RKCS1(R2)	; SET INTERRUPT ENABLE	
10416	051622	112737	177777	003074		MOVB	#-1,I.ISRL	; SET INTERRUPT ENABLE ISSUED	
10417	051630	005720				TST	(R0)+	; ADJUST FOR NORMAL RETURN	
10418	051632	000200				RTS	R0	; RETURN	

K16

.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

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

2\$: ;CHECK IF ANY CONTROLLER ERROR IS SET
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

10419
10420
10421
10422
10423
10424
10425
10426
10427
10428
10429
10430
10431
10432
10433
10434
10435
10436
10437
10438
10439
10440
10441
10442
10443
10444
10445
10446
10447 051634 013746 003000
10448 051640 005037 003002
10449 051644 116537 000000 003002
10450 051652 013762 003002 000010
10451 051660 116537 000007 003001
10452 051666 142737 177753 003001
10453
10454 051674 013762 003000 000000
10455 051702 105762 000000
10456 051706 100375
10457 051710 004737 052056
10458 051714 032737 100000 003000
10459 051722 001437
10460 051724 032737 001000 003002
10461 051732 001406
10462 051734 052737 100000 003052
10463 051742 004737 052562
10464 051746 000440
10465
10466
10467 051750 032737 024000 003000 2\$:
10468 051756 001027
10469 051760 032737 176400 003002
10470 051766 001023
10471 051770 032737 131761 003016
10472 051776 001017
10473
10474 052000 122716 000005

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 207
*COMMAND ISSUED BY DRIVER SERVICE ROUTINE

SEQ 0206

10475	052004	001003				BNE	3\$:NO, DO NOT SET DRIVE HARD ERROR
10476	052006	052765	000020	000014		BIS	#DRVHRD,P.PRST(R5)		:SET HARD DRIVE ERROR
10477	052014	004037	051552		3\$:	JSR	RO,I.CCLR		:GO ISSUE A CONTROLLER CLEAR
10478	052020	052050				10\$:ERROR RETURN
10479	052022	012762	000100	000000	5\$:	MOV	#IE,RKCS1(R2)		:SET INTERRUPT ENABLE
10480	052030	005726				TST	(SP)+		:ADJUST STACK
10481	052032	005720				TST	(RO)+		:ADJUST RO FOR NORMAL RETURN
10482	052034	000200				RTS	RO		:RETURN
10483									
10484	052036	052737	001000	003052	7\$:	BIS	#E.CERR,E.CONT		:SET CONTROLLER ERROR DURING
10485									:DRIVER SERVICING
10486	052044	004737	052562			JSR	PC,R.CONT		:REPORT ERROR
10487	052050	005726			10\$:	TST	(SP)+		:ADJUST STACK
10488	052052	011000				MOV	(RO),RO		:ADJUST RO FOR ERROR RETURN
10489	052054	000200				RTS	RO		:RETURN

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-7P 12:02 PAGE 208
*STORE RK611 UNIBUS REGISTERS

SEQ 0207

M16

.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
*
*****
```

10490				
10491				
10492				
10493				
10494				
10495				
10496				
10497				
10498				
10499				
10500				
10501				
10502				
10503				
10504				
10505				
10506				
10507	052056	016237	000000	003000
10508	052064	016237	000010	003002
10509	052072	016237	000002	003004
10510	052100	016237	000004	003006
10511	052106	016237	000006	003010
10512	052114	016237	000012	003020
10513	052122	016237	000014	003016
10514	052130	016237	000016	003014
10515	052136	016237	000020	003012
10516	052144	016237	000026	003022
10517	052152	016237	000034	003024
10518	052160	016237	000036	003026
10519	052166	016237	000030	003030
10520	052174	016237	000032	003032
10521	052202	000207		

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

10522
10523
10524
10525
10526
10527
10528
10529
10530
10531
10532
10533
10534
10535
10536
10537
10538
10539
10540
10541
10542
10543
10544
10545
10546
10547
10548
10549
10550
10551
10552
10553
10554
10555
10556
10557
10558
10559
10560
10561
10562
10563
10564
10565
10566

```
.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

.....

```

I.CSTS: BIC          #177741,P.CS1(R5)          ;CLEAR ALL BITS EXCEPT FUNCTION
                                                ;OF LAST COMMAND ISSUED
        BIC          #36,T.CS1                  ;CLEAR FUNCTION OF CS1 STATUS
        BIS          T.CS1,P.CS1(R5)            ;GENERATE CS1 STATUS INFORMATION
I.CST1: MOV          RKCS2(R2),P.CS2(R5)        ;STORE COMMAND AND STATUS REGISTER 2
        MOV          RKWC(R2),P.WCR(R5)        ;STORE WORD COUNT REGISTER
        MOV          RKBA(R2),P.BAR(R5)        ;STORE BUS ADDRESS REGISTER
        MOV          RKDA(R2),P.DTS(R5)        ;STORE DESIRED TRACK AND SECTOR
        MOV          RKDS(R2),P.DS(R5)         ;STORE DRIVE STATUS REGISTER
        MOV          RKER(R2),P.ER(R5)         ;STORE ERROR REGISTER
        MOV          RKASOF(R2),P.ASOF(R5)      ;STORE ATTENTION SUMMARY AND
                                                ;OFFSET
        MOV          RKDCYL(R2),P.DCYL(R5)      ;STORE CYLINDER ADDRESS
        RTS          PC                        ;RETURN

```

ADDRESS	DATA	INSTR	OPCODE	OPERAND	COMMENT
0593	052310	012762	000001	000026	I. STAT: MOV #1, RKMR1(R2)
0594					; LOAD MAINTENANCE REGISTER 1
0595	052316	112737	000001	003000	FOR STATUS BYTE 01
0596	052324	004037	051634		LOAD COMMAND
0597	052330	052520			JSR RO, I. ISSU
0598	052332	013765	003024	000044	3\$
0599	052340	013765	003026	000046	MOV T.MR2, P.A01(R5)
0600	052346	012762	000002	000026	MOV T.MR3, P.B01(R5)
0601					MOV #2, RKMR1(R2)
0602	052354	112737	000001	003000	; LOAD MAINTENANCE REGISTER 1
0603	052362	004037	051634		FOR STATUS BYTE 10
0604	052366	052520			LOAD COMMAND
0605	052370	013765	003024	000050	JSR RO, I. ISSU
0606	052376	013765	003026	000052	3\$
0607	052404	012762	000003	000026	MOV T.MR2, P.A10(R5)
0608					MOV T.MR3, P.B10(R5)
0609	052412	112737	000001	003000	MOV #3, RKMR1(R2)
0610	052420	004037	051634		; LOAD MAINTENANCE REGISTER
0611	052424	052520			FOR STATUS BYTE 11
0612	052426	013765	003024	000054	LOAD COMMAND
0613	052434	013765	003026	000056	JSR RO, I. ISSU
0614	052442	005062	000026		3\$
0615					MOV T.MR2, P.A11(R5)
0616	052446	112737	000001	003000	MOV T.MR3, P.B11(R5)
0617	052454	004037	051634		CLR RKMR1(R2)
0618	052460	052520			; LOAD MAINTENANCE REGISTER 1
0619	052462	013765	003024	000040	FOR STATUS BYTE 00
0620	052470	013765	003026	000042	LOAD COMMAND
0621	052476	032737	001000	003026	JSR RO, I. ISSU
0622	052504	001407			3\$
					MOV T.MR2, P.A00(R5)
					MOV T.MR3, P.B00(R5)
					BIT #5, PAR, T.MR3
					BEQ \$\$
					; CHECK IF BAD PARITY DETECTED BY DRIVE
					; NO, RETURN NORMALLY

CZR6NDO RK611/06 SS VERIF 2 MACY11 30A(1052) 10-JAN-78 12:02 PAGE 211
CZR6ND.P11 10-JAN-78 10:05 *GATHER DRIVE STATUS

10623	052506	052737	002000	003052		BIS	#E.DPAR,E.CONT	;INDICATE BAD PARITY DETECTED BY DRIVE
10624	052514	004737	052562			JSR	PC,R.CONT	;REPORT ERROR
10625	052520	011000			3\$:	MOV	(R0),R0	;LOAD R0 FOR ERROR RETURN
10626	052522	000200				RTS	R0	;RETURN
10627								
10628	052524	052765	001000	000014	5\$:	BIS	#PBSVAL,P.PRST(R5)	;SET PARAMETER BLOCK STATUS VALID
10629	052532	005720				TST	(R0),+	;ADJUST R0 FOR NORMAL RETURN
10630	052534	000200				RTS	R0	;RETURN
10631								

E01

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 212
*COMMON DRIVER RETURNS

SEQ 0211

10632				.SBTTL	*COMMON DRIVER RETURNS	
10633						
10634	052536	105037	003101	R.ABNL:	CLRB	INTMSK
10635	052542	004777	130300		JSR	PC, @A.ABNL
10636	052546	000207			RTS	PC
10637						;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10638	052550	105037	003101	R.NORM:	CLRB	INTMSK
10639	052554	004777	130264		JSR	PC, @A.NORM
10640	052560	000207			RTS	PC
10641						;INDICATE NORMAL RETURN
10642	052562	105037	003101	R.CONT:	CLRB	INTMSK
10643	052566	105037	003100		CLRB	W.TIME
10644	052572	005037	003114		CLR	W.DRV
10645	052576	004777	130246		JSR	PC, @A.CONT
10646	052602	000207			RTS	PC
						;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
						;RESET WATCH DOG TIMING ON THIS DRIVE
						;CLEAR TIMING COUNT FOR THIS DRIVE
						;INDICATE CONTROLLER ERROR RETURN
						;RETURN

.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
*****

```

```

10647
10648
10649
10650
10651
10652
10653
10654
10655
10656
10657
10658
10659
10660
10661
10662
10663
10664
10665
10666
10667
10668
10669
10670
10671
10672
10673
10674
10675
10676
10677
10678 052604 010546
10679 052606 010446
10680 052610 010346
10681 052612 010246
10682 052614 010146
10683 052616 010046
10684 052620 013746 177776
10685 052624 013737 003042 177776
10686 052632 017605 000016
10687 052636 062766 000002 000016
10688 052644 016504 000000
10689 052650 042704 177770
10690 052654 010537 003112
10691 052660 116437 003102 003101
10692 052666 116437 003102 003100
10693 052674 013737 003062 003114
10694
10695 052702 013702 003036
10696
10697
10698
10699
10700
10701
10702

```

```

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      #1C<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

```

GO1

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 214
*COMMAND INITIATOR

SEQ 0213

10703	052706	042765	075176	000014		BIC	#1C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAIL>,P.PRST(R5)	
10704								
10705	052714	010500				MOV	R5,R0	;STORE PARAMETER BLOCK ADDRESS
10706	052716	062700	000016			ADD	#P.CS1,R0	;CALCULATE FIRST LOCATION TO BE CLEARED
10707	052722	010501				MOV	R5,R1	;STORE PARAMETER BLOCK ADDRESS
10708	052724	062701	000062			ADD	#P.EPAT,R1	;CALCULATE LAST LOCATION TO BE CLEARED
10709								
10710	052730	005020			1\$:	CLR	(R0)+	;CLEAR RETURN PARAMETER
10711	052732	020001				CMP	R0,R1	;CHECK IF FINISHED
10712	052734	101775				BLOS	1\$;NO, CLEAR NEXT RETURN PARAMETER
10713	052736	105037	003074			CLRB	I.ISRL	;CLEAR RELEASE OR INTERRUPT ISSUED
10714	052742	010465	000020			MOV	R4,P.CS2(R5)	;STORE DRIVE NUMBER
10715	052746	005062	000026			CLR	RKMRI(R2)	;CLEAR RK06 MAINTENANCE REGISTER 1
10716	052752	132765	000040	000001		BITB	#BITS,P.CMND(R5)	;CHECK IF SPECIAL COMMAND
10717	052760	001402				BEQ	3\$;NO, PROCESS
10718	052762	000137	053476			JMP	C.SPEC	;JUMP TO SPECIAL COMMAND PROCESSOR
10719								
10720	052766	122765	000107	000001	3\$:	CMPB	#UNLOAD,P.CMND(R5)	;CHECK IF POSITIONING COMMAND
10721								START SPINDLE
10722								RECALIBRATE
10723								OFFSET
10724								SEEK
10725								UNLOAD
10726								
10727	052774	101174				BHI	25\$;NO, DRIVE COMMAND
10728								SELECT DRIVE
10729								PACK ACKNOWLEDGE
10730								CLEAR
10731								
10732	052776	122765	000117	000001		CMPB	#SEEK,P.CMND(R5)	;CHECK IF DATA TRANSFER
10733	053004	103540				BLO	20\$;YES, DATA TRANSFER COMMAND
10734								READ DATA
10735								WRITE DATA
10736								READ HEADER
10737								WRITE HEADER
10738								WRITE CHECK
10739	053006	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2)	;LOAD DRIVE NUMBER
10740	053014	052765	000002	000014		BIS	#DRVPOS,P.PRST(R5)	;SET DRIVE POSITIONING
10741	053022	005037	003054			CLR	O.WAIT	;CLEAR WAIT FOR COMMAND
10742	053026	122765	000117	000001		CMPB	#SEEK,P.CMND(R5)	;CHECK IF SEEK
10743	053034	001007				BNE	5\$;NO, CHECK FOR OFFSET
10744	053036	016562	000002	000020		MOV	P.CYLN(R5),RKDCYL(R2)	;LOAD CYLINDER ADDRESS
10745	053044	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	;LOAD SECTOR AND TRACK
10746	053052	000431				BR	8\$;GO ISSUE COMMAND
10747								
10748	053054	122765	000115	000001	5\$:	CMPB	#OFFSET,P.CMND(R5)	;CHECK IF OFFSET
10749	053062	001007				BNE	6\$;NO, CHECK FOR UNLOAD
10750	053064	116565	000006	000032		MOVB	P.OFST(R5),P.ASOF(R5)	;STORE OFFSET
10751	053072	016562	000032	000016		MOV	P.ASOF(R5),RKASOF(R2)	;LOAD OFFSET REGISTER
10752	053100	000416				BR	8\$;GO ISSUE COMMAND
10753								
10754	053102	122765	000111	000001	6\$:	CMPB	#SRTSPL,P.CMND(R5)	;CHECK IF START SPINDLE
10755	053110	001003				BNE	7\$;NO, CHECK IF RECAL
10756	053112	013737	003066	003114		MOV	W.MIN,W.DRV	;LOAD WATCH DOG TIME FOR 1 MINUTE
10757	053120	122765	000113	000001	7\$:	CMPB	#RECAL,P.CMND(R5)	;CHECK IF RECAL
10758	053126	001003				BNE	8\$;NO, CONTINUE

```

10759 053130 013737 003064 003114      MOV      W.BSEC,W.DRV      ;LOAD RECAL TIME FOR 8 SECONDS
10760 053136 116565 000007 000017 8$:      MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10761 053144 042765 165777 000016      BIC      #1<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10762                                     ; AND DRIVE TYPE
10763 053152 116565 000001 000016      MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10764 053160 042765 000200 000014      BIC      #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10765 053166 032765 000400 000014      BIT      #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10766 053174 001533 30$      BEQ      30$      ;NO SKIP CLEAR OF INTERRUPT ENABLE
10767 053176 042765 000100 000016      BIC      #IE,P.CS1(R5) ;CLEAR INTERRUPT ENABLE
10768 053204 016562 000016 000000      MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10769 053212 004737 047154 10$:      JSR      PC.W.WTCH ;CALL WATCH DOG TIMER
10770 053216 016237 000000 003000      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
10771 053224 032737 000200 003000      BIT      #RDY,T.CS1 ;WAIT FOR READY
10772 053232 001767 10$      BEQ      10$
10773 053234 032737 100000 003000      BIT      #CERR,T.CS1 ;CHECK FOR ERROR
10774 053242 001011 15$      BNE      15$      ;YES, GIVE NORMAL RETURN
10775 053244 004737 047154 11$:      JSR      PC.W.WTCH ;CALL WATCH DOG TIMER
10776 053250 016237 000016 003014      MOV      RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10777 053256 133737 003101 003015      BITB     INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
10778 053264 001767 11$      BEQ      11$      ;WAIT FOR DRIVE INTERRUPT
10779 053266 105037 003100 15$:      CLRB     W.TIME ;RESET TIMING ON THIS DRIVE
10780 053272 005037 003114      CLR      W.DRV ;CLEAR DRIVE TIMING COUNT
10781 053276 004737 052550      JSR      PC.R.NORM ;INDICATE COMMAND IS FINISHED
10782 053302 000137 054456      JMP      C.ATRN ;RESTORE REGISTERS
10783
10784 053306 016562 000010 000004 20$:      MOV      P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
10785 053314 016562 000012 000002      MOV      P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
10786 053322 016562 000002 000020      MOV      P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
10787 053330 016562 000004 000006      MOV      P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
10788 053336 122765 000131 000001      CMPB     #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
10789 053344 001010 25$      BNE      25$      ;NO, GO ISSUE THE COMMAND
10790 053346 032765 000200 000014      BIT      #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
10791 053354 001404 25$      BEQ      25$      ;NO, GO ISSUE THE COMMAND
10792 053356 012765 000123 000016      MOV      #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
10793 053364 000406 26$      BR       26$      ;GO ISSUE COMMAND
10794
10795 053366 116565 000001 000016 25$:      MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10796 053374 042765 000200 000014      BIC      #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10797 053402 116565 000007 000017 26$:      MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10798 053410 142765 177750 000017      BICB     #1<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
10799                                     ; FORMAT, DRIVE TYPE, AND BUS ADDRESS
10800                                     ; BITS 16-17
10801 053416 010537 003054      MOV      R5,0.WAIT ;LOAD WAITING FOR COMMAND
10802 053422 032765 100000 000014      BIT      #DBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
10803 053430 001403 27$      BEQ      27$      ;NO, LOAD CS2
10804 053432 052765 000020 000020      BIS      #BAI,P.CS2(R5) ;SET INHIBIT BUS ADDRESS INCREMENT
10805 053440 016562 000020 000010 27$:      MOV      P.CS2(R5),RKCS2(R2) ;LOAD CS2
10806 053446 032765 000400 000014      BIT      #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10807 053454 001403 30$      BEQ      30$      ;NO SKIP CLEAR OF INTERRUPT ENABLE
10808 053456 042765 000100 000016      BIC      #IE,P.CS1(R5) ;CLEAR INTERRUPT ENABLE
10809 053464 016562 000016 000000 30$:      MOV      P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10810 053472 000137 054456      JMP      C.ATRN ;RESTORE REGISTERS
10811
10812 .SBTTL *SPECIAL COMMAND PROCESSING
10813
10814 053476 122765 000141 000001 C.SPEC: CMPB     #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS

```


10815	053504	001132			BNE	10\$:NO, PROCESS OTHER COMMANDS
10816	053506	016562	000020	000010	MOV	P.CS2(R5),RKCS1(R2)	:LOAD CS2 FOR COMMAND
10817	053514	116565	000007	000017	MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10818	053522	042765	165777	000016	BIC	*C<CFMT!COT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT
10819							: AND DRIVE TYPE
10820	053530	112765	000001	000016	MOVB	*DR.SEL,P.CS1(R5)	:STORE COMMAND
10821	053536	016562	000016	000000	MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
10822	053544	004737	047154		JSR	PC.W.WTCH	:CALL WATCH-DOG TIMER
10823	053550	016265	000000	000016	MOV	RKCS1(R2),P.CS1(R5)	:STORE COMMAND AND STATUS REG. 1
10824	053556	032765	000200	000016	BIT	*RDY,P.CS1(R5)	:WAIT FOR READY
10825	053564	001767			BEQ	2\$	
10826	053566	004737	052226		JSR	PC.I.CST1	:STORE CONTROLLER REGISTERS
10827	053572	016265	000034	000040	MOV	RKMR2(R2),P.A00(R5)	:STORE STATUS BYTE 00 MESSAGE A
10828	053600	016265	000036	000042	MOV	RKMR3(R2),P.B00(R5)	:STORE STATUS BYTE 00 MESSAGE B
10829	053606	032765	100000	000016	BIT	*CERR,P.CS1(R5)	:CHECK IF CONTROLLER ERROR
10830	053614	001436			BEQ	6\$:NO, GATHER DRIVE STATUS
10831	053616	105037	003100		CLRB	W.TIME	:RESET WATCH DOG TIMING ON THIS DRIVE
10832	053622	005037	003114		CLR	W.DRV	:CLEAR WATCH DOG COUNT
10833	053626	032765	000400	000014	BIT	*NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
10834	053634	001043			BNE	8\$:YES, INDICATE NORMAL RETURN
10835	053636	032765	001000	000020	BIT	*MDS,P.CS2(R5)	:CHECK IF MULTIPLE DRIVE SELECT
10836	053644	001043			BNE	9\$:YES, INDICATE CONTROLLER ERROR
10837	053646	004037	051552		JSR	RO,I.CCLR	:CLEAR ERROR
10838	053652	054456			C.RTRN		:ERROR RETURN
10839	053654	032765	010400	000020	BIT	*NED!UFE,P.CS2(R5)	:CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
10840	053662	001007			BNE	5\$:REPORT ERROR
10841	053664	032765	000001	000036	BIT	*DRA,P.DS(R5)	:CHECK IF DRIVE AVAILIABLE
10842	053672	001003			BNE	5\$:YES, REPORT ERROR
10843	053674	052765	010000	000014	BIS	*DRVSZD,P.PRST(R5)	:INDICATE DRIVE IS SEIZED BY OTHER PORT
10844	053702	004737	052536		JSR	PC.R.ABNL	:INDICATE ABNORMAL RETURN
10845	053706	000137	054456		JMP	C.RTRN	:RESTORE REGISTERS
10846							
10847	053712	004037	052310		JSR	RO,I.STAT	:GATHER DRIVE STATUS
10848	053716	054456			C.RTRN		:ERROR RETURN
10849	053720	105037	003100		CLRB	W.TIME	:STOP WATCH-DOG TIMING ON DRIVE
10850	053724	005037	003114		CLR	W.DRV	:RESET WATCH-DOG TIME
10851	053730	032765	000400	000014	BIT	*NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
10852	053736	001402			BEQ	8\$:NO, REPORT ERROR
10853	053740	005062	000000		CLR	RKCS1(R2)	:CLEAR INTERRUPT ENABLE
10854	053744	004737	052550		JSR	PC.R.NORM	:REPORT COMMAND COMPLETE
10855	053750	000137	054456		JMP	C.RTRN	:RESTORE REGISTERS
10856							
10857	053754	052737	100000	003052	BIS	*E.MDS,E.CONT	:SET MULTIPLE DRIVE SELECT
10858	053762	004737	052562		JSR	PC.R.CONT	:INDICATE CONTROLLER ERROR
10859	053766	000137	054456		JMP	C.RTRN	
10860							
10861	053772	122765	000140	000001	CMPB	*RELEASES,P.CMND(R5)	:CHECK IF RELEASE COMMAND
10862	054000	001040			BNE	13\$:NO, CHECK IF READ ALL HEADERS
10863	054002	010537	003054		MOV	R5,O.WAIT	:STORE PARAMETER BLOCK ADDRESS IN
10864							: WAIT FOR COMMAND
10865	054006	052765	000010	000020	BIS	*RLS,P.CS2(R5)	:SET RELEASE BIT
10866	054014	016562	000020	000010	MOV	P.CS2(R5),RKCS2(R2)	:LOAD CS2 FOR DESELECT
10867	054022	112737	000001	003074	MOVB	*I.I.ISRL	:SET FLAG FOR RELEASE COMMAND
10868	054030	116565	000007	000017	MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10869	054036	042765	165777	000016	BIC	*C<CFMT!COT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT
10870							: AND DRIVE TYPE

10871	054044	112765	000101	000016		MOVB	#SELDV,P.CS1(R5) ;STORE COMMAND
10872	054052	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10873	054060	001403				BEQ	11\$;NO, DO NOT RESET INTERRUPT ENABLE
10874	054062	042765	000100	000016		BIC	#IE,P.CS1(R5) ;RESET INTERRUPT ENABLE
10875	054070	016562	000016	000000	11\$:	MOV	P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10876	054076	000137	054456			JMP	C.RTRN ;RESTORE REGISTERS
10877							
10878	054102	122765	000164	000001	13\$:	CMPB	#RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
10879	054110	001053				BNE	30\$;NO, CHECK IF CONTROLLER CLEAR
10880	054112	010537	003054			MOV	R5,0.WAIT ;SET WAITING FOR COMMAND COMPLETION
10881	054116	016537	000010	003070		MOV	P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
10882	054124	132765	000020	000007		BITB	#B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
10883	054132	001404				BEQ	14\$;YES, LOAD 22 IN HEADER COUNT
10884	054134	012737	000024	003072		MOV	#20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
10885	054142	000403				BR	22\$;GO ISSUE READ HEADER COMMAND
10886							
10887	054144	012737	000026	003072	14\$:	MOV	#22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
10888	054152	016562	000002	000020	22\$:	MOV	P.CYL(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10889	054160	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
10890	054166	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10891	054174	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10892	054202	042765	165777	000016		BIC	#1C(CFMT!CDT),P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
10893							AND FORMAT
10894	054210	112765	000125	000016		MOVB	#RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
10895	054216	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10896	054224	001027				BNE	34\$;YES, INDICATE ILLEGAL DRIVER COMMAND
10897	054226	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
10898	054234	000137	054456			JMP	C.RTRN ;RESTORE REGISTERS
10899							
10900	054240	122765	000176	000001	30\$:	CMPB	#CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
10901	054246	001012				BNE	32\$;NO, CHECK IF SUBSYSTEM CLEAR
10902	054250	004037	051552			JSR	RD,I.CCLR ;CLEAR CONTROLLER
10903	054254	054456				C.RTRN	ERROR RETURN
10904	054256	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10905	054264	001472				BEQ	40\$;NO, INDICATE NORMAL RETURN
10906	054266	005062	000000			CLR	RKCS1(R2) ;RESET INTERRUPT ENABLE
10907	054272	000467				BR	40\$;INDICATE NORMAL RETURN
10908							
10909	054274	122765	000177	000001	32\$:	CMPB	#SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
10910	054302	001406				BEQ	36\$;YES, CLEAR SUBSYSTEM
10911	054304	052737	000100	003052	34\$:	BIS	#E.ILLD,E.CONT ;SET ILLEGAL DRIVER COMMAND
10912	054312	004737	052562			JSR	PC,R.CONT ;REPORT ERROR
10913	054316	000457				BR	C.RTRN ;RESTORE REGISTERS
10914							
10915	054320	012762	000040	000010	36\$:	MOV	#SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
10916	054326	016265	000000	000016		MOV	RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
10917	054334	032765	100000	000016		BIT	#CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
10918	054342	001406				BEQ	37\$;NO, FINISH COMMAND
10919	054344	052737	000001	003052		BIS	#BIT0,E.CONT ;SET CLEAR SUBSYSTEM DID NOT CLEAR
10920							CONTROLLER ERROR
10921	054352	004737	052562			JSR	PC,R.CONT ;REPORT ERROR
10922	054356	000437				BR	C.RTRN ;RESTORE REGISTERS
10923							
10924	054360	013746	003060		37\$:	MOV	W.MILI,-(SP) ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
10925							TO DISAPPEAR
10926	054364	016265	000000	000016	38\$:	MOV	RKCS1(R2),P.CS1(R5) ;STORE CS1

K01

CZR6ND0 PK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 218
*SPECIAL COMMAND PROCESSING

SEQ 0217

```

10927 054372 032765 040000 000016      BIT      #DI,P.CS1(R5)      ;CHECK IF ATTENTIONS CLEARED
10928 054400 001411                      BEQ      39$              ;YES, FINISH COMMAND
10929 054402 005316                      DEC      (SP)          ;DECREMENT 16 MILLISECOND COUNT
10930 054404 001367                      BNE      38$              ;CHECK DRIVE INTERRUPT AGAIN
10931 054406 005726                      TST      (SP)+          ;ADJUST STACK
10932 054410 052737 000040 003052      BIS      #E.SCLR,E.CONT ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10933                                ;DRIVE ATTENTIONS
10934 054416 004737 052562                      JSR      PC,R.CONT      ;REPORT ERROR
10935 054422 000415                      BR       C.RTRN        ;RESTORE REGISTER
10936
10937 054424 005726 000400 000014 39$:    TST      (SP)+          ;ADJUST STACK
10938 054426 032765                      BIT      #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10939 054434 001010                      BNE      C.RTRN        ;YES, RESTORE REGISTERS
10940 054436 112737 177777 003074      MOVB     #-1,I.ISRL    ;SET INTERRUPT ENABLE SET
10941 054444 012762 000100 000000      MOV      #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
10942 054452 004737 052550 40$:    JSR      PC,R.NORM      ;INDICATE NORMAL TERMINATION
10943
10944 054456 012637 177776      C.RTRN: MOV      (SP)+,PS      ;RESTORE PSW
10945 054462 012600                      MOV      (SP)+,R0      ;RESTORE R0
10946 054464 012601                      MOV      (SP)+,R1      ;RESTORE R1
10947 054466 012602                      MOV      (SP)+,R2      ;RESTORE R2
10948 054470 012603                      MOV      (SP)+,R3      ;RESTORE R3
10949 054472 012604                      MOV      (SP)+,R4      ;RESTORE R4
10950 054474 012605                      MOV      (SP)+,R5      ;RESTORE R5
10951 054476 000207                      RTS       PC           ;RETURN
10952
10953      .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
10954      ;*****
10955      ;
10956      ; THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10957      ; WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL
10958      ; IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10959      ; ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HI0CT.
10960      ;
10961      ;CALL
10962      ;
10963      ; MOV      <ADDRESS OF ASCII STRING>,-(SP)
10964      ; JSR      PC,OCTBIN
10965      ; <ADDRESS OF ERROR RETURN>
10966      ; RETURN
10967      ;*****
10968
10969 054500 010046      OCTBIN: MOV      R0,-(SP)      ;SAVE R0
10970 054502 010146                      MOV      R1,-(SP)      ;SAVE R1
10971 054504 010246                      MOV      R2,-(SP)      ;SAVE R2
10972 054506 016600 000010      MOV      10(SP),R0    ;GET ADDRESS OF ASCII STRING
10973 054512 005001                      CLR      R1           ;CLEAR DATA WORDS
10974 054514 005002                      CLR      R2
10975 054516 112046 2$:    MOVB     (R0)+,-(SP)    ;PICK THIS CHARACTER
10976 054520 001423                      BEQ      3$           ;IF ZERO GET OUT
10977 054522 121627 000054      CMPB     (SP),#',      ;CHECK IF COMMA
10978 054526 001420                      BEQ      3$           ;IF COMMA GET OUT
10979 054530 122716 000060      CMPB     #'0,(SP)     ;MAKE SURE THIS CHARACTER IS
10980 054534 003030                      BGT      4$           ; AN OCTAL DIGIT
10981 054536 12271E 000067      CMPB     #'7,(SP)
10982 054542 002425                      BLT      4$

```

L01

CZR6NDD RK611/06 SS VERIF 2
CZR6'D.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 219
OCTAL TO BINARY CONVERSION ROUTINE

SEQ 0218

10983	054544	006301	ASL	R1	; *2
10984	054546	006102	ROL	R2	
10985	054550	006301	ASL	R1	; *4
10986	054552	006102	ROL	R2	
10987	054554	006301	ASL	R1	; *8
10988	054556	006102	ROL	R2	
10989	054560	042716	BIC	#1C7,(SP)	; STRIP THE ASCII JUNK
10990	054564	062601	ADD	(SP)+,R1	; ADD THIS DIGIT
10991	054566	000753	BR	2\$; LOOP
10992	054570	005726	3\$: TST	(SP)+	; CLEAN PARTIAL FROM STACK
10993	054572	010166	MOV	R1,10(SP)	; SAVE RESULT
10994	054576	010237	MOV	R2,\$SHIOCT	
10995	054602	012602	MOV	(SP)+,R2	; RESTORE R2
10996	054604	012601	MOV	(SP)+,R1	; RESTORE R1
10997	054606	012600	MOV	(SP)+,R0	; RESTORE R0
10998	054610	062716	ADD	#2,(SP)	; ADJUST RETURN
10999	054614	000207	RTS	PC	; RETURN
11000					
11001	054616	005726	4\$: TST	(SP)+	; CLEAN UP PARTIAL FROM STACK
11002	054620	012602	MOV	(SP)+,R2	; RESTORE R2
11003	054622	012601	MOV	(SP)+,R1	; RESTORE R1
11004	054624	012600	MOV	(SP)+,R0	; RESTORE R0
11005	054626	013616	MOV	2(SP)+,(SP)	; PUT ADDRESS OF ERROR ROUTINE ON STACK
11006	054630	000207	RTS	PC	; GO PROCESS ERROR
11007	054632	000000	\$SHIOCT: .WORD	0	; HIGH ORDER BITS GO HERE
11008			.SBTTL	RANDOM NUMBER GENERATOR ROUTINE	
11009					
11010					
11011					
11012					
11013					
11014					
11015					
11016					
11017					
11018					
11019	054634				
11020	054634	010046	\$RAND: MOV	R0,-(SP)	; PUSH R0 ON STACK
11021	054636	010146	MOV	R1,-(SP)	; PUSH R1 ON STACK
11022	054640	010246	MOV	R2,-(SP)	; PUSH R2 ON STACK
11023	054642	013700	MOV	\$LONUM,R0	; SET R0 WITH LOW
11024	054646	013701	MOV	\$SHINUM,R1	; SET R1 WITH HIGH
11025	054652	012702	MOV	#-7,R2	; SET SHIFT COUNT
11026	054656	006300	1\$: ASL	R0	; SHIFT R0 LEFT AND
11027	054660	006101	ROL	R1	; ROTATE CARRY INTO R1 AND
11028	054662	005202	INC	R2	; CHECK FOR DONE
11029	054664	001374	BNE	1\$; CONTINUE SHIFT LOOP
11030	054666	063700	ADD	\$LONUM,R0	; ADD NUMBER TO MAKE X 129
11031	054672	005501	ADC	R1	; PROPOGATE CARRY
11032	054674	063701	ADD	\$SHINUM,R1	; ADD NUMBER TO MAKE X 129
11033	054700	062700	ADD	#1057,R0	; ADD LOW CONSTANT
11034	054704	005501	ADC	R1	; PROPOGATE CARRY
11035	054706	062701	ADD	#47401,R1	; ADD HIGH CONSTANT
11036	054712	010037	MOV	R0,\$LONUM	; SAVE R0
11037	054716	010137	MOV	R1,\$SHINUM	; SAVE R1
11038	054722	012602	MOV	(SP)+,R2	; POP STACK INTO R2

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 220
RANDOM NUMBER GENERATOR ROUTINE

SEQ 0219

```

11039 054724 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11040 054726 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
11041 054730 000207      RTS      PC          ;;RETURN
11042 054732 176543      $HINUM: .WORD 176543
11043 054734 123456      $LONUM: .WORD 123456
11044                                .SBTTL  TYPE ROUTINE
11045
11046                                *****
11047                                *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11048                                *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11049                                *NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11050                                *NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11051                                *NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11052                                *
11053                                *CALL:
11054                                *1) USING A TRAP INSTRUCTION
11055                                *      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11056                                *OR
11057                                *      TYPE
11058                                *      MESADR
11059                                *
11060
11061 054736 105737 001157      $TYPE:  TSTB      $TFPLG      ;; IS THERE A TERMINAL?
11062 054742 100002          BPL      1$          ;; BR IF YES
11063 054744 000000          HALT          ;; HALT HERE IF NO TERMINAL
11064 054746 000430          BR      3$          ;; LEAVE
11065 054750 010046      1$:  MOV      R0,-(SP)      ;; SAVE R0
11066 054752 017600      MOV      22(SP),R0      ;; GET ADDRESS OF ASCIZ STRING
11067 054756 122737 000001 001340      CMPB      #APTENV,$ENV      ;; RUNNING IN APT MODE
11068 054764 001011      BNE      62$          ;; NO GO CHECK FOR APT CONSOLE
11069 054766 132737 000100 001341      BITB      #APTSP00L,$ENVM      ;; SPOOL MESSAGE TO APT
11070 054774 001405      BEQ      62$          ;; NO GO CHECK FOR CONSOLE
11071 054776 010037 055006      MOV      R0,61$      ;; SETUP MESSAGE ADDRESS FOR APT
11072 055002 004737 057434      JSR      PC,$ATY3      ;; SPOOL MESSAGE TO APT
11073 055006 000000      61$:  .WORD 0          ;; MESSAGE ADDRESS
11074 055010 132737 000040 001341      62$:  BITB      #APTC SUP,$ENVM      ;; APT CONSOLE SUPPRESSED
11075 055016 001003      BNE      60$          ;; YES, SKIP TYPE OUT
11076 055020 112046      2$:  MOVB      (R0)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
11077 055022 001005      BNE      4$          ;; BR IF IT ISN'T THE TERMINATOR
11078 055024 005726      TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
11079 055026 012600      60$:  MOV      (SP)+,R0      ;; RESTORE R0
11080 055030 062716 000002      3$:  ADD      #2,(SP)      ;; ADJUST RETURN PC
11081 055034 000002      RTI          ;; RETURN
11082 055036 122716 000011      4$:  CMPB      #HT,(SP)      ;; BRANCH IF <HT>
11083 055042 001430      BEQ      8$          ;; BRANCH IF NOT <CRLF>
11084 055044 122716 000200      CMPB      #CRLF,(SP)
11085 055050 001006      BNE      5$          ;; POP <CR><LF> EQUIV
11086 055052 005726      TST      (SP)+      ;; TYPE A CR AND LF
11087 055054 104401      TYPE
11088 055056 001315      $CRLF
11089 055060 105037 055214      CLRB      $CHARCNT      ;; CLEAR CHARACTER COUNT
11090 055064 000755      BR      2$          ;; GET NEXT CHARACTER
11091 055066 004737 055150      5$:  JSR      PC,$TYPEC      ;; GO TYPE THIS CHARACTER
11092 055072 123726 001156      6$:  CMPB      $FILLC,(SP)+      ;; IS IT TIME FOR FILLER CHARS.?
11093 055076 001350      BNE      2$          ;; IF NO GO GET NEXT CHAR.
11094 055100 013746 001154      MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED

```

```

11095
11096 055104 105366 000001      7$:   DECB   1(SP)      ;; AND THE NULL CHAR.
11097 055110 002770              BLT    6$          ;; DOES A NULL NEED TO BE TYPED?
11098 055112 004737 055150      JSR    PC,$TYPEC      ;; BR IF NO--GO POP THE NULL OFF OF STACK
11099 055116 105337 055214      DECB   $CHARCNT      ;; GO TYPE A NULL
11100 055122 000770              BR     7$          ;; DO NOT COUNT AS A COUNT
11101
11102
11103      ;HORIZONTAL TAB PROCESSOR
11104 055124 112716 000040      8$:   MOVB   #' (SP)      ;; REPLACE TAB WITH SPACE
11105 055130 004737 055150      9$:   JSR    PC,$TYPEC      ;; TYPE A SPACE
11106 055134 132737 000007 055214 BITB   #7,$CHARCNT      ;; BRANCH IF NOT AT
11107 055142 001372              BNE    9$          ;; TAB STOP
11108 055144 005726              TST    (SP)+        ;; POP SPACE OFF STACK
11109 055146 000724              BR     2$          ;; GET NEXT CHARACTER
11110 055150 105777 123774      $TYPEC: TSTB   2$TPS      ;; WAIT UNTIL PRINTER IS READY
11111 055154 100375              BPL    $TYPEC
11112 055156 116677 000002 123766 MOVB   2(SP),2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
11113 055164 122766 000015 000002 CMPB   #CR,2(SP)      ;; IS CHARACTER A CARRIAGE RETURN?
11114 055172 001003              BNE    1$          ;; BRANCH IF NO
11115 055174 105037 055214      CLRB   $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
11116 055200 000406              BR     $TYPEX
11117 055202 122766 000012 000002 1$:   CMPB   #LF,2(SP)      ;; IS CHARACTER A LINE FEED?
11118 055210 001402              BEQ    $TYPEX      ;; BRANCH IF YES
11119 055212 105227              INCB   (PC)+        ;; COUNT THE CHARACTER
11120 055214 000000      $CHARCNT: .WORD 0      ;; CHARACTER COUNT STORAGE
11121 055216 000207      $TYPEX: RTS    PC
11122
11123
11124
11125
11126
11127      ;*****
11128      ;SBTTL DOUBLE-PRECISION MULTIPLY SUBROUTINE
11129      ;SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
11130      ;USES ALL REGISTERS (R0-R5)
11131      ;
11132      ;ENTER WITH JSR PC,M.DPIM
11133      ;MULTIPLIER IN R2-R3
11134      ;MULTIPLICAND IN R4-R5
11135      ;PRODUCT RETURNED IN R0-R1-R2-R3
11136      ;*****
11137 055220 005000      M.DPIM: CLR    R0          ;CLEAR HI ORDER WORDS
11138 055222 005001      CLR    R1
11139 055224 012746 000041      MOV    #41,-(SP)      ;MOVE 33 (DEC) TO COUNTER
11140 055230 006000      M.DP01: ROR    R0
11141 055232 006001      ROR    R1
11142 055234 006002      ROR    R2          ;SHIFT TO ADD
11143 055236 006003      ROR    R3
11144 055240 103003      BCC    M.DP02      ;NO CARRY NO ADD
11145 055242 060501      ADD    R5,R1
11146 055244 005500      ADC    R0          ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
11147 055246 C60400      ADD    R4,R0      ;PRODUCT
11148 055250 005316      M.DP02: DEC    2$P          ;DECREMENT COUNTER
11149 055252 001366      BNE    M.DP01
11150 055254 005726      TST    (SP)+        ;REMOVE THE COUNTER

```

```

11151 055256 000207          RTS      PC
11152
11153
11154          ;*****
11155          ;SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
11156          ;      SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
11157          ;      USES ALL REGISTERS (R0-R5)
11158          ;
11159          ;      ENTER WITH JSR  PC,M.DPID
11160          ;      DIVIDEND IN R0-R1-R2-R3
11161          ;      DIVISOR IN R4-R5
11162          ;      REMAINDER RETURNED IN R0-R1
11163          ;      QUOTIENT RETURNED IN R2-R3
11164          ;*****
11165          ;*****
11166 055260 012746 000040 M.DPID: MOV     #40,-(SP)      ;COUNTER FOR DIVISION CYCLES
11167 055264 010446          MOV     R4,-(SP)      ;HI ORDER
11168 055266 010546          MOV     R5,-(SP)      ;LO ORDER DIVISOR TO THE STACK
11169 055270 005466 000002          NEG     2(SP)      ;FORM NEGATIVE
11170 055274 005416          NEG     @SP          ;      VERSION OF THE DIVISOR
11171 055276 005666 000002          SBC     2(SP)
11172 055302 061601          ADD     @SP,R1
11173 055304 005500          ADC     R0          ;PERFORM THE INITIAL SUBTRACTION
11174 055306 066600 000002          ADD     2(SP),R0
11175 055312 103445          BCS     M.DP50      ;IF CARRY THEN OVERFLOW HAS OCCURRED
11176 055314 005046          CLR     -(SP)      ;THIS IS A LONGER LASTING CARRY BIT
11177 055316 006103 M.DP40: ROL     R3
11178 055320 006102          ROL     R2
11179 055322 006101          ROL     R1
11180 055324 006100          ROL     R0
11181 055326 075716          TST     @SP          ;TEST "CARRY INDICATOR"
11182 055330 001410          BEQ     M.DP41      ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
11183 055332 005016          CLR     @SP          ;CLEAR UP FOR NEXT TIME
11184 055334 066601 000002          ADD     2(SP),R1
11185 055340 005500          ADC     R0          ;ADD -(DIVISOR)
11186 055342 005516          ADC     @SP          ;SET "CARRY"
11187 055344 066600 000004          ADD     4(SP),R0 ; I
11188 055350 000404          BR      M.DP42      ; I
11189 055352 060501 M.DP41: ADD     R5,R1          ;ADD +(DIVISOR)
11190 055354 005500          ADC     R0          ;SET "CARRY"
11191 055356 005516          ADC     @SP          ; I
11192 055360 060400          ADD     R4,R0          ; I
11193 055362 005516 M.DP42: ADC     @SP          ;SET "CARRY"
11194 055364 005716          TST     @SP          ;TEST THE UPDATE INDICATOR
11195 055366 001401          BEQ     .+4          ;IF ZERO FORGET IT
11196 055370 005203          INC     R3          ;NO CARRY POSSIBLE HERE
11197 055372 005366 000006          DEC     6(SP)      ;DECREMENT COUNTER
11198 055376 003347          BGT     M.DP40      ;BR IF MORE TO DO
11199 055400 006003          ROR     R3
11200 055402 103404          BCS     M.DP44
11201 055404 060501          ADD     R5,R1
11202 055406 005500          ADC     R0
11203 055410 060400          ADD     R4,R0
11204 055412 000241          CLC
11205 055414 006103 M.DP44: ROL     R3
11206 055416 062706 000010          ADD     #10,SP      ;ADJUST STACK BY 4 WORDS

```

11207 055422 000242
11208 055424 000207
11209 055426 062706 000006
11210 055432 000262
11211 055434 000207
11212
11213
11214
11215
11216
11217
11218
11219
11220
11221
11222
11223
11224
11225
11226 055436 104407
11227 055440 016601 000002
11228 055444 012705 055555
11229 055450 012704 000014
11230 055454 012703 177770
11231 055460 012100
11232 055462 012101
11233 055464 005002
11234 055466 110245
11235 055470 010002
11236 055472 005304
11237 055474 003007
11238 055476 001405
11239 055500 005205
11240 055502 010566 000002
11241 055506 104410
11242 055510 000207
11243 055512 006203
11244 055514 006001
11245 055516 006000
11246 055520 006001
11247 055522 006000
11248 055524 006001
11249 055526 006000
11250 055530 040302
11251 055532 062702 000060
11252 055536 000753
11253 055540 000016
11254
11255
11256
11257
11258
11259
11260
11261
11262

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

.SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

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

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

\$DB20: SAVREG ;; SAVE ALL REGISTERS
MOV 2(SP), R1 ;; PICKUP THE POINTER TO LOW WORD
MOV #S\$OCTVL+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
BIC R3, R2 ;; MASK OUT ALL JUNK
ADD #10, R2 ;; MAKE THIS CHAR. ASCII
BR 1\$;; GO PUT IT IN THE DATA TABLE
\$OCTVL: .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

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 224
DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

SEQ 0223

```

11263          ;*      RETURN          ;; THE FIRST ADDRESS OF ASCII
11264          ;; IS ON THE STACK
11265
11266          $DB20:  SAVREG          ;; SAVE REGISTERS
11267          MOV     2(SP),R2          ;; PICKUP THE DATA POINTER
11268          055556 016602 000002      MOV     $DECVL,R0          ;; GET ADDRESS OF "$DECVL" STRING
11269          055564 012700 055736      MOV     R0,2(SP)          ;; PUT ADDRESS OF ASCII STRING ON STACK
11270          055570 010066 000002      MOV     (R2)+,R1          ;; PICKUP THE BINARY NUMBER
11271          055574 012201              MOV     (R2)+,R2
11272          055576 012202              MOV     #10,R3          ;; SET UP TO DO 10 CONVERSIONS
11273          055600 012737 000012 055654 MOV     $STNPWR,R4          ;; ADDRESS OF TEN POWER
11274          055606 012704 055666      MOV     $STNPWR+2,R5
11275          055612 012705 055670      CLR     R3          ;; CLEAR PARTIAL
11276          055616 005003              SUB     (R4),R1          ;; SUBTRACT TEN POWER
11277          055620 161401              SBC     R2
11278          055622 005602              SUB     (R5),R2
11279          055624 161502              BLT     R3          ;; BR IF TEN POWER TOO LARGE
11280          055626 002402              INC     R3          ;; ADD 1 TO PARTIAL
11281          055630 005203              BR     2$          ;; LOOP
11282          055632 000772              ADD     (R4)+,R1          ;; RESTORE SUBTRACTED VALUE
11283          055634 062401              ADC     R2
11284          055636 005502              ADD     (R4)+,R2
11285          055640 062402              CMP     (R5)+,(R5)+          ;; MOVE TO NEXT TEN POWER
11286          055642 022525              BIS     #0,R3          ;; CHANGE PARTIAL TO ASCII
11287          055644 052703 000060      MOVB    R3,(R0)+          ;; SAVE IT
11288          055650 110320              DEC     (PC)+          ;; DONE?
11289          055652 005327              .WORD   0
11290          055654 000000              BNE     1$          ;; BR IF NO
11291          055656 001357              CLRB   (R0)+          ;; TERMINATOR
11292          055660 105020              RESREG          ;; RESTORE REGISTERS
11293          055662 104410              RTS     PC          ;; RETURN
11294          055664 000207              $STNPWR: 145000          ;; 1.0E09
11295          055666 145000              35632
11296          055670 035632              ;; 1.0E08
11297          055672 160400              160400
11298          055674 002765              2765
11299          055676 113200              ;; 1.0E07
11300          055700 000230              230
11301          055702 041100              ;; 1.0E06
11302          055704 000017              041100
11303          055706 103240              17
11304          055710 000001              ;; 1.0E05
11305          055712 023420              103240
11306          055714 000000              1
11307          055716 001750              23420
11308          055720 000000              ;; 1.0E04
11309          055722 000144              0
11310          055724 000000              1750
11311          055726 000012              ;; 1.0E03
11312          055730 000000              0
11313          055732 000001              144
11314          055734 000000              ;; 1.0E02
11315          055736 000014              0
11316              12          ;; 1.0E01
11317              0          ;; 1.0E00
11318              1          ;; 1.0E00
          $DECVL: .BLKB 12          ;; RESERVE STORAGE FOR ASCII STRING
          .SETTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
          ;;*****

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 225
TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

SEQ 0224

```

11319 ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
11320 ;*LEADING NUMBERS.
11321 ;*CALL
11322 ;*      MOV      #NUMADR, -(SP)      ;; FIRST ADDRESS OF ASCIZ STRING
11323 ;*      JSR      PC, @#SSUPRS
11324
11325 $SUPRS: MOV      RO, -(SP)           ;; SAVE RO
11326 055752 010046 000004      MOV      4(SP), RO           ;; PICKUP THE POINTER
11327 055754 016600      TSTB      (RO)           ;; TERMINATE OR?
11328 055760 105710      BEQ      2$           ;; BR IF YES
11329 055762 001403      CMPB      #'0, (RO)+      ;; IS THIS AN ASCII "0" ?
11330 055764 122720 000060      BEQ      1$           ;; BR IF YES
11331 055770 001773      DEC      RO           ;; BACKUP BY "1"
11332 055772 005300      MOV      RO, 3$           ;; SAVE FOR TYPING
11333 055774 010037 056002      TYPE           ;; GO TYPE
11334 056000 104401      .WORD      0           ;; ASCIZ POINTER GOES HERE
11335 056002 000000      MOV      (SP)+, RO           ;; RESTORE RO
11336 056004 012600      MOV      (SP)+, (SP)       ;; RESTORE THE STACK
11337 056006 012616      RTS      PC           ;; RETURN
11338 056010 000207
11339 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
11340
11341 ;*****
11342 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11343 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11344 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11345 ;*CALL:
11346 ;*      MOV      NUM, -(SP)           ;; NUMBER TO BE TYPED
11347 ;*      TYPOS           ;; CALL FOR TYPEOUT
11348 ;*      .BYTE      N           ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11349 ;*      .BYTE      M           ;; M=1 OR 0
11350 ;*                               ;; 1=TYPE LEADING ZEROS
11351 ;*                               ;; 0=SUPPRESS LEADING ZEROS
11352
11353 ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11354 ;*$TYPOS OR $TYPOC
11355 ;*CALL:
11356 ;*      MOV      NUM, -(SP)           ;; NUMBER TO BE TYPED
11357 ;*      TYPON           ;; CALL FOR TYPEOUT
11358
11359 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11360 ;*CALL:
11361 ;*      MOV      NUM, -(SP)           ;; NUMBER TO BE TYPED
11362 ;*      TYPOC           ;; CALL FOR TYPEOUT
11363
11364 056012 017646 000000 056235 $TYPOS: MOV      @2(SP), -(SP)       ;; PICKUP THE MODE
11365 056016 116637 000001      MOVB     1(SP), $OFILL      ;; LOAD ZERO FILL SWITCH
11366 056024 112637 056237      MOVB     (SP)+, $OMODE+1    ;; NUMBER OF DIGITS TO TYPE
11367 056030 062716 000002      ADD      #2, (SP)         ;; ADJUST RETURN ADDRESS
11368 056034 000406      BR      $TYPON
11369 056036 112737 000001 056235 $TYPOC: MOVB     #1, $OFILL      ;; SET THE ZERO FILL SWITCH
11370 056044 112737 000006 056237      MOVB     #6, $OMODE+1    ;; SET FOR SIX(6) DIGITS
11371 056052 112737 000005 056234 $TYPON: MOVB     #5, $OCNT      ;; SET THE ITERATION COUNT
11372 056060 010346      MOV      R3, -(SP)           ;; SAVE R3
11373 056062 010446      MOV      R4, -(SP)           ;; SAVE R4
11374 056064 010546      MOV      R5, -(SP)           ;; SAVE R5

```

```

11375 056066 113704 056237      MOVB $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
11376 056072 005404      NEG R4
11377 056074 062704 000006      ADD #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
11378 056100 110437 056236      MOVB R4,$OMODE      ;;SAVE IT FOR USE
11379 056104 113704 056235      MOVB $OFILL,R4      ;;GET THE ZERO FILL SWITCH
11380 056110 016605 000012      MOV 12(SP),R5      ;;PICKUP THE INPUT NUMBER
11381 056114 005003      CLR R3      ;;CLEAR THE OUTPUT WORD
11382 056116 006105      1$: ROL R5      ;;ROTATE MSB INTO "C"
11383 056120 000404      BR 3$      ;;GO DO MSB
11384 056122 006105      2$: ROL R5      ;;FORM THIS DIGIT
11385 056124 006105      ROL R5
11386 056126 006105      ROL R5
11387 056130 010503      MOV R5,R3
11388 056132 006103      3$: ROL R3      ;;GET LSB OF THIS DIGIT
11389 056134 105337 056236      DECB $OMODE      ;;TYPE THIS DIGIT?
11390 056140 100016      BPL 7$      ;;BR IF NO
11391 056142 042703 177770      BIC #177770,R3      ;;GET RID OF JUNK
11392 056146 001002      BNE 4$      ;;TEST FOR 0
11393 056150 005704      TST R4      ;;SUPPRESS THIS 0?
11394 056152 001403      BEQ 5$      ;;BR IF YES
11395 056154 005204      4$: INC R4      ;;DON'T SUPPRESS ANYMORE 0'S
11396 056156 052703 000060      BIS #'0,R3      ;;MAKE THIS DIGIT ASCII
11397 056162 052703 000040      5$: BIS #'1,R3      ;;MAKE ASCII IF NOT ALREADY
11398 056166 110337 056232      MOVB R3,$$      ;;SAVE FOR TYPING
11399 056172 104401 056232      TYPE 8$      ;;GO TYPE THIS DIGIT
11400 056176 105337 056234      7$: DECB $OCNT      ;;COUNT BY 1
11401 056202 003347      BGT 2$      ;;BR IF MORE TO DO
11402 056204 002402      BLT 6$      ;;BR IF DONE
11403 056206 005204      INC R4      ;;INSURE LAST DIGIT ISN'T A BLANK
11404 056210 000744      BR 2$      ;;GO DO THE LAST DIGIT
11405 056212 012605      6$: MOV (SP)+,R5      ;;RESTORE R5
11406 056214 012604      MOV (SP)+,R4      ;;RESTORE R4
11407 056216 012503      MOV (SP)+,R3      ;;RESTORE R3
11408 056220 016666 000002 000004      MOV 2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
11409 056226 012616      MOV (SP)+,(SP)
11410 056230 000002      RTI      ;;RETURN
11411 056232 000      8$: .BYTE 0      ;;STORAGE FOR ASCII DIGIT
11412 056233 000      .BYTE 0      ;;TERMINATOR FOR TYPE ROUTINE
11413 056234 000      $OCNT: .BYTE 0      ;;OCTAL DIGIT COUNTER
11414 056235 000      $OFILL: .BYTE 0      ;;ZERO FILL SWITCH
11415 056236 000000      $OMODE: .WORD 0      ;;NUMBER OF DIGITS TO TYPE
11416      .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11417
11418      ;*****
11419      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11420      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11421      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11422      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11423      ;*REPLACED WITH SPACES.
11424      ;*CALL:
11425      ;*      MOV NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
11426      ;*      TYPDS      ;;GO TO THE ROUTINE
11427
11428      $TYPDS:
11429      MOV R0,-(SP)      ;;PUSH R0 ON STACK
11430      MOV R1,-(SP)      ;;PUSH R1 ON STACK

```

```

11431 056244 010246      MOV      R2,-(SP)      ; PUSH R2 ON STACK
11432 056246 010346      MOV      R3,-(SP)      ; PUSH R3 ON STACK
11433 056250 010546      MOV      R5,-(SP)      ; PUSH R5 ON STACK
11434 056252 012746 020200  MOV      #20200,-(SP)    ; SET BLANK SWITCH AND SIGN
11435 056256 016605 000020  MOV      20(SP),R5    ; GET THE INPUT NUMBER
11436 056262 100004      BPL      1$      ; BR IF INPUT IS POS.
11437 056264 005405      NEG      R5      ; MAKE THE BINARY NUMBER POS.
11438 056266 112766 000055 000001  MOVB     #'-,1(SP)    ; MAKE THE ASCII NUMBER NEG.
11439 056274 005000      CLR      R0      ; ZERO THE CONSTANTS INDEX
11440 056276 012703 056454      MOV      #SDBLK,R3    ; SETUP THE OUTPUT POINTER
11441 056302 112723 000040      MOVB     #'',(R3)+    ; SET THE FIRST CHARACTER TO A BLANK
11442 056306 005002      CLR      R2      ; CLEAR THE BCD NUMBER
11443 056310 016001 056444      $DTBL(R0),R1    ; GET THE CONSTANT
11444 056314 160105      3$:      SUB      R1,R5    ; FORM THIS BCD DIGIT
11445 056316 002402      BLT      4$      ; BR IF DONE
11446 056320 005202      INC      R2      ; INCREASE THE BCD DIGIT BY 1
11447 056322 000774      BR      3$
11448 056324 060105      4$:      ADD      R1,R5    ; ADD BACK THE CONSTANT
11449 056326 005702      TST      R2      ; CHECK IF BCD DIGIT=7
11450 056330 001002      BNE      5$      ; FALL THROUGH IF 0
11451 056332 105716      TSTB     (SP)      ; STILL DOING LEADING 0'S?
11452 056334 100407      BMI      7$      ; BR IF YES
11453 056336 106316      5$:      ASLB     (SP)      ; MSD?
11454 056340 103003      BCC      6$      ; BR IF NO
11455 056342 116663 000001 177777  MOVB     1(SP),-1(R3)  ; YES--SET THE SIGN
11456 056350 052702 000060      BIS      #'0,R2    ; MAKE THE BCD DIGIT ASCII
11457 056354 052702 000040      7$:      BIS      #' ,R2    ; MAKE IT A SPACE IF NOT ALREADY A DIGIT
11458 056360 110223      MOVB     R2,(R3)+    ; PUT THIS CHARACTER IN THE OUTPUT BUFFER
11459 056362 005720      TST      (R0)+    ; JUST INCREMENTING
11460 056364 020027 000010      CMP      R0,#10    ; CHECK THE TABLE INDEX
11461 056370 002746      BLT      8$      ; GO DO THE NEXT DIGIT
11462 056372 003002      BGT      8$      ; GO TO EXIT
11463 056374 010502      MOV      R5,R2    ; GET THE LSD
11464 056376 000764      BR      6$      ; GO CHANGE TO ASCII
11465 056400 105726      8$:      TSTB     (SP)+    ; WAS THE LSD THE FIRST NON-ZERO?
11466 056402 100003      BPL      9$      ; BR IF NO
11467 056404 116663 177777 177776  MOVB     -1(SP),-2(R3)  ; YES--SET THE SIGN FOR TYPING
11468 056412 105013      9$:      CLRB     (R3)      ; SET THE TERMINATOR
11469 056414 012605      MOV      (SP)+,R5    ; POP STACK INTO R5
11470 056416 012603      MOV      (SP)+,R3    ; POP STACK INTO R3
11471 056420 012602      MOV      (SP)+,R2    ; POP STACK INTO R2
11472 056422 012601      MOV      (SP)+,R1    ; POP STACK INTO R1
11473 056424 012600      MOV      (SP)+,R0    ; POP STACK INTO R0
11474 056426 104401 056454      TYPE     #SDBLK    ; NOW TYPE THE NUMBER
11475 056432 016666 000002 000004  MOV      2(SP),4(SP)  ; ADJUST THE STACK
11476 056440 012616      MOV      (SP)+,(SP)
11477 056442 000002      RTI
11478 056444 023420      $DTBL: 10000.
11479 056446 001750      1000.
11480 056450 000144      100.
11481 056452 000012      10.
11482 056454 000004      $SDBLK: .BLKW 4
11483      .SBTTL  ERROR HANDLER ROUTINE
11484
11485      ;*****
11486      ;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.

```

```

11487 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11488 ;*AND GO TO TYPERR ON ERROR
11489 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11490 ;*SW15=1      HALT ON ERROR
11491 ;*SW13=1      INHIBIT ERROR TYPEOUTS
11492 ;*SW10=1      BELL ON ERROR
11493 ;*SW09=1      LOOF ON ERROR
11494 ;*CALL
11495 ;*          ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
11496
11497 $ERROR:
11498 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
11499      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
11500      MOV      $STNM,$DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
11501      BIT      $BIT10,$SWR      ;;BELL ON ERROR?
11502      BEQ      1$      ;;NO - SKIP
11503      TYPE      $BELL      ;;RING BELL
11504      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
11505      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
11506      SUB      #2,$ERRPC
11507      MOVB      $ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
11508      BIT      $BIT13,$SWR      ;;SKIP TYPEOUT IF SET
11509      BNE      20$      ;;SKIP TYPEOUTS
11510      JSR      PC,TYPERR      ;;GO TO USER ERROR ROUTINE
11511      TYPE      $CRLF
11512
11513 20$:      CMPB      $APTENV,$ENV      ;;RUNNING IN APT MODE
11514      BNE      2$      ;;NO SKIP APT ERROR REPORT
11515      MOVB      $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
11516      JSR      PC,$ATY4      ;;REPORT FATAL ERROR TO APT
11517
11518 21$:      .BYTE      0
11519      .BYTE      0
11520
11521 22$:      BR      22$      ;;APT ERROR LOOP
11522 2$:      TST      $SWR      ;;HALT ON ERROR
11523      BPL      3$      ;;SKIP IF CONTINUE
11524      HALT      ;;HALT ON ERROR!
11525      BIT      $BIT09,$SWR      ;;LOOP ON ERROR SWITCH SET?
11526      BEQ      4$      ;;BR IF NO
11527      MOV      $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
11528      TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
11529      BEQ      5$      ;;BR IF NONE
11530      MOV      $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
11531
11532 5$:      CMP      $SENDAD,$#42      ;;ACT-11 AUTO-ACCEPT?
11533      BNE      6$      ;;BRANCH IF NO
11534      HALT      ;;YES
11535
11536 6$:      RTI      ;;RETURN
11537      .SBTTL      TTY INPUT ROUTINE
11538 ;*****
11539 .ENABL      LSB
11540 .DSABL      LSB
11541
11542

```

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 229
TTY INPUT ROUTINE

SEQ 0228

```

11543 ;*****
11544 ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11545 ;CALL:
11546 ;*      RDCHR      ;INPUT A SINGLE CHARACTER FROM THE TTY
11547 ;*      RETURN HERE ;CHARACTER IS ON THE STACK
11548 ;*                ;WITH PARITY BIT STRIPPED OFF
11549 ;
11550
11551 056660 011646 000004 000002 $RDCHR: MOV      (SP), -(SP) ;PUSH DOWN THE PC
11552 056662 016666 000004 000002 1$: MOV      4(SP), 2(SP) ;SAVE THE PS
11553 056670 105777 122250 1$: TSTB     @STKS ;WAIT FOR
11554 056674 100375 122244 000004 BPL      1$ ;A CHARACTER
11555 056676 117766 122244 000004 MOV      @STKB, 4(SP) ;READ THE TTY
11556 056704 042766 177600 000004 BIC      #177, 4(SP) ;GET RID OF JUNK IF ANY
11557 056712 026627 000004 000023 CMP      4(SP), #23 ;IS IT A CONTROL-S?
11558 056720 001013 122216 2$: BNE      3$ ;BRANCH IF NO
11559 056722 105777 122216 2$: TSTB     @STKS ;WAIT FOR A CHARACTER
11560 056726 100375 122212 2$: BPL      2$ ;LOOP UNTIL ITS THERE
11561 056730 117746 177600 000021 MOV      @STKB, -(SP) ;GET CHARACTER
11562 056734 042716 177600 000021 BIC      #177, (SP) ;MAKE IT 7-BIT ASCII
11563 056740 022627 000021 2$: CMP      (SP)+, #21 ;IS IT A CONTROL-Q?
11564 056744 001366 122212 2$: BNE      2$ ;IF NOT DISCARD IT
11565 056746 000750 122212 1$: BR       1$ ;YES, RESUME
11566 056750 026627 000004 000140 3$: CMP      4(SP), #140 ;IS IT UPPER CASE?
11567 056756 002407 122212 000175 BLT      4$ ;BRANCH IF YES
11568 056760 026627 000004 000175 CMP      4(SP), #175 ;IS IT A SPECIAL CHAR?
11569 056766 003003 122212 000004 BGT      4$ ;BRANCH IF YES
11570 056770 042766 000040 000004 BIC      #40, 4(SP) ;MAKE IT UPPER CASE
11571 056776 000002 122212 000004 4$: RTI ;GO BACK TO USER
11572 057000 052536 005015 000 $CNTLU: .ASCIZ /U/<15><12> ;CONTROL "U"
11573 057005 136 006507 000012 $CNTLG: .ASCIZ /G/<15><12> ;CONTROL "G"
11574 057012 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
11575 057020 020075 000 $MNEW: .ASCIZ / NEW = /
11576 057023 040 047040 053505
11577 057030 036440 000040
11578
11579
11580
11581 .SBTTL POWER DOWN AND UP ROUTINES
11582
11583 :POWER DOWN ROUTINE
11584 057034 012737 057046 000024 $PWRDN: MOV      $PWRUP, PWRVEC ;SET VECTOR FOR POWER UP
11585 057042 000000 000000 HALT ;HANG UP
11586 057044 000776 000000 BR       -2
11587
11588 :POWER UP ROUTINE
11589 057046 005037 057120 $PWRUP: CLR      $PWRCT ;WAIT LOOP FOR TTY TO COME UP
11590 057052 005237 057120 4$: INC      $PWRCT
11591 057056 001375 000024 BNE      4$
11592 057060 012737 057034 000024 MOV      $PWRDN, PWRVEC ;SET VECTOR FOR POWER DOWN
11593 057066 012737 000340 000026 MOV      #PR7, PWRVEC+2 ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11594 057074 012737 000340 000036 MOV      #PR7, TRAPVEC+2
11595 057102 012706 001100 MOV      #STACK, SP ;RE-INITIALIZE THE STACK
11596 057106 104401 057122 TYPE      ,PWRMSG ;TYPE "POWER FAILED"
11597 057112 000005 000005 RESET ;CLEAR THE UNIBUS
11598 057114 000177 121766 JMP      @SLPADR ;RESTART THE CURRENT TEST

```

```

$PWRCT: .WORD      0
PWRMSG: .ASCIIZ    <15><12>/POWER FAILED/<15><12>

.EVEN

```

```

;*****
;* 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

```

```

;*****
;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
;SW11=1          INHIBIT ITERATIONS
;SW09=1          LOOP ON ERROR
;CALL
;          SCOPE          ;;SCOPE=IOT

```

```

$SCOPE:      TST      $TIMES      ;CHECK CURRENT ITERATION NUMBER
              BEQ      $XTSTR      ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
1$:          BIT      $BIT14,$SWR  ;LOOP ON PRESENT TEST?
              BNE      $OVER       ;YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR    6$
              MOV      2*$ERRVEC, -(SP) ;IF RUNNING ON THE "XOR" TESTER CHANGE
              MOV      2*$ERRVEC      ;THIS INSTRUCTION TO A "NOP" (NOP=240)
              TST      2*$177060      ;SAVE THE CONTENTS OF THE ERROR VECTOR
              MOV      (SP)+, 2*$ERRVEC ;SET FOR TIMEOUT
              BR        $SVLAD         ;TIME OUT ON XOR?
5$:          CMP      (SP)+, (SP)+    ;RESTORE THE ERROR VECTOR
              MOV      (SP)+, 2*$ERRVEC ;GO TO THE NEXT TEST
              BR        7$           ;CLEAR THE STACK AFTER A TIME OUT
6$:;*****END OF CODE FOR THE XOR TESTER*****
2$:          TSTB     $ERFLG         ;HAS AN ERROR OCCURRED?
              BEQ      3$           ;BR IF NO
              CMPB     $ERMAX, $ERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?
              BHI      3$           ;BR IF NO
              BIT      $BIT09,$SWR  ;LOOP ON ERROR?

```

```

11655 057270 001404      BEQ      4$      ;;BR IF NO
11656 057272 013737 001110 001106 7$:      MOV      $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
11657 057300 000443      BR        $OVER
11658 057302 105037 001103      4$:      CLRB     $ERFLG      ;;ZERO THE ERROR FLAG
11659 057306 005037 001304      CLR      $TIMES     ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11660 057312 000412      BR        1$      ;;ESCAPE TO THE NEXT TEST
11661 057314 032777 004000 121616 3$:      BIT      #BIT11,$SWR      ;;INHIBIT ITERATIONS?
11662 057322 001006      BNE      1$      ;;BR IF YES
11663 057324 005237 001104      INC      $ICNT      ;;INCREMENT ITERATION COUNT
11664 057330 023737 001304 001104      CMP      $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
11665 057336 002024      BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
11666 057340 012737 000001 001104 1$:      MOV      #1,$ICNT    ;;REINITIALIZE THE ITERATION COUNTER
11667 057346 013737 057424 001304      MOV      $MXCNT,$TIMES  ;;SET NUMBER OF ITERATIONS TO DO
11668 057354 105237 001102      $SVLAD: INCB     $TSTNM      ;;COUNT TEST NUMBERS
11669 057360 113737 001102 001324      MOV      $TSTNM,$TSTNM  ;;SET TEST NUMBER IN APT MAILBOX
11670 057366 011637 001106      MOV      (SP),$LPADR    ;;SAVE SCOPE LOOP ADDRESS
11671 057372 011637 001110      MOV      (SP),$LPERR    ;;SAVE ERROR LOOP ADDRESS
11672 057376 005037 001306      CLR      $ESCAPE     ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11673 057402 112737 000001 001115      MOV      #1,$ERMAX    ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11674 057410 013777 001102 121524 $OVER:  MOV      $TSTNM,$DISPLAY  ;;DISPLAY TEST NUMBER
11675 057416 013716 001106      MOV      $LPADR,(SP)    ;;FUDGE RETURN ADDRESS
11676 057422 000002      RTI
11677 057424 003720      $MXCNT: 2000.    ;;FIXES PS
11678                                     .SBTTL  APT COMMUNICATIONS ROUTINE
11679
11680 *****
11681 057426 112737 000001 057672 $ATY1:  MOV      #1,$FFLG      ;;TO REPORT FATAL ERROR
11682 057434 112737 000001 057670 $ATY3:  MOV      #1,$MFLG      ;;TO TYPE A MESSAGE
11683 057442 000403      BR        $ATYC
11684 057444 112737 000001 057672 $ATY4:  MOV      #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
11685 057452 $ATYC:
11686 057452 010046      MOV      RO,-(SP)    ;;PUSH RO ON STACK
11687 057454 010146      MOV      R1,-(SP)    ;;PUSH R1 ON STACK
11688 057456 105737 057670      TSTB     $MFLG      ;;SHOULD TYPE A MESSAGE?
11689 057462 001450      BEQ      5$      ;;IF NOT: BR
11690 057464 122737 000001 001340      CMPB     #APTENV,$ENV      ;;OPERATING UNDER APT?
11691 057472 001031      BNE      3$      ;;IF NOT: BR
11692 057474 132737 000100 001341      BITB     #APTSPOOL,$ENVM    ;;SHOULD SPOOL MESSAGES?
11693 057502 001425      BEQ      3$      ;;IF NOT: BR
11694 057504 017600 000004      MOV      @4(SP),RO    ;;GET MESSAGE ADDR.
11695 057510 062766 000002 000004      ADD      #2,4(SP)    ;;BUMP RETURN ADDR.
11696 057516 005737 001320 1$:      TST      $MSGTYPE    ;;SEE IF DONE W/ LAST XMISSION?
11697 057522 001375      BNE      1$      ;;IF NOT: WAIT
11698 057524 010037 001334      MOV      RO,$MSGAD    ;;PUT ADDR IN MAILBOX
11699 057530 105720 2$:      TSTB     (RO)+      ;;FIND END OF MESSAGE
11700 057532 001376      BNE      2$
11701 057534 163700 001334      SUB      $MSGAD,RO    ;;SUB START OF MESSAGE
11702 057540 006200      ASR      RO      ;;GET MESSAGE LNTH IN WORDS
11703 057542 010037 001336      MOV      RO,$MSGGLT    ;;PUT LENGTH IN MAILBOX
11704 057546 012737 000004 001320      MOV      #4,$MSGTYPE  ;;TELL APT TO TAKE MSG.
11705 057554 000413      BR        5$
11706 057556 017637 000004 057602 3$:      MOV      @4(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE
11707 057564 062766 000002 000004      ADD      #2,4(SP)    ;;BUMP RETURN ADDRESS
11708 057572 013746 177776      MOV      177776,-(SP)    ;;PUSH 177776 ON STACK
11709 057576 004737 054736      JSR      PC,$TYPE    ;;CALL TYPE MACRO
11710 057602 000000      4$:      .WORD      0

```



```

11711 057604
11712 057604 105737 057672
11713 057610 001416
11714 057612 005737 001340
11715 057616 001413
11716 057620 005737 001320
11717 057624 001375
11718 057626 017637 000004 001322
11719 057634 062766 000002 000004
11720 057642 005237 001320
11721 057646 105037 057672
11722 057652 105037 057671
11723 057656 105037 057670
11724 057662 012601
11725 057664 012600
11726 057666 000207
11727 057670 000
11728 057671 000
11729 057672 000
11730 057674
11731 000200
11732 000001
11733 000100
11734 000040
11735
11736
11737
11738
11739
11740
11741
11742
11743
11744
11745
11746
11747
11748
11749
11750
11751 057674 010046
11752 057676 010146
11753 057700 010246
11754 057702 010346
11755 057704 013746 000004
11756 057710 013746 000006
11757 057714 010600
11758
11759 057716 104400
11760 057720 012637 000006
11761 057724 012701 003776
11762 057730 105727
11763 057732 000200
11764 057734 100062
11765 057736 012737 060074 000004
11766 057744 005737 177572

5$:
10$: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
      BEQ 12$ ;; IF NOT: BR
      TST $ENV ;; RUNNING UNDER APT?
      BEQ 12$ ;; IF NOT: BR
11$: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
      BNE 11$ ;; IF NOT: WAIT
      MOV 24(SP), $FATAL ;; GET ERROR #
      ADD #2, 4(SP) ;; BUMP RETURN ADDR.
      INC $MSGTYPE ;; TELL APT TO TAKE ERROR
12$: CLRB $FFLG ;; CLEAR FATAL FLAG
      CLRB $LFLG ;; CLEAR LOG FLAG
      CLRB $MFLG ;; CLEAR MESSAGE FLAG
      MOV (SP)+, R1 ;; POP STACK INTO R1
      MOV (SP)+, R0 ;; POP STACK INTO R0
      RTS PC ;; RETURN
      $MFLG: .BYTE 0 ;; MESSG. FLAG
      $LFLG: .BYTE 0 ;; LOG FLAG
      $FFLG: .BYTE 0 ;; FATAL FLAG
      .EVEN

APTSIZE=200
APTENV=001
APTSPool=100
APTCsup=040
.SBTTL ROUTINE TO SIZE MEMORY

*****
*CALL:
* JSR PC, $SIZE
* RETURN
* $LSTAD WILL CONTAIN:
* WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
* WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
* $LSTBK WILL CONTAIN THE LAST BANK AS A SAF
* $KT11 IS THE MEMORY MANAGEMENT KEY
* $BIT07 = 0 DON'T USE MEMORY MANAGEMENT
* MUST BE SETUP BEFORE THE CALL
* $BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
* DETERMINED BY ROUTINE

$SIZE: MOV R0, -(SP) ;; SAVE R0 ON THE STACK
      MOV R1, -(SP) ;; SAVE R1 ON THE STACK
      MOV R2, -(SP) ;; SAVE R2 ON THE STACK
      MOV R3, -(SP) ;; SAVE R3 ON THE STACK
      MOV 2*$ERRVEC, -(SP) ;; SAVE PRESENT ERROR VECTOR PS & PC
      MOV 2*$ERRVEC+2, -(SP)
      MOV SP, R0 ;; SAVE THE STACK POINTER
;; SET THE ERRVEC PS TO THE PRESENT PS
      TRAP PS ;; PUSH OLD PSW AND PC ON STACK
      MOV (SP)+, 2*$ERRVEC+2 ;; SAVE THE PSW IN 2*$ERRVEC+2
      MOV #3776, R1 ;; SETUP ADDRESS
      TSTB (PC)+ ;; USE MEMORY MANAGEMENT?
$KT11: .WORD 200 ;; SET TO USE MEMORY MANAGEMENT
      BPL $SCORE ;; BR IF NO
      MOV $SKTNEX, 2*$ERRVEC ;; SET FOR TIMEOUT
      TST 2*$SRO ;; KT11 ARE YOU THERE?

```

```

11767 057750 052737 100000 057732 BIS #100000, $KT11 ;; YES--SET KT11 KEY
11768 057756 005046 CLR -(SP) ;; INITIALIZE FOR "PAR" LOADING
11769 057760 012702 172340 MOV #KIPAR0, R2 ;; ADDRESS OF FIRST "PAR"
11770 057764 012703 000010 MOV #108, R3 ;; LOAD EIGHT "PAR."S AND EIGHT "PDR."S"
11771 057770 012762 077406 177740 1$: MOV #77406, -40(R2) ;; PDR = 4K, UP, READ/WRITE
11772 057776 011622 MOV (SP), (R2)+ ;; LOAD "PAR"
11773 060000 062716 000200 ADD #200, (SP) ;; UPDATE FOR NEXT "PAR"
11774 060004 077307 SOB R3, 1$ ;; LOOP UNTIL ALL EIGHT ARE LOADED
11775 060006 012742 177600 MOV #177600, -(R2) ;; SETUP KIPAR7 FOR I/O
11776 060012 005042 CLR -(R2) ;; SETUP KIPAR6 FOR TESTING
11777 060014 012737 060032 000004 MOV #2$, 2$ERRVEC ;; CATCH TIMEOUT IF NO SR3
11778 060022 012737 000020 172516 MOV #20, 2$SR3 ;; ENABLE 22 BIT MODE
11779 060030 000401 BR 3$ ;; THIS PDP-11 HAS A SR3 REGISTER
11780 060032 022626 2$: CMP (SP)+, (SP)+ ;; CLEAN OFF THE STACK- NO SR3
11781 060034 005237 177572 3$: INC 2$SR0 ;; TURN ON MEMORY MANAGEMENT
11782 060040 012737 060064 000004 MOV #SKTOUT, 2$ERRVEC ;; SET FOR TIME OUT
11783 060046 005737 143776 4$: TST 2$143776 ;; TRAP ON NON-EX-MEM
11784 060052 062712 000040 ADD #40, (R2) ;; MAKE A 1K STEP
11785 060056 023712 172356 CMP 2$KIPAR7, (R2) ;; LAST ONE?
11786 060062 101371 BHI 4$ ;; NO--TRY IT
11787 060064 011202 SKTOUT: MOV (R2), R2 ;; GET LAST BANK+1
11788 060066 005037 177572 CLR 2$SR0 ;; TURN OFF MEMORY MANAGEMENT
11789 060072 000421 BR $SIZEX
11790 060074 042737 100000 057732 SKTNEX: BIC #100000, $KT11 ;; KT11 NON-EXISTENT
11791 060102 012737 060132 000004 SCORE: MOV #SCROUT, 2$ERRVEC ;; SET FOR TIMEOUT
11792 060110 005002 CLR R2 ;; SET UP BANK
11793 060112 062701 004000 1$: ADD #4000, R1 ;; INCREMENT BY 1K
11794 060116 062702 000040 ADD #40, R2 ;; 1K STEP
11795 060122 005711 TST (R1) ;; TRAP ON TIME OUT
11796 060124 022701 177776 CMP #177776, R1 ;; LAST ONE
11797 060130 001370 BNF 1$ ;; NO--TRY AGAIN
11798 060132 162701 004000 SCROUT: SUB #4000, R1 ;; DROP BACK
11799 060136 162702 000040 $SIZEX: SUB #40, R2 ;; RESTORE THE STACK
11800 060142 010006 RO, SP ;; RESTORE ERROR VECTOR
11801 060144 012637 000006 MOV (SP)+, 2$ERRVEC+2 ;; RESTORE ERROR VECTOR
11802 060150 012637 000004 MOV (SP)+, 2$ERRVEC
11803 060154 010137 060176 MOV R1, $LSTAD ;; LAST ADDRESS
11804 060160 010237 060200 MOV R2, $LSTBK ;; LAST BANK
11805 060164 012603 MOV (SP)+, R3 ;; RESTORE R3
11806 060166 012602 MOV (SP)+, R2 ;; RESTORE R2
11807 060170 012601 MOV (SP)+, R1 ;; RESTORE R1
11808 060172 012600 MOV (SP)+, R0 ;; RESTORE R0
11809 060174 000207 RTS PC
11810 060176 000000 $LSTAD: .WORD 0 ;; CONTAINS THE LAST ADDRESS
11811 060200 000000 $LSTBK: .WORD 0 ;; CONTAINS THE LAST BANK
11812 .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
11813
11814 *****
11815 *SAVE R0-R5
11816 *CALL:
11817 * SAVREG
11818 *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11819 *
11820 *TOP---(+16)
11821 * +2---(+18)
11822 * +4---R5

```

```

11823      ; * +6---R4
11824      ; * +8---R3
11825      ; * +10---R2
11826      ; * +12---R1
11827      ; * +14---R0
11828
11829      $SAVREG:
11830      060202      010046      MOV      R0, -(SP)      ; PUSH R0 ON STACK
11831      060204      010146      MOV      R1, -(SP)      ; PUSH R1 ON STACK
11832      060206      010246      MOV      R2, -(SP)      ; PUSH R2 ON STACK
11833      060210      010346      MOV      R3, -(SP)      ; PUSH R3 ON STACK
11834      060212      010446      MOV      R4, -(SP)      ; PUSH R4 ON STACK
11835      060214      010546      MOV      R5, -(SP)      ; PUSH R5 ON STACK
11836      060216      016646      000022      MOV      22(SP), -(SP)      ; SAVE PS OF MAIN FLOW
11837      060222      016646      000022      MOV      22(SP), -(SP)      ; SAVE PC OF MAIN FLOW
11838      060226      016646      000022      MOV      22(SP), -(SP)      ; SAVE PS OF CALL
11839      060232      016646      000022      MOV      22(SP), -(SP)      ; SAVE PC OF CALL
11840      060236      000002      RTI
11841
11842      ; *RESTORE R0-R5
11843      ; *CALL:
11844      ; *RESREG:
11845      060240      012666      000022      MOV      (SP)+, 22(SP)      ; RESTORE PC OF CALL
11846      060244      012666      000022      MOV      (SP)+, 22(SP)      ; RESTORE PS OF CALL
11847      060250      012666      000022      MOV      (SP)+, 22(SP)      ; RESTORE PC OF MAIN FLOW
11848      060254      012666      000022      MOV      (SP)+, 22(SP)      ; RESTORE PS OF MAIN FLOW
11849      060260      012605      MOV      (SP)+, R5      ; POP STACK INTO R5
11850      060262      012604      MOV      (SP)+, R4      ; POP STACK INTO R4
11851      060264      012603      MOV      (SP)+, R3      ; POP STACK INTO R3
11852      060266      012602      MOV      (SP)+, R2      ; POP STACK INTO R2
11853      060270      012601      MOV      (SP)+, R1      ; POP STACK INTO R1
11854      060272      012600      MOV      (SP)+, R0      ; POP STACK INTO R0
11855      060274      000002      RTI
11856
11857      .SBTTL TRAP DECODER
11858
11859      ; *****
11860      ; THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11861      ; AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11862      ; OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11863      ; GO TO THAT ROUTINE.
11864
11865      $TRAP: MOV      R0, -(SP)      ; SAVE R0
11866      060300      016600      000002      MOV      2(SP), R0      ; GET TRAP ADDRESS
11867      060304      005740      TST      -(R0)      ; BACKUP BY 2
11868      060306      111000      MOV      (R0), R0      ; GET RIGHT BYTE OF TRAP
11869      060310      006300      ASL      R0      ; POSITION FOR INDEXING
11870      060312      016000      060332      MOV      $TRPAD(R0), R0      ; INDEX TO TABLE
11871      060316      000200      RTS      R0      ; GO TO ROUTINE
11872
11873      ; ; THIS IS USE TO HANDLE THE "GETPRI" MACRO
11874
11875      $TRAP2: MOV      (SP), -(SP)      ; MOVE THE PC DOWN
11876      060320      011646      000004      000002      MOV      4(SP), 2(SP)      ; MOVE THE PSW DOWN
11877      060322      016666      000004      000002      RTI      ; RESTORE THE PSW
11878      060330      000002

```

.SBTTL TRAP TABLE

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

	ROUTINE			
\$TRPAD:	WORD	\$TRAP2		
	\$TYPE	::CALL=TYPE	TRAP+1(104401)	TTY TYPEOUT ROUTINE
	\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPDS	::CALL=TYPDS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
	\$RDCHR	::CALL=RDCHR	TRAP+6(104406)	TTY TYPEIN CHARACTER ROUTINE
	\$SAVREG	::CALL=SAVREG	TRAP+7(104407)	SAVE RO-R5 ROUTINE
	\$RESREG	::CALL=RESREG	TRAP+10(104410)	RESTORE RO-R5 ROUTINE
	\$SCOPE1	::CALL=SCOPE1	TRAP+11(104411)	INTERNAL LOOP ON ERROR ROUTINE

11879						
11880						
11881						
11882						
11883						
11884						
11885						
11886						
11887	060332	060320				
11888	060334	054736				
11889	060336	056036				
11890	060340	056012				
11891	060342	056052				
11892	060344	056240				
11893						
11894						
11895	060346	056660				
11896	060350	060202				
11897	060352	060240				
11898	060354	057144				
11899						
11900	060356	020040	020040	042524	TSTMSG:	.ASCIZ / TEST /
11901	060364	052123	000040			
11902	060370	025052	020040	000	AS2SP2:	.ASCIZ /** /
11903	060375	125	044516	052502	EM1:	.ASCIZ /UNIBUS PAR ERR/
11904	060402	020123	040520	020122		
11905	060410	051105	000122			
11906	060414	047516	026516	054105	EM2:	.ASCIZ /NON-EXIST MEM/
11907	060422	051511	020124	042515		
11908	060430	000115				
11909	060432	047516	026516	054105	EM3:	.ASCIZ /NON-EXIST DRV/
11910	060440	051511	020124	051104		
11911	060446	000126				
11912	060450	047125	052111	043040	EM4:	.ASCIZ /UNIT FIELD ERR/
11913	060456	042511	042114	042440		
11914	060464	051122	000			
11915	060467	123	041125	054523	EM5:	.ASCIZ /SUBSYS TIMEOUT/
11916	060474	020123	044524	042515		
11917	060502	052517	000124			
11918	060506	020104	047524	041440	EM6:	.ASCIZ /D TO C PAR ERR/
11919	060514	050040	051101	042440		
11920	060522	051122	000			
11921	060525	103	052040	020117	EM7:	.ASCIZ /C TO D PAR ERR/
11922	060532	020104	040520	020122		
11923	060540	051105	000122			
11924	060544	041501	046040	053517	EM10:	.ASCIZ /AC LOW/
11925	060552	000				
11926	060553	123	042520	042105	EM11:	.ASCIZ /SPEED LOSS/
11927	060560	046040	051517	000123	EM12:	.ASCIZ /ILL FUNCT/
11928	060566	046111	020114	052506		
11929	060574	041516	000124			
11930	060600	051120	043517	042440	EM13:	.ASCIZ /PROG ERR/
11931	060606	051122	000			
11932	060611	116	047117	042455	EM14:	.ASCIZ /NON-EXIST FUNCT/
11933	060616	044530	052123	043040		
11934	060624	047125	052103	000		

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 236
TRAP TABLE

SEQ 0235

11935	060631	104	053122	052040	EM15:	.ASCIZ	/DRV TYP ERR/
11936	060636	050131	042440	051122			
11937	060644	000					
11938	060645	106	040515	020124	EM16:	.ASCIZ	/FMAT ERR/
11939	060652	051105	000122				
11940	060656	051127	020124	047514	EM17:	.ASCIZ	/WRT LOCK ERR/
11941	060664	045503	042440	051122			
11942	060672	000					
11943	060673	104	053122	052440	EM20:	.ASCIZ	/DRV UNSAFE/
11944	060700	051516	043101	000105			
11945	060706	042523	045505	044440	EM21:	.ASCIZ	/SEEK INCOMP/
11946	060714	041516	046517	000120			
11947	060722	054503	020114	053117	EM22:	.ASCIZ	/CYL OVRFL0/
11948	060730	043122	047514	000			
11949	060735	111	046114	041440	EM23:	.ASCIZ	/ILL CYL ADDR/
11950	060742	046131	040440	042104			
11951	060750	000122					
11952	060752	051104	020126	043117	EM24:	.ASCIZ	/DRV OFF TRACK/
11953	060760	020106	051124	041501			
11954	060766	000113					
11955	060770	051104	020126	044524	EM25:	.ASCIZ	/DRV TIMING ERR/
11956	060776	044515	043516	042440			
11957	061004	051122	000				
11958	061007	104	052101	020101	EM26:	.ASCIZ	/DATA LATE/
11959	061014	040514	042524	000			
11960	061021	103	047117	051124	EM27:	.ASCIZ	/CONTR TIMEOUT/
11961	061026	052040	046511	047505			
11962	061034	052125	000				
11963	061037	117	042520	040522	EM30:	.ASCIZ	/OPERATION INCOMP/
11964	061044	044524	047117	044440			
11965	061052	041516	046517	000120			
11966	061060	042510	042101	051105	EM31:	.ASCIZ	/HEADER VRC ERR/
11967	061066	053040	041522	042440			
11968	061074	051122	000				
11969	061077	104	052101	020101	EM32:	.ASCIZ	/DATA CHECK ERR/
11970	061104	044103	041505	020113			
11971	061112	051105	000122				
11972	061116	051127	020124	044103	EM33:	.ASCIZ	/WRT CHK ERR/
11973	061124	020113	051105	000122			
11974	061132	040504	040524	046440	EM34:	.ASCIZ	/DATA MISCOMPARE/
11975	061140	051511	047503	050115			
11976	061146	051101	000105				
11977	061152	047516	042040	053122	EM35:	.ASCIZ	/NO DRV RESPONSE-UFE & NXD/
11978	061160	051040	051505	047520			
11979	061166	051516	026505	043125			
11980	061174	020105	020046	054116			
11981	061202	000104					
11982	061204	051104	020126	051105	EM36:	.ASCIZ	/DRV ERR WILL NOT CLEAR/
11983	061212	020122	044527	046114			
11984	061220	047040	052117	041440			
11985	061226	042514	051101	000			
11986	061233	104	053122	051440	EM37:	.ASCIZ	/DRV STATUS CHANGE WILL NOT CLEAR/
11987	061240	040524	052524	020123			
11988	061246	044103	047101	042507			
11989	061254	053440	046111	020114			
11990	061262	047516	020124	046103			

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 237
TRAP TABLE

SEQ 0236

11991	061270	040505	000122		
11992	061274	052101	023524	020116	EM40: .ASCIZ /ATT'N BUT NO STATUS CHANGE OR FAULT/
11993	061302	052502	020124	047516	
11994	061310	051440	040524	052524	
11995	061316	020123	044103	047101	
11996	061324	042507	047440	020122	
11997	061332	040506	046125	000124	
11998	061340	052101	023524	020116	EM41: .ASCIZ /ATT'N BUT DRV NOT AVAIL/
11999	061346	052502	020124	051104	
12000	061354	020126	047516	020124	
12001	061362	053101	044501	000114	
12002	061370	052101	023524	020116	EM42: .ASCIZ /ATT'N WHEN NOT EXPECTED/
12003	061376	044127	047105	047040	
12004	061404	052117	042440	050130	
12005	061412	041505	042524	000104	
12006	061420	051105	020122	040507	EM43: .ASCIZ /ERR GATHERING DRV STATUS/
12007	061426	044124	051105	047111	
12008	061434	020107	051104	020126	
12009	061442	052123	052101	051525	
12010	061450	000			
12011	061451	115	046125	020124	EM52: .ASCIZ /MULT DRV SEL/
12012	061456	051104	020126	042523	
12013	061464	000114			
12014	061466	042510	042101	051105	EM53: .ASCIZ /HEADER COMPARE ERR/
12015	061474	041440	046517	040520	
12016	061502	042522	042440	051122	
12017	061510	000			
12018	061511	123	041125	054523	EM56: .ASCIZ /SUBSYS TIMEOUT/
12019	061516	020123	044524	042515	
12020	061524	052517	000124		
12021	061530	051105	020122	047111	EM60: .ASCIZ /ERR IN RECAL FOR RECOVERY/
12022	061536	051040	041505	046101	
12023	061544	043040	051117	051040	
12024	061552	041505	053117	051105	
12025	061560	000131			
12026	061562	051104	050117	044520	EM61: .ASCIZ /DROPPING DRIVE FATAL ERROR/
12027	061570	043516	042040	044522	
12028	061576	042526	043040	052101	
12029	061604	046101	042440	051122	
12030	061612	051117	000		
12031	061615	103	046131	046440	EM62: .ASCIZ /CYL MISCOMP/
12032	061622	051511	047503	050115	
12033	061630	000			
12034	061631	103	042514	051101	EM63: .ASCIZ /CLEAR CONTR DID NOT CLEAR ERR/
12035	061636	041440	047117	051124	
12036	061644	042040	042111	047040	
12037	061652	052117	041440	042514	
12038	061660	051101	042440	051122	
12039	061666	000			
12040	061667	116	020117	052101	EM64: .ASCIZ /NO ATT'N IN ATT'N REG/
12041	061674	023524	020116	047111	
12042	061702	040440	052124	047047	
12043	061710	051040	043505	000	
12044	061715	125	051516	046117	EM65: .ASCIZ /UNSOLICITED ATT'N/
12045	061722	041511	052111	042105	
12046	061730	040440	052124	047047	

E03

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 238
TRAP TABLE

SEQ 0237

12047	061736	000			
12048	061737	125	042516	050130	EM66: .ASCIZ /UNEXPECTED DATA ERR/
12049	061744	041505	042524	020104	
12050	061752	040504	040524	042440	
12051	061760	051122	000		
12052	061763	101	052124	047047	EM67: .ASCIZ /ATT'N DID NOT RESET WITH CLEAR/
12053	061770	042040	042111	047040	
12054	061776	052117	051040	051505	
12055	062004	052105	053440	052111	
12056	062012	020110	046103	040505	
12057	062020	000122			
12058	062022	052523	051502	051531	EM70: .ASCIZ /SUBSYS CLEAR DID NOT CLEAR DRV ATT'N/
12059	062030	041440	042514	051101	
12060	062036	042040	042111	047040	
12061	062044	052117	041440	042514	
12062	062052	051101	042040	053122	
12063	062060	040440	052124	047047	
12064	062066	C 7			
12065	062067	104	052101	020101	EM71: .ASCIZ /DATA LATE WHEN UNLOADING HEADER/
12066	062074	040514	042524	053440	
12067	062102	042510	020116	047125	
12068	062110	047514	042101	047111	
12069	062116	020107	042510	042101	
12070	062124	051105	000		
12071	062127	103	047117	051124	EM72: .ASCIZ /CONTR ERR WHEN DRIVER SERV/
12072	062134	042440	051122	053440	
12073	062142	042510	020116	051104	
12074	062150	053111	051105	051440	
12075	062156	051105	000126		
12076	062162	051104	020126	042504	EM73: .ASCIZ /DRV DET PAR ERR/
12077	062170	020124	040520	020122	
12078	062176	051105	000122		
12079	062202	047125	042504	020106	EM74: .ASCIZ /UNDEF ERR/
12080	062210	051105	000122		
12081	062214	040515	045522	047111	EM75: .ASCIZ /MARKING THIS SECTOR BAD/
12082	062222	020107	044124	051511	
12083	062230	051440	041505	047524	
12084	062236	020122	040502	000104	
12085	062244	040502	020104	040504	EM76: .ASCIZ /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
12086	062252	040524	053040	051105	
12087	062260	043111	047047	053440	
12088	062266	052111	020110	042522	
12089	062274	042101	020056	041505	
12090	062302	020103	043117	046040	
12091	062310	051501	020124	042522	
12092	062316	051124	020131	051511	
12093	062324	000072			
12094	062326	042522	051124	020131	EM77: .ASCIZ /RETRY SUCCESSFUL/
12095	062334	052523	041503	051505	
12096	062342	043123	046125	000	
12097	062347	122	052105	054522	EM100: .ASCIZ /RETRY UNSUCCESSFUL/
12098	062354	052440	051516	041525	
12099	062362	042503	051523	052506	
12100	062370	000114			
12101	062372	040503	047116	052117	EM101: .ASCIZ /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
12102	062400	043040	047111	020104	

12103	062406	020101	040526	044514	
12104	062414	020104	042510	042101	
12105	062422	051105	044440	020116	
12106	062430	051124	041501	020113	
12107	062436	052512	052123	051040	
12108	062444	040505	000104		
12109	062450	040502	020104	042523	EM102: .ASCIZ /BAD SECT ERR ON SECT NOT LISTED BAD/
12110	062456	052103	042440	051122	
12111	062464	047440	020116	042523	
12112	062472	052103	047040	052117	
12113	062500	046040	051511	042524	
12114	062506	020104	040502	000104	
12115	062514	044524	042515	026504	EM103: .ASCIZ /TIMED-OUT ON READ HDR/
12116	062522	052517	020124	047117	
12117	062530	051040	040505	020104	
12118	062536	042110	000122		
12119	062542	044524	042515	026504	EM104: .ASCIZ /TIMED-OUT ON SEEK/
12120	062550	052517	020124	047117	
12121	062556	051440	042505	000113	
12122	062564	051104	020126	044523	EM105: .ASCIZ /DRV SIEZED BY OTHER PORT/
12123	062572	055105	042105	041040	
12124	062600	020131	052117	042510	
12125	062606	020122	047520	052122	
12126	062614	000			
12127	062615	104	052101	020101	EM106: .ASCIZ /DATA MISCMPR WHILE BAI SET/
12128	062622	044515	041523	050115	
12129	062630	020122	044127	046111	
12130	062636	020105	040502	020111	
12131	062644	042523	000124		
12132	062650	047516	047040	046505	EM107: .ASCIZ /NO NEM ERR WHEN REF'ING LOC 760000/
12133	062656	042440	051122	053440	
12134	062664	042510	020116	042522	
12135	062672	023506	047111	020107	
12136	062700	047514	020103	033067	
12137	062706	030060	030060	000	
12138	062713	111	052116	050122	EM110: .ASCIZ /INTRPT WHEN CNTRLR NOT RDY/
12139	062720	020124	044127	047105	
12140	062726	041440	052116	046122	
12141	062734	020122	047516	020124	
12142	062742	042122	000131		
12143	062746	047516	040440	052124	EM111: .ASCIZ /NO ATT'N ON SEEK/
12144	062754	047047	047440	020116	
12145	062762	042523	045505	000	
12146	062767	104	053122	051447	EM112: .ASCIZ /DRV'S CYL INCORRECT/
12147	062774	041440	046131	044440	
12148	063002	041516	051117	042522	
12149	063010	052103	000		
12150	063013	101	047502	052122	EM113: .ASCIZ /ABORT- CAN'T READ BSF/
12151	063020	020055	040503	023516	
12152	063026	020124	042522	042101	
12153	063034	041040	043123	000	
12154	063041	113	030524	020061	EM114: .ASCIZ /KT11 FAILURE/
12155	063046	040506	046111	051125	
12156	063054	000105			
12157	063056	042515	020115	040520	EM115: .ASCIZ /MEM PAR ERR/
12158	063064	020122	051105	000122	

G03

CZR6ND0 RK611 06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 240
TRAP TABLE

SEQ 0239

12159	063072	052503	051122	047105	DH100:	.ASCIZ	/CURRENT CMD :/
12160	063100	020124	046503	020104			
12161	063106	000072					
12162	063110	051120	053105	041440	DH105:	.ASCIZ	/PREV CMD :/
12163	063116	042115	035040	000			
12164	063123	122	033113	030461	DH200:	.ASCIZ	/PK611 REGS :/
12165	063130	051040	043505	020123			
12166	063136	000072					
12167	063140	042522	040515	047111	DH500:	.ASCIZ	/REMAINING REGS NOT VALID/
12168	063146	047111	020107	042522			
12169	063154	051507	047040	052117			
12170	063162	053040	046101	042111			
12171	063170	000					
12172	063171	124	042510	043040	DH501:	.ASCIZ	/THE FOLLOWING REG DATA MAY BE INCORRECT :/
12173	063176	046117	047514	044527			
12174	063204	043516	051040	043505			
12175	063212	042040	052101	020101			
12176	063220	040515	020131	042502			
12177	063226	044440	041516	051117			
12178	063234	042522	052103	035040			
12179	063242	000					
12180	063243	105	051122	050040	DH101:	.ASCIZ	/ERR PC DRIVE CMND CYLNDR TRACK SECTOR WD CNT/
12181	063250	020103	042040	044522			
12182	063256	042526	020040	041440			
12183	063264	047115	020104	020040			
12184	063272	041440	046131	042116			
12185	063300	020122	052040	040522			
12186	063306	045503	020040	051440			
12187	063314	041505	047524	020122			
12188	063322	053440	020104	047103			
12189	063330	000124					
12190	063332	044510	041040	020101	DH102:	.ASCIZ	/HI BA LO BA/
12191	063340	020040	047514	041040			
12192	063346	000101					
12193	063350	051120	053105	020056	DH103:	.ASCIZ	/PREV. UNIBUS MAP :/
12194	063356	047125	041111	051525			
12195	063364	046440	050101	035040			
12196	063372	000					
12197	063373	110	020111	042522	DH104:	.ASCIZ	/HI REGO LO REGO/
12198	063400	030107	046040	020117			
12199	063406	042522	030107	000			
12200	063413	103	051125	020056	DH106:	ASCIZ	/CUR. UNIBUS MAP :/
12201	063420	047125	041111	051525			
12202	063426	046440	050101	035040			
12203	063434	000					
12204	063435	122	041513	030523	DH201:	.ASCIZ	/RKCS1 RKCS2 RKDC RKDA RKWCR RKBA RKASOF/
12205	063442	020040	051040	041513			
12206	063450	031123	020040	051040			
12207	063456	042113	020103	020040			
12208	063464	051040	042113	020101			
12209	063472	020040	051040	053513			
12210	063500	051103	020040	051040			
12211	063506	041113	020101	020040			
12212	063514	051040	040513	047523			
12213	063522	000106					
12214	063524	045522	051504	020040	DH202:	.ASCIZ	/RKDS RKER/

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 241
TRAP TABLE

SEO 0240

[illegible]

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 242
TRAP TABLE

SEQ 0241

12271	064202	020124	042101	020040					
12272	064210	044513	040520	033122					
12273	064216	000							
12274	064217	107	047517	020104	DH702:	.ASCIZ	/GOOD	BAD/	
12275	064224	020040	041040	042101					
12276	064232	000							
12277	064233	106	044501	044514	DH703:	.ASCIZ	/FAILING DATA WORD :/		
12278	064240	043516	042040	052101					
12279	064246	020101	047527	042122					
12280	064254	035040	000						
12281	064257	123	030122	051411	DH704:	.ASCIZ	/SR0	SR1	SR2
12282	064264	030522	051411	031122					
12283	064272	051411	031522	000					
12284	064277	113	030524	020061	DH705:	.ASCIZ	/KT11 REGS :/		
12285	064304	042522	051507	035040					
12286	064312	000							
12287	064313	115	046505	051040	DH706:	.ASCIZ	/MEM REGS :/		
12288	064320	043505	020123	000072					
12289	064326	041520	046011	042517	DH707:	.ASCIZ	/PC	LOERAD	HIERAD
12290	064334	040522	004504	044510					MEMSYS/
12291	064342	051105	042101	046411					
12292	064350	046505	054523	000123					
12293	064356	041520	041411	051123	DH710:	.ASCIZ	/PC	CSR AD	CSR/
12294	064364	040440	004504	051503					
12295	064372	000122							
12296	064374	054503	044514	042116	DH711:	.ASCIZ	/CYLINDERS :/		
12297	064402	051105	020123	000072					
12298	064410	045522	051504	020040	DH204:	.ASCIZ	/RKDS	RKER	RKMR2
12299	064416	020040	045522	051105					RKMR3/
12300	064424	020040	020040	045522					
12301	064432	051115	020062	020040					
12302	064440	045522	051115	000063					
12303	064446	051105	047522	020122	DH502:	.ASCIZ	/ERROR TRYING TO GET FAILURE INFO/		
12304	064454	051124	044531	043516					
12305	064462	052040	020117	042507					
12306	064470	020124	040506	046111					
12307	064476	051125	020105	047111					
12308	064504	047506	000						
12309	064507	123	041505	047117	DH503:	.ASCIZ	/SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/		
12310	064514	020104	044524	042515					
12311	064522	052517	020124	041517					
12312	064530	052503	051122	042105					
12313	064536	043440	052105	044524					
12314	064544	043516	042040	044522					
12315	064552	042526	051440	040524					
12316	064560	052524	000123						
12317	064564	052516	041115	051105	DH800:	.ASCIZ	/NUMBER OF RETRIES:/		
12318	064572	047440	020106	042522					
12319	064600	051124	042511	035123					
12320	064606	000							
12321		064610							
12322	064610	001116	005500	005476	DT100:	.EVEN			
12323	064616	001162	001164	001166		.WORD	\$ERRPC, DRIVE, ERRCOM, \$REG0, \$REG1, \$REG2, \$REG3		
12324	064624	001170							
12325	064626	001256	001260		DT102:	.WORD	\$REG36, \$REG37		
12326	064632	001174	001176	001200	DT201:	.WORD	\$REG5, \$REG6, \$REG7, \$REG10, \$REG11, \$REG12, \$REG13		

12327	064640	001202	001204	001206		
12328	064646	001210				
12329	064650	001212	001214		DT202:	.WORD \$REG14,\$REG15
12330	064654	001216	001220	001222	DT203:	.WORD \$REG16,\$REG17,\$REG20,\$REG21,\$REG22,\$REG23,\$REG24,\$REG25
12331	064662	001224	001226	001230		
12332	064670	001232	001234			
12333	064674	001174	001176	001200	DT601:	.WORD \$REG5,\$REG6,\$REG7
12334	064702	001202	001204	001206	DT602:	.WORD \$REG10,\$REG11,\$REG12,\$REG13,\$REG14,\$REG15,\$REG16
12335	064710	001210	001212	001214		
12336	064716	001216				
12337	064720	005256	005254		DT103:	.WORD PRMPHO,PRMPLO
12338						
12339	064724	000006			DF01:	.WORD 6 ;NUMBER OF HEADER LINES
12340	064726	000				.BYTE 0 ;NUMBER OF COL FOR FIRST HDR
12341	064727	000				.BYTE 0 ;ALL CHARACTERS OCTAL
12342	064730	063243				.WORD DH101 ;SECOND HDR LINE
12343	064732	007	000			.BYTE 7,0 ;NUM OF COL-ALL OCTAL
12344	064734	063332				.WORD DH102
12345	064736	002	000			.BYTE 2,0
12346	064740	063123				.WORD DH200
12347	064742	000	000			.BYTE 0,0
12348	064744	063435				.WORD DH201
12349	064746	007	000			.BYTE 7,0
12350	064750	063140				.WORD DH500
12351	064752	000	000			.BYTE 0,0
12352						
12353	064754	000007			DF02:	.WORD 7
12354	064756	000	000			.BYTE 0,0
12355	064760	063243				.WORD DH101
12356	064762	007	000			.BYTE 7,0
12357	064764	063332				.WORD DH102
12358	064766	002	000			.BYTE 2,0
12359	064770	063123				.WORD DH200
12360	064772	000	000			.BYTE 0,0
12361	064774	063435				.WORD DH201
12362	064776	007	000			.BYTE 7,0
12363	065000	063524				.WORD DH202
12364	065002	002	000			.BYTE 2,0
12365	065004	063541				.WORD DH203
12366	065006	010	000			.BYTE 10,0
12367						
12368	065010	000010			DF03:	.WORD 10
12369	065012	000	000			.BYTE 0,0
12370	065014	063243				.WORD DH101
12371	065016	007	000			.BYTE 7,0
12372	065020	063332				.WORD DH102
12373	065022	002	000			.BYTE 2,0
12374	065024	063123				.WORD DH200
12375	065026	000	000			.BYTE 0,0
12376	065030	063435				.WORD DH201
12377	065032	007	000			.BYTE 7,0
12378	065034	063171				.WORD DH501
12379	065036	000	000			.BYTE 0,0
12380	065040	063524				.WORD DH202
12381	065042	002	000			.BYTE 2,0
12382	065044	063541				.WORD DH203

;"THE FOLLOWING REG DATA MB INCORRECT"

K03

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 244
TRAP TABLE

SEQ 0243

12383	065046	010	000	.BYTE	10,0	
12384						
12385	065050	000003		DF04: .WORD	3	
12386	065052	000	000	.BYTE	0,0	
12387	065054	063243		.WORD	0H101	
12388	065056	007	000	.BYTE	7,0	
12389	065060	063332		.WORD	0H102	
12390	065062	002	000	.BYTE	2,0	
12391						
12392	065064	000007		DF05: .WORD	7	;OPI, HVRC
12393	065066	000	000	.BYTE	0,0	
12394	065070	063243		.WORD	0H101	
12395	065072	007	000	.BYTE	7,0	
12396	065074	063332		.WORD	0H102	
12397	065076	002	000	.BYTE	2,0	
12398	065100	063635		.WORD	0H601	
12399	065102	000	000	.BYTE	0,0	
12400	065104	064062		.WORD	0H606	
12401	065106	000	000	.BYTE	3,0	
12402	065110	063637		.WORD	0H602	
12403	065112	000	000	.BYTE	0,0	
12404	065114	064062		.WORD	0H606	
12405	065116	003	000	.BYTE	3,0	
12406						
12407	065120	000007		DF07: .WORD	7	
12408	065122	000	000	.BYTE	0,0	
12409	065124	063243		.WORD	0H101	
12410	065126	007	000	.BYTE	7,0	
12411	065130	063332		.WORD	0H102	
12412	065132	002	000	.BYTE	2,0	
12413	065134	063715		.WORD	0H604	
12414	065136	000	000	.BYTE	0,0	
12415	065140	063743		.WORD	0H6041	
12416	065142	003	000	.BYTE	3,0	
12417	065144	064010		.WORD	0H605	
12418	065146	000	000	.BYTE	0,0	
12419	065150	064025		.WORD	0H6051	
12420	065152	003	000	.BYTE	3,0	
12421						
12422	065154	000005		DF10: .WORD	5	
12423	065156	000	000	.BYTE	0,0	
12424	065160	063243		.WORD	0H101	
12425	065162	007	000	.BYTE	7,0	
12426	065164	063332		.WORD	0H102	
12427	065166	002	000	.BYTE	2,0	
12428	065170	063715		.WORD	0H604	
12429	065172	000	000	.BYTE	0,0	
12430	065174	063743		.WORD	0H6041	
12431	065176	003	000	.BYTE	3,0	
12432						
12433	065200	000004		DF11: .WORD	4	
12434	065202	000	000	.BYTE	0,0	
12435	065204	063715		.WORD	0H604	
12436	065206	000	000	.BYTE	0,0	
12437	065210	063743		.WORD	0H6041	
12438	065212	003	000	.BYTE	3,0	

12439	065214	064130		.WORD	DH701	
12440	065216	000	000	.BYTE	0,0	
12441						
12442	065220	000006		DF12: .WORD	6	
12443	065222	000	000	.BYTE	0,0	
12444	065224	063243		.WORD	DH101	
12445	065226	007	000	.BYTE	7,0	
12446	065230	063332		.WORD	DH102	
12447	065232	002	000	.BYTE	2,0	
12448	065234	063123		.WORD	DH200	
12449	065236	000	000	.BYTE	0,0	
12450	065240	063435		.WORD	DH201	
12451	065242	007	000	.BYTE	7,0	
12452	065244	064410		.WORD	DH204	
12453	065246	004	000	.BYTE	4,0	
12454						
12455	065250	000006		DF13: .WORD	6	; FORMAT FOR 2ND LEVEL ERROR
12456	065252	000	000	.BYTE	0,0	; IN HEADER COMPARE ERROR
12457	065254	063243		.WORD	DH101	; AND 2ND LEVEL HEADER
12458	065256	007	000	.BYTE	7,0	; VRC ERROR
12459	065260	063332		.WORD	DH102	
12460	065262	002	000	.BYTE	2,0	
12461	065264	063635		.WORD	DH601	
12462	065266	000	000	.BYTE	0,0	
12463	065270	064062		.WORD	DH606	
12464	065272	003	000	.BYTE	3,0	
12465	065274	064446		.WORD	DH502	
12466	065276	000	000	.BYTE	0,0	
12467						
12468	065300	000006		DF14: .WORD	6	; FORMAT FOR 2ND LEVEL ERROR
12469	065302	000	000	.BYTE	0,0	; IN OPERATION INCOMPLETE ERROR
12470	065304	063243		.WORD	DH101	
12471	065306	007	000	.BYTE	7,0	
12472	065310	063332		.WORD	DH102	
12473	065312	002	000	.BYTE	2,0	
12474	065314	063635		.WORD	DH601	
12475	065316	000	000	.BYTE	0,0	
12476	065320	064062		.WORD	DH606	
12477	065322	003	000	.BYTE	3,0	
12478	065324	064446		.WORD	DH502	
12479	065326	000	000	.BYTE	0,0	
12480						
12481	065330	000011		DF15: .WORD	11	
12482	065332	000	000	.BYTE	0,0	
12483	065334	063243		.WORD	DH101	
12484	065336	007	000	.BYTE	7,0	
12485	065340	063332		.WORD	DH102	
12486	065342	002	000	.BYTE	2,0	
12487	065344	063123		.WORD	DH200	
12488	065346	000	000	.BYTE	0,0	
12489	065350	063435		.WORD	DH201	
12490	065352	007	000	.BYTE	7,0	
12491	065354	063524		.WORD	DH202	
12492	065356	002	000	.BYTE	2,0	
12493	065360	064507		.WORD	DH503	
12494	065362	000	000	.BYTE	0,0	

12495	065364	063171		.WORD	DH501
12496	065365	000	000	.BYTE	0,0
12497	065370	063541		.WORD	DH203
12498	065372	010	000	.BYTE	10,0
12499					
12500	065374	000011		DF16: .WORD	11
12501	065376	000	000	.BYTE	0,0
12502	065400	063243		.WORD	DH101
12503	065402	007	000	.BYTE	7,0
12504	065404	063332		.WORD	DH102
12505	065406	002	000	.BYTE	2,0
12506	065410	063123		.WORD	DH200
12507	065412	000	000	.BYTE	0,0
12508	065414	063435		.WORD	DH201
12509	065416	007	000	.BYTE	7,0
12510	065420	063524		.WORD	DH202
12511	065422	002	000	.BYTE	2,0
12512	065424	064446		.WORD	DH502
12513	065426	000	000	.BYTE	0,0
12514	065430	063171		.WORD	DH501
12515	065432	000	000	.BYTE	0,0
12516	065434	063541		.WORD	DH203
12517	065436	010	000	.BYTE	10,0
12518					
12519	065440	000005		DF17: .WORD	5
12520	065442	000	000	.BYTE	0,0
12521	065444	063243		.WORD	DH101
12522	065446	007	000	.BYTE	7,0
12523	065450	063332		.WORD	DH102
12524	065452	002	000	.BYTE	2,0
12525	065454	064374		.WORD	DH711
12526	065456	000	000	.BYTE	0,0
12527	065460	064217		.WORD	DH702
12528	065462	002	000	.BYTE	2,0
12529					
12530	065464	000002		DF20: .WORD	2
12531	065466	000	000	.BYTE	0,0
12532	065470	063373		.WORD	DH104
12533	065472	002	000	.BYTE	2,0
12534					
12535	065474	000002		DF21: .WORD	2
12536	065476	000	000	.BYTE	0,0
12537	065500	064025		.WORD	DH6051
12538	065502	003	000	.BYTE	3,0
12539					
12540	065504	000006		DF22: .WORD	6
12541	065506	000	000	.BYTE	0,0
12542	065510	063072		.WORD	DH100
12543	065512	000	000	.BYTE	0,0
12544	065514	063243		.WORD	DH101
12545	065516	007	000	.BYTE	7,0
12546	065520	063332		.WORD	DH102
12547	065522	002	000	.BYTE	2,0
12548	065524	063410		.WORD	DH106
12549	065526	000	000	.BYTE	0,0
12550	065530	063373		.WORD	DH104

CZ6ND0 RK611/06 SS VERIF 2
CZREND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 247
TRAP TABLE

SEQ 0246

12551	065532	002	000		.BYTE	2,0
12552						
12553	065534	000002		DF23:	.WORD	2
12554	065536	000	000		.BYTE	0,0
12555	065540	064564			.WORD	04800
12556	065542	001	000		.BYTE	1,0
12557						
12558	065544	000001		DF24:	.WORD	1
12559	065546	002	000		.BYTE	2,0
12560						
12561	065550	000000		DF25:	.WORD	0
12562	065552	007	000		.BYTE	7,0
12563						
12564	065554	000002		DF26:	.WORD	2
12565	065556	002	000		.BYTE	2,0
12566	065560	063541			.WORD	04203
12567	065562	010	000		.BYTE	10,0
12568						
12569	065564	000005		DF27:	.WORD	5
12570	065566	000	000		.BYTE	0,0
12571	065570	063243			.WORD	04101
12572	065572	007	000		.BYTE	7,0
12573	065574	063332			.WORD	04102
12574	065576	002	000		.BYTE	2,0
12575	065600	064233			.WORD	04703
12576	065602	000	000		.BYTE	0,0
12577	065604	064217			.WORD	04702
12578	065606	002	000		.BYTE	2,0
12579						
12580	065610	000005		DF30:	.WORD	5
12581	065612	000	000		.BYTE	0,0
12582	065614	063243			.WORD	04101
12583	065616	007	000		.BYTE	7,0
12584	065620	063332			.WORD	04102
12585	065622	002	000		.BYTE	2,0
12586	065624	064277			.WORD	04705
12587	065626	000	000		.BYTE	0,0
12588	065630	064257			.WORD	04704
12589	065632	004	000		.BYTE	4,0
12590						
12591	065634	000005		DF31:	.WORD	5
12592	065636	000	000		.BYTE	0,0
12593	065640	063243			.WORD	04101
12594	065642	007	000		.BYTE	7,0
12595	065644	063332			.WORD	04102
12596	065646	002	000		.BYTE	2,0
12597	065650	064313			.WORD	04706
12598	065652	000	000		.BYTE	0,0
12599	065654	064356			.WORD	04710
12600	065656	003	000		.BYTE	3,0

12601						
12602	065660				RWBUF:	:READ/WRITE DATA BUF <AT LEAST 512(L. WORDS>
12603						
12604	065660	005015	020040	020040	NOTMSG: .ASCII	<15><12>/ *** NOTE ***/<15><12><12>
12605	065666	020040	020040	020040		
12606	065674	025052	020052	047516		

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 248
TRAP TABLE

SEQ 0247

12607	065702	042524	025040	025052	
12608	065710	005015	012		
12609	065713	101	046114	042040	.ASCII /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12610	065720	044522	042526	020123	
12611	065726	047524	041040	020105	
12612	065734	042524	052123	042105	
12613	065742	046440	051525	020124	
12614	065750	040510	042526	035040	
12615	065756	005015	012		
12616	065761	061	020056	020101	.ASCII /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12617	065766	030062	047440	020122	
12618	065774	031062	051440	041505	
12619	066002	047524	020122	047506	
12620	066010	046522	052101	042524	
12621	066016	020104	040503	052122	
12622	066024	044522	043504	006505	
12623	066032	012			
12624	066033	062	020056	042510	.ASCII /2. HEADS MANUALLY LOADED/<15><12>
12625	066040	042101	020123	040515	
12626	066046	052516	046101	054514	
12627	066054	046040	040517	042504	
12628	066062	006504	012		
12629	066065	063	020056	042504	.ASCII /3. DESIRED PORT SELECTED/<15><12>
12630	066072	044523	042522	020104	
12631	066100	047520	052122	051440	
12632	066106	046105	041505	042524	
12633	066114	006504	012		
12634	066117	064	020056	051127	.ASCII /4. WRITE LOCK DISABLED/<15><12>
12635	066124	052111	020105	047514	
12636	066132	045503	042040	051511	
12637	066140	041101	042514	006504	
12638	066146	012			
12639	066147	065	020056	051104	.ASCII /5. DRIVE READY LIGHT ON/<15><12><12>
12640	066154	053111	020105	042522	
12641	066162	042101	020131	044514	
12642	066170	044107	020124	047117	
12643	066176	005015	012		
12644	066201	104	044522	042526	.ASCII /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12645	066206	020123	047516	020124	
12646	066214	047524	041040	020105	
12647	066222	042524	052123	042105	
12648	066230	046440	051525	020124	
12649	066236	040510	042526	041040	
12650	066244	052117	006510	012	

C04

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 249
TRAP TABLE

SEQ 0248

12651	066251	120	051117	051524	.ASCIIZ /PORTS DE-SELECTED./<15><12><12>
12652	066256	042040	026505	042523	
12653	066264	042514	052103	042105	
12654	066272	006456	005012	000	
12655					
12656	066277	015	044412	051516	HLPFIL: .ASCII <15><12>/INSTRUCTIONS FOR USING RK06-RK07 HEAD ALIGNMENT AID :/<15><12>
12657	066304	051124	041525	044524	
12658	066312	047117	020123	047506	
12659	066320	020122	051525	047111	
12660	066326	020107	045522	033060	
12661	066334	051055	030113	020067	
12662	066342	042510	042101	040440	
12663	066350	044514	047107	042515	
12664	066356	052116	040440	042111	
12665	066364	035040	005015		
12666	066370	025052	025052	025052	.ASCII /*****<15><12>
12667	066376	025052	025052	025052	
12668	066404	025052	025052	025052	
12669	066412	025052	025052	025052	
12670	066420	025052	025052	025052	
12671	066426	025052	025052	025052	
12672	066434	025052	025052	025052	
12673	066442	025052	025052	005015	
12674	066450	047515	047125	020124	.ASCII /MOUNT AN RK06 ALIGNMENT CARTRIDGE ON THE DESIRED /<15><12>
12675	066456	047101	051040	030113	
12676	066464	020066	046101	043511	
12677	066472	046516	047105	020124	
12678	066500	040503	052122	044522	
12679	066506	043504	020105	047117	
12680	066514	052040	042510	042040	
12681	066522	051505	051111	042105	
12682	066530	006440	012		
12683	066533	104	044522	042526	.ASCII /DRIVE, AND INSURE THAT THE DRIVE IS WRITE-LOCKED. /<15><12>
12684	066540	020054	047101	020104	
12685	066546	047111	052523	042522	
12686	066554	052040	040510	020124	
12687	066562	044124	020105	051104	
12688	066570	053111	020105	051511	
12689	066576	053440	044522	042524	
12690	066604	046055	041517	042513	
12691	066612	027104	006440	012	
12692	066617	103	047117	042516	.ASCII /CONNECT THE ALIGNMENT INDICATOR TO THE DESIRED DRIVE,/<15><12>
12693	066624	052103	052040	042510	
12694	066632	040440	044514	047107	
12695	066640	042515	052116	044440	
12696	066646	042116	041511	052101	
12697	066654	051117	052040	020117	
12698	066662	044124	020105	042504	
12699	066670	044523	042522	020104	
12700	066676	051104	053111	026105	
12701	066704	005015			
12702	066706	044526	020101	044124	.ASCII /VIA THE HEAD ALIGNMENT CABLE ONLY, AND CYCLE UP THE/<15><12>
12703	066714	020105	042510	042101	
12704	066722	040440	044514	047107	
12705	066730	042515	052116	041440	
12706	066736	041101	042514	047440	

12707	066744	046116	026131	040440	
12708	066752	043116	041440	041531	
12709	066760	042514	052440	020120	
12710	066766	044124	006505	012	
12711	066773	104	044522	042526	.ASCII /DRIVE. AFTER MOUNTING THE PACK ON THE DRIVE, /<15><12>
12712	067000	020056	043101	042524	
12713	067006	020122	047515	047125	
12714	067014	044524	043516	052040	
12715	067022	042510	050040	041501	
12716	067030	020113	047117	052040	
12717	067036	042510	042040	044522	
12718	067044	042526	020054	005015	
12719	067052	044124	020105	050117	.ASCII /THE OPERATOR SHOULD WAIT 30 MINUTES FOR THE DRIVE/<15><12>
12720	067060	051105	052101	051117	
12721	067066	051440	047510	046125	
12722	067074	020104	040527	052111	
12723	067102	031440	020060	044515	
12724	067110	052516	042524	020123	
12725	067116	047506	020122	044124	
12726	067124	020105	051104	053111	
12727	067132	006505	012		
12728	067135	124	046505	042520	.ASCII /TEMPERATURE TO STABILIZE, BEFORE PROCEEDING WITH /<15><12>
12729	067142	040522	052524	042522	
12730	067150	052040	020117	052123	
12731	067156	041101	046111	055111	
12732	067164	026105	041040	043105	
12733	067172	051117	020105	051120	
12734	067200	041517	042505	044504	
12735	067206	043516	053440	052111	
12736	067214	020110	005015		
12737	067220	046101	043511	046516	.ASCII /ALIGNMENT./<15><12><12>
12738	067226	047105	027124	005015	
12739	067234	012			
12740	067235	122	051505	047520	.ASCII /RESPOND TO ALL REQUESTS FOR PARAMETERS, BY ENTERING /<15><12>
12741	067242	042116	052040	020117	
12742	067250	046101	020114	042522	
12743	067256	052521	051505	051524	
12744	067264	043040	051117	050040	
12745	067272	051101	046501	052105	
12746	067300	051105	026123	041040	
12747	067306	020131	047105	042524	
12748	067314	044522	043516	006440	
12749	067322	012			
12750	067323	124	042510	042040	.ASCII /THE DESIRED PARAMETER VALUE (NO <CR> NEEDED)./<15><12>
12751	067330	051505	051111	042105	
12752	067336	050040	051101	046501	
12753	067344	052105	051105	053040	
12754	067352	046101	042525	024040	
12755	067360	047516	036040	051103	
12756	067366	020076	042516	042105	
12757	067374	042105	027051	005015	
12758	067402	044124	051105	020105	.ASCII /THERE ARE TWO MODES OF OPERATION : MANUAL MODE /<15><12>
12759	067410	051101	020105	053524	
12760	067416	020117	047515	042504	
12761	067424	020123	043117	047440	
12762	067432	042520	040522	044524	

12763	067440	047117	035040	020040	
12764	067446	040515	052516	046101	
12765	067454	046440	042117	020105	
12766	067462	005015			
12767	067464	046101	047514	051527	.ASCII /ALLOWS SELECTION OF DRIVES AND HEADS BY TTY INPUT, /<15><12>
12768	067472	051440	046105	041505	
12769	067500	044524	047117	047440	
12770	067506	020106	051104	053111	
12771	067514	051505	040440	042116	
12772	067522	044040	040505	051504	
12773	067530	041040	020131	052124	
12774	067536	020131	047111	052520	
12775	067544	026124	005015		
12776	067550	047101	020104	052501	.ASCII /AND AUTO MODE ALLOWS DRIVES AND HEADS TO BE SELECTED /<15><12>
12777	067556	047524	046440	042117	
12778	067564	020105	046101	047514	
12779	067572	051527	042040	044522	
12780	067600	042526	020123	047101	
12781	067606	020104	042510	042101	
12782	067614	020123	047524	041040	
12783	067622	020105	042523	042514	
12784	067630	052103	042105	005015	
12785	067636	054502	047440	043106	.ASCII /BY OFF-ON OPERATION OF DRIVE PORT SELECT SWITCHES. /<15><12>
12786	067644	047455	020116	050117	
12787	067652	051105	052101	047511	
12788	067660	020116	043117	042040	
12789	067666	044522	042526	050040	
12790	067674	051117	020124	042523	
12791	067702	042514	052103	051440	
12792	067710	044527	041524	042510	
12793	067716	027123	005015		
12794	067722	047111	042440	052111	.ASCII /IN EITHER MODE, UP TO 5 MINUTES OF SEEK EXERCISES /<15><12>
12795	067730	042510	020122	047515	
12796	067736	042504	020054	050125	
12797	067744	052040	020117	020065	
12798	067752	044515	052516	042524	
12799	067760	020123	043117	051440	
12800	067766	042505	020113	054105	
12801	067774	051105	044503	042523	
12802	070002	006523	012		
12803	070005	115	054501	041040	.ASCII /MAY BE REQUESTED FOR EACH DRIVE. /<15><12>
12804	070012	020105	042522	05252	
12805	070020	051505	042524	020101	
12806	070026	047506	020122	040503	
12807	070034	044103	042040	044522	
12808	070042	042526	020056	005015	
12809	070050	046101	047523	044440	.ASCII /ALSO IN EITHER MODE, A VERIFY OPTION ALLOWS HEAD /<15><12>
12810	070056	020116	044505	044124	
12811	070064	051105	046440	042117	
12812	070072	026105	040440	053040	
12813	070100	051105	043111	020131	
12814	070106	050117	044524	047117	
12815	070114	040440	046114	053517	
12816	070122	020123	042510	042101	
12817	070130	005015			
12818	070132	042523	042514	052103	.ASCII /SELECTION WITHOUT THE UNLOADING AND LOADING OF /<15><12>

12819	070140	047511	020116	044527
12820	070146	044124	052517	020124
12821	070154	044124	020105	047125
12822	070162	047514	042101	047111
12823	070170	020107	047101	020104
12824	070176	047514	042101	047111
12825	070204	020107	043117	006440
12826	070212	012		
12827	070213	124	042510	042040
12828	070220	044522	042526	041040
12829	070226	020131	044124	020105
12830	070234	051120	043517	040522
12831	070242	026115	053440	044510
12832	070250	044103	047440	044124
12833	070256	051105	044527	042523
12834	070264	047440	041503	051125
12835	070272	006523	012	
12836	070275	124	020117	046101
12837	070302	047514	020127	040515
12838	070310	044516	052520	040514
12839	070316	044524	047117	047440
12840	070324	020106	044124	020105
12841	070332	046101	043511	046516
12842	070340	047105	020124	047524
12843	070346	046117	006456	012
12844	070353	124	020117	042522
12845	070360	052123	051101	020124
12846	070366	044505	044124	051105
12847	070374	046440	042117	026105
12848	070402	052040	050131	020105
12849	070410	057074	037132	006456
12850	070416	012		
12851	070417	124	020117	042522
12852	070424	052123	051101	020124
12853	070432	046101	043511	046516
12854	070440	047105	020124	044501
12855	070446	026104	052040	050131
12856	070454	020105	057074	037122
12857	070462	006456	012	
12858	070465	124	020117	042523
12859	070472	042514	052103	047040
12860	070500	053505	042040	044522
12861	070506	042526	020123	047111
12862	070514	046440	047101	040525
12863	070522	020114	047515	042504
12864	070530	020054	054524	042520
12865	070536	036040	041536	027076
12866	070544	005015	012	
12867	070547	106	051117	044040
12868	070554	040505	020104	046101
12869	070562	043511	046516	047105
12870	070570	020124	051120	041517
12871	070576	042105	051125	026105
12872	070604	051040	043105	051105
12873	070612	052040	020117	044506
12874	070620	046105	020104	005015

.ASCII /THE DRIVE BY THE PROGRAM, WHICH OTHERWISE OCCURS/<15><12>

.ASCII /TO ALLOW MANIPULATION OF THE ALIGNMENT TOOL./<15><12>

.ASCII /TO RESTART EITHER MODE, TYPE <↑Z>./<15><12>

.ASCII /TO RESTART ALIGNMENT AID, TYPE <↑R>./<15><12>

.ASCII /TO SELECT NEW DRIVES IN MANUAL MODE, TYPE <↑C>./<15><12><12>

.ASCII /FOR HEAD ALIGNMENT PROCEDURE, REFER TO FIELD /<15><12>

G04

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 253
TRAP TABLE

SEQ 0252

.ASCIZ /TEST BOX (RK06-07TA, RK06-07TB) OPERATOR'S MANUAL./('15>('12'

12875	070626	042524	052123	041040
12876	070634	054117	024040	045522
12877	070642	033060	030055	052067
12878	070650	026101	051040	030113
12879	070656	026466	033460	041124
12880	070664	020051	050117	051105
12881	070672	052101	051117	051447
12882	070700	046440	047101	040525
12883	070706	027114	005015	000
12884				
12885				
12886				
12887	000001			

.END

SEQ 0253

[illegible]

	APTSPO=	000100	11069	11692	11733#												
	ASKDRV	025016	6420#	6462	6466	6481	6513	6583									
	ASKHED	025442	6499#	6527	6539	6546											
	ASKMDE	016020	5137#	5143	5144	5149	5164	5169	5174	5199	5213	5224	5233	5249	5268		
			5277#	5303	5326	5398	7955										
	ASKMOD	024724	6398#	6412													
	ASKTMD	015254	5019#	5022	5023	5028	5052	5060	5066	5082	5093	5102	5127				
	ASKTST	015250	4935	5010#													
	ASKTYP	024620	6367	6373	6376#												
	ASTART	024134	2219	6292#													
	ASWREG=	000000	2348	2361													
	AS2SP2	060370	9223	11902#													
	ATESTN=	000000	2348	2352													
	AUNIT =	000000	2348	2355													
	AUSWR =	070000	2348	2362													
	AUTAL	026266	6610	6617	6624#	6745	6766										
	AUTALN	012752	4632#	6629													
	AUTEX	027520	6613	6824#	6845	6886											
	AUTEXR	013066	4646#	6827													
	AUTO	026144	6410	6596#	6645	6813	6816	6819	6835	6909	6924						
	AUTOOR	013176	4659#	6890*													
	AUTOEX	013157	4656#	6891													
	AUOFG	003150	3689#	4748*	6336*	6418*	6597*	7462	7478	7488	7511						
	AUTVRF	013010	4637#	6627													
	AVECT1=	120210	2255#	2348	2387	4799											
	AVECT2=	000000	2348	2388													
	A.ABNL	003046	3583#	5606*	5609*	6657*	6726*	6848*	6881*	7375*	7455*	7459*	7550*	7554*	8495*		
			8791*	8808*	8817*	9437*	9693*	9706*	9708*	10635							
	A.CONT	003050	3584#	10645													
	A.NORM	003044	3582#	6172*	6233*	7376*	9438*	9694*	10639								
	BADDRV	007522	4312#	6380*	6381	7464*	7465*	7466	7480*	7481*	7482	7490*	7491*	7492	7519*		
			7520*	7521													
	BADSEC=	001000	4248#	8404	8627	9080	9088	9639	9742								
	BAD632	013202	4661#	7516													
	BAI =	000020	3261#	10804													
	BAIA	025306	6460	6463#													
	BAIFLU	025300	6428	6430	6461#												
	BA16 =	000400	3237#														

BIT10 = 002000	2127#	3239	3268	3290	3325	3339	3352	3367	3381	3460	3601	4249	9353
BIT11 = 004000	11501												
BIT12 = 010000	2126#	3240	3269	3291	3310	3326	3340	3353	3368	3382	3461	4250	11661
	2125#	3242	3270	3292	3327	3341	3354	3355	3369	3383	3462	4251	9235
	9263												
BIT13 = 020000	2124#	3243	3271	3293	3311	3328	3342	3356	3370	3384	3463	7355	7357
	8301	8304	8375	8378	8574	8577	8684	8687	11508				
BIT14 = 040000	2123#	3244	3272	3294	3312	3329	3343	3357	3371	3385	3464	3602	11636
BIT15 = 100000	2122#	3245	3246	3273	3295	3313	3330	3390	3465	3603	4252	5210	7885
	9356												
BIT2 = 000004	2145#	3253	3280	3281	3301	3449	3593	3685	4241	9328			
BIT3 = 000010	2144#	3259	3260	3282	3302	3450	3594	3686	4242	9335			
BIT4 = 000020	2143#	3254	3261	3283	3303	3304	3319	3361	3375	3451	3595	3687	4243
	5493	9339											
BIT5 = 000040	2142#	3262	3263	3284	3305	3320	3334	3347	3362	3376	3452	3596	4244
	6959	9200	9343	10716									
BIT6 = 000100	2141#	3235	3264	3285	3306	3321	3335	3348	3363	3377	3453	3598	4245
	5059	5212	5232	6254	6568	6922	6964	6997	8447	8676			
BIT7 = 000200	2140#	3236	3265	3286	3307	3308	3322	3336	3349	3364	3378	3455	4246
	4797	4971	5469	5472	5623	5628	7468	9839	9842				
BIT8 = 000400	2139#	3237	3266	3287	3288	3309	3323	3337	3350	3365	3379	3456	3599
	4247	9347											
BIT9 = 001000	2138#	3238	3267	3289	3324	3338	3351	3366	3380	3457	3600	4248	5478
	7143	9350	11613										
BKSLSH 013303	4673#	7751	7759										
BLDEXH 047026	9590	9615	9886#										
BPTVEC= 000014	2154#												
BSE = 000200	3286#	8822	9630	10069	10235	10471							
BSERR = 000002	4240#	6022	6175	9002	9632	9642	9735						
BSFACT 004224	3707#	6093	6096	8790	8838	9071							
BSSOFT 003224	3706#	6085	8802	8871	9076	9078							
BUFFD 005264	3716#	4843	4875	4878	4939	4943	4945	4950	4961	4965	5026	5029	5032
	5035	5070	5072	5103	5106	5108*	5113	5131	5147	5150	5153	5156	5159
	5253	5255	5287	5304	5330	5337	5347	5349	5352	5357	5366	5404	5407
	7088	7092	7754	7768*	7773*	7774	7778*	7895	7897	7917	7920	7922*	7927
	7943												
B.BA16= 000001	3251#	10798											
B.BA17= 000002	3252#	10798											

SEQ 0256

[illegible]

CZR6NDO RK611/06 SS VERIF 2
CZR6NDO.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 259
CROSS REFERENCE TABLE -- USER SYMBOLS

DF16	065374
DF17	065440
DF20	065464
DF21	065474
DF22	065504
DF23	065534
DF24	065544
DF25	065550
DF26	065554
DF27	065564
DF30	065610
DF31	065634
DH100	063072

2689	12500#												
2707	12519#												
2875	12530#												
2785	12535#												
2881	12540#												
2791	2797	2803	12553#										
2809	12558#												
2713	8338*	8392*	8606*	8632*	12561#								
2863	12564#												
2839	12569#												
2893	7161*	12580#											
2899	7204*	7205*	12591#										
2411	2417	2423	2429	2435	2441	2447	2453	2459	2465	2471	2477	2483	
2489	2495	2501	2507	2513	2519	2525	2531	2537	2543	2549	2555	2561	
2567	2573	2579	2597	2603	2609	2615	2621	2627	2633	2639	2645	2651	
2657	2663	2669	2675	2681	2687	2705	2717	2723	2729	2735	2741	2747	
2753	2759	2765	2771	2813	2819	2825	2831	2837	2849	2855	2885	2891	
2897	12159#	12542											
9239	12180#	12342	12355	12370	12387	12394	12409	12424	12444	12457	12470	12483	
12502	12521	12544	12571	12582	12593								
9250	12190#	12344	12357	12372	12389	12396	12411	12426	12446	12459	12472	12485	
12504	12523	12546	12573	12584	12595								
2873	12193#												
12197#	12532	12550											
9237	12162#												
12200#	12548												
12164#	1246	12359	12374	12448	12487	12506							
12204#	1248	12361	12376	12450	12489	12508							
12214#	12363	12380	12491	12510									
12217#	12365	12382	12497	12516	12566								
12298#	12452												
12167#	12350												
12172#	12378	12495	12514										
12303#	12465	12478	12512										
12309#	12493												
12228#	12398	12461	12474										
12232#	12402												
12238#	12413	12428	12435										
8341	12242#	12415	12430	12437									
2807	12246#												
2783	12249#	12417											
12552#	12419	12537											
12557#	12400	12404	12463	12476									
12261#													
8337*	8391*	8605*	8631*	12264#	12439								
2861	12274#	12527	12577										
12277#	12575												

SEQ 0259

[illegible]

[illegible]

4491# 6364

```
IE      = 000100
ILC     = 000001
ILF     = 000004
INCRMA  041052
INCRSC  041646
INITMM  031160
INITSS  032310
```

```
INPUTP 016014
INTCHR 005522
```

INTG 026430
INTMSK 003101

```
INTR      = 000300
IOTVEC    = 000020
IR         = 000100
I.AERR    051166
I.ATTN    050516
I.CCLR     051552
I.CSTS     052204
I.CST1     052226
I.DRV      003102
I.ERR      047740
I.ERRA     047720
I.ERRC     047726
I.HDAL     050164
I.INTR     047304
I.ISRL     003074
I.ISSU     051634
I.IOO      047564
I.RTRN     051534
```

```
I. STAT      052310
I. STOR      052056
I. UNLD      051434
KBDMDL      030436
KILLDR      000170
KIPAR0=     172340
KIPAR1=     172342
KIPAR2=     172344
KIPAR3=     172346
KIPAR4=     172350
KIPAR5=     172352
KIPAR6=     172354
KIPAR7=     172356
KIPDR0=     172300
KIPDR1=     172302
KIPDR2=     172304
KIPDR3=     172306
KIPDR4=     172310
KIPDR5=     172312
```

[illegible]

SEQ 0263

2183#												
2184#												
4773	7152#											
3746#	8222	8396*	8619*									
3807#	5436*	5448*	5552	5553	6437*	6448*	6459*	6664*	6675*	6695*	6854*	6865*
5434*	5446*	5499#	6435*	6446*	6662*	6673*	6852*	6863*	8705	8707		
3695#												
5765	8554#											
5769	5845	5849	5944	8558#								
8557	8560#											
4247#	9434	9489	9586	9592	9611	9617	9709	9769				
2061#	11117	11123										
3691#												
5940	8269#											
4919	5071	5076*										
5083	5254	5259#										
5905	8198#											
8160#	8182											
2200#	7201											
3717#	5290*	5370*	7824*	7829	7986	7988	7991	7994	7999	8004	8007	
3725	5476*	5482*	5548*	5572	5574	5667	5669	5684*	5688*			
5437*	5449*	5500#	5553	5563	6089	6443*	6454*	6459	6665*	6676*	6695	6855*
6866*	8094	8149	8782									
4267#	7057											
4238#	7506	8095										
5031	5040#											
3809#	5170											
3749#	5751	5752	5758	5759	5830	5831	5837	5838	5919	5920	5926	5927
8031	8033	8708*	8711*	8712*	8713*	8714*	8766*	8767*	8768*	8975	8976	
3747#	7029*	7030*	7044*	7045*	7058	7278	7282	7284	8029	8030		
6483	6496#											
4591#	6497											
6486	6551#											
4580#	6556	6590										
6407	6417#											
6489	6544#											
4597#	6545											
2198#	9153											
2197#	7260	9154										
4397#	8011											
4229#												
4231#												
4233#												
4235#												
4228#												
4230#												
4232#												
4234#												
3323#												
3657#	4692*	4701*	4793	4917	5081	5264	5545	6355*	7671	8483		
4430#	7888											
3267#	7441	10033	10460	10835								
3679#	7211											
2199#	7175	7181	7222	7230								

CZR6NDD RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 266
CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

SEQ 0265

[illegible]

H05

CZR6NDO RK611.06 SS VERIF 2
CZR6NDO.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 268
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0266

```
PIP      = 020000
PIRQ     = 1777772
PIRQVE   = 000240
PKRB     003160
PKS      003154
PKSB     003156
PLOFST   005546
PMA      005604
```

PRDFLT 005706

PREPAR 031646
PREPKB 030576

PRMBUF	013316
PRMINP	007352
PRMLIM	005734

```
PRMLST      005660
PRMNEM      006010
PRMPHO      005256
PRMPLO      005254
PRMPSP      013337
PROMPT      013334
PRVCMO      005224
```

```
PR0      = 0000009
PR1      = 0000040
PR2      = 0000100
PR3      = 0000140
PR4      = 0000200
PR5      = 0000240
PR6      = 0000300
PR7      = 0000340
PS       = 1777776
```

```
PSTART 013376
PSW    = 177776
PT      005700
PWRMSG  057122
PWRVEC= 000024
P.ASOF= 000032
P.A00 = 000040
P.A01 = 000044
P.A10 = 000050
P.A11 = 000054
P.BAHI= 000007
P.BALO= 000010
```

```
P.BAR = 000024
P.B00 = 000042
P.B01 = 000046
P.B10 = 000052
P.B11 = 000056
P.CMND= 000001
```

3311
2067
2161
3694
3692
3693
3739
3751
8561
3820
6867
5760
5042
6472
4681
4291
3836
6853
3805
3863
3713
3712
4685
4684
3708
2084
2085
2086
2087
2088
2089
2090
2091
2064
10944
2217
2065
3814
11596
2156
3478
3481
3483
3485
3487
3440
3441
8950
3475
3482
3484
3486
3488
3434
5945
6558

5597*
5758*
8603
5208

5839
5189
6500
7822*
4811
5435*
6857*
5207
5202
9151*
9152*
5322
4925
4777
4776

4769
3581

4704
2065

4701*
9762*
5210
11600*
4719*
9390
7476
6755
9397
8059
5754*
5751*
9382
9389
9394
9396
6196
9400
5466*
6018*
6569*

5632*
5759*
8670
5259

5928
5201
6559
7823

5439*
6864*
5260
5281
12337
12337

5089
6358
5418

6345

4753
4704*

9834*
5906

4720*
10297*
7486
6778
10605*
9399
5833*
5830
10784
10328*
10283*
10599*
9398
10204*
5600*
6024*
6576*

5652
5837*
8672
5262

7242*
5226
6602

5447*
6868*
5300
5337

5140
9140
6349

4774
4776*

6010

6307*
10298
7563
7624

10612*
5922*
5919*
10881
10558*
10284

9824
10613*
5603*
6029*
6581*

5656
5838*
8947*
5301

8956
5543
6630

5451*
7986
5341
7822

5273
9403
7652

6292
5418*

7885

6308*
10331*
7580
9395

7268*
6015*

10620*
10606*

5605*
6106*
6586*

5926*
8948*
5342

9014
5732
6654

6436*
7988
7824
7826

7905

9762

6348
6292*

11584*
10562*
9393
10598*

8953*
6027*

10828*

5610*
6134*
6689*

5927*
8949*
5438*

5809
6808

6439*
7996
7828
7882

9834

7157
6349*

11592*
10750*
9802

9379
6132*

5612*
6169*
6743*

7244
8950
5450*

5897
6826

6447*
7999
7991*
7979

7199
7652*

11593*
10751
10288*

7267*

5681*
6177*
6753*

7246
8951
6438*

5998
6894

6450*
8071
8007*

11593
9955

10289

7507*

5766*
6181*
6757*

8290
8971*
6449*

6081
6911

6663*
8004
8008*

11694
961*

10305*

8096*

5770*
6232*
6774*

8361
8972*
6666*

6385
6995*

6667*

9975*

10306

8533*

5846*
6506*
6776*

8441
8975*
6677*

6399
7627

6674*

10684

10619*

8790*

5850*
6511*
6780*

8443
8976*
6856*

6421
8514

6678*

10685*

10827*

8802*

5941*
6516*
6789*

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 269
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0267

P.CS1 = 000016

P.CS1H= 000007

P.CS2 = 000020

P.CYLN= 000002

P.DCYL= 000030

P.DRVN= C00000

P.DS = 000036

P.DTS = 000026

P.EPAT= 000062

P.EPOS= 000060

P.ER = 000034

P.OFST= 000006

P.PRST= 000014

P.SECT= 000004

P.TRCK= 000005

P.WC = 000012

P.WCR = 000022

QNEWSW= 000000

G.INIT= 040000

RCDASW= 000010

RCLREQ= 004000

RDALHD= 000164

RDCHR = 104406

RDCHRS 034034

RDDATA= 000121

RDGATE= 100000

ROHDO 046536

ROHEAD= 000125

ROSTAT= 000141

RDY = 000200

RECAL = 000113

RECODE 005474

6878*	6892*	6902*	6907*	6914*	7381*	7443*	7454*	7496*	7498*	7500*	7549*	7575*
7588*	7590*	7601*	7620*	7622*	7634*	7650*	8055*	8093*	8183*	8497*	8536*	8541*
8781*	9370	9705*	9752*	9797*	9833*	9836*	10099	10716	10720	10732	10742	10748
10754	10757	10763	10788	10795	10814	10861	10878	10900	10909			
3472*	6227*	7401	7444*	9383	9501	9519	9865	9898	10120*	10172*	10210*	10211*
10213*	10214	10323*	10325*	10552*	10555*	10706	10760*	10761*	10763*	10767*	10768	10792*
10795*	10797*	10798*	10808*	10809	10817*	10818*	10820*	10821	10823*	10824	10829	10868*
10869*	10871*	10874*	10875	10891*	10892*	10894*	10897	10916*	10917	10926*	10927	
3439*	6161*	6163*	6215*	6217*	7385*	7386*	7509*	7510*	8089*	8780*	9844*	9845*
10210	10451	10760	10797	10817	10868	10882	10891					
3473*	5694	6930	9384	9458	9463	9468	9470	9476	9478	9526	9576	9660
10126	10326*	10556*	10714*	10739	10804*	10805	10816	10835	10839	10865*	10866	10890
3435*	5549*	5559*	5562*	5563	5568*	5577	5579*	5587*	5643	5671	5673*	5679*
6000*	6109*	6126*	6157*	6171*	6561*	6896*	7503*	7505*	7602*	7604*	7607	7610
8094*	8250	8702*	8707*	8782*	8884	8938*	9372	9832*	10208	10744	10786	10888
3477*	8888	8934	9385	9633	9855	9887	10330*	10564*				
3433*	6116*	6156*	6170*	6194*	6638*	6688*	6752*	6770*	6784*	6849*	7383*	7430*
7548*	10096	10273	10449	10688								
3480*	9391	9511	9515	9566	10333*	10560*	10841					
3476*	8890	8892	8935	8936	9386	9634	9636	9823	9858	9859	9888	9894
10329*	10559*											
3490*	9657	10122*	10708									
3489*	9656	10123*										
3479*	5697	8822	9392	9457	10332*	10561*						
3438*	5602*	5613*	10750									
3443*	6007*	6133*	9484	9497	9669	9674	9679	9684	9799	9968*	10081*	10115*
10143*	10150	10154*	10251*	10274	10278*	10291*	10309*	10319*	10345	10349*	10359*	10371*
10476*	10628*	10703*	10740*	10764*	10765	10790	10796*	10802	10806	10833	10843*	10851
10872	10895	10904	10938									
3436*	5550*	5572	5574*	5580*	5588*	5648	5667	5669*	5674*	6002*	6127*	8092*
8177	8188*	8227	8703*	8787*	8796*	8797	8803*	8813*	8814	8886	8940*	9375
9831*	10209	10745	10787	10889								
3437*	5551*	5565	5569*	5570	5575*	5586*	5637	5676*	5677	6001*	6532*	6798*
7506*	8095*	8178*	8180	8185*	8186	8241	8704*	8783*	8885	8939*	9373	
3442*	5746*	5750*	5825*	5829*	5914*	5918*	6003*	6006*	6128*	6131*	7508*	8097*
8534*	8758	8789*	8943*	8969	9006	9377	10785					
3474*	9387	10327*	10557*									
3193												
3464*												
3686*												
4250*	9504	9509	9554	9559	9564	9569	9703					
3222*	10099	10878										
6369	7717	11895*										
4869	4927	5020	5064	5091	5141	5247	5275	5324	5396	7079	7711*	7889
7907												
3211*	5770	5850	5945	6029	7500	8093	8781					
3330*												
9591	9616	9821*										
3213*	5466	9836	10213	10894								
3221*	6181	6753	6776	7454	7549	7622	8055	9797	10814			
3236*	6224	9699	10771	10824								
3208*	5681	6506	6511	6516	6569	6576	6581	6586	6789	6902	6907	6914
7498	7588	8497	9705	10757								
3719*	6022	6175	7448*	7514	7647*	8404	8627	8999	9002	9080*	9088*	9176
9186*	9433*	9434	9461*	9466*	9487	9489	9504*	9509*	9549*	9554*	9559*	9564*
9569*	9574*	9582*	9586	9592	9607*	9611	9617	9632*	9639	9642*	9646*	9649*

J05

CZR6NDO RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 270
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0268

	9602*	9663*	9672*	9677*	9682*	9687*	9695	9701*	9703	9709	9711*	9735	9737*
	9742	9749*	9769*	9801*									
REDBSF 041120	5485	8778*											
REFNEM 040750	6019	6030	8729*										
REISSU 003137	3674*	9332*											
RELEAS= 000140	3220*	10861											
REPSUP 043720	6035	6192	6206	6228	7152	7196	7446	8341	8607	8799	9367*	9439	
RESREG= 104410	7063	7144	7269	7323	8020	8078	8165	8189	8206	8253	8322	8403	8464
	8543	8583	8626	8694	8769	8874	8899	8957	8985	9081	9131	9163	9303
	9418	9764	9813	9847	9906	11241	11293	11897*					
RESVEC= 000010	2151*												
RETANL 046340	9437	9706	9768*										
RETNML 046356	5701	6945	8826	9438	9771*								
RKASOF= 000016	3189*	9790	10109	10179	10259	10514	10562	10751*	10776				
RKBA = 000004	3184*	9789	10510	10558	10784*								
RKBADR 007400	4295*	4815											
RKBAS 003036	3579*	4796*	4800*	4814	4817*	7377	9594	9619	9698	9751	9782	9962	10031
	10695												
RKCS1 = 000000	3182*	6740*	7472*	7570*	9595*	9599	9620*	9624	9784	10054	10106	10164	10214*
	10223	10405*	10406	10415*	10454*	10455	10479*	10507	10768*	10770	10809*	10821*	10823
	10853*	10875*	10897*	10906*	10916	10926	10941*						
RKCS2 = 000010	3186*	7432*	7441	9785	10032	10192	10450*	10508	10556	10739*	10805*	10816*	10866*
	10890*	10915*											
RKDA = 000006	3185*	9787	10209*	10511	10559	10745*	10787*	10889*					
RKDB = 000024	3192*	5477	5478	9596	9597	9598	9621	9622	9623	10188	10189	10190	
RKDC = 000020	3190*	9786											
RKDCYL= 000020	3191*	10208*	10515	10564	10744*	10786*	10888*						
RKDS = 000012	3187*	9699	9791	10057	10512	10560							
RKECPS= 000030	3197*	10123	10519										
RKECPT= 000032	3199*	10122	10520										
RKER = 000014	3188*	7450	9792	10066	10232	10513	10561						
RKMR1 = 000026	3193*	10200*	10279*	10516	10593*	10600*	10607*	10614*	10715*				
RKMR2 = 000034	3194*	10517	10827										
RKMR3 = 000036	3195*	10518	10828										
RKPAT = 000032	3198*												
RKPOS = 000030	3196*												
RKPRI 003042	3581*	4772	4805*	4824	4840*	9961	10685						
RKPRTY 007446	4303*	4829											
RKVADR 007423	4299*	4820											
RKVEC 003040	3580*	4770	4801*	4802*	4819	4822*	6346						
RKWC = 000002	3183*	9788	10509	10557	10785*								
RLS = 000010	3260*	10865											
RNDADR 035666	6140	6560	6895	8146*									
RNDLNG= 016514	4237*	6557	6842										
RNDGHT= 002734	4236*	6838											
RUNTST 017224	5175	5393*											
RWBUF 065660	3749	5590	5735	5812	5900	6027	6028*	6033	6037	7209	7507	7517	8096
	8101	8161	8163	8275	8346	8527	8533	8708	8971	12602*			
RWNRDY 012651	4619*	7626											
R.ABNL 052536	9974	10084	10159	10176	10354	10374	10634*	10844					
R.CONT 052562	10036	10074	10138	10147	10218	10240	10263	10269	10302	10342	10366	10411	10463
	10486	10624	10642*	10858	10912	10921	10934						
R.NORM 052550	10112	10205	10314	10638*	10781	10854	10942						
SAVPA 003176	3702*	7256*	8294	8365	8566	8612							
SAVPRS 005552	3741*	8968	8979										
SAVREG= 104407	7024	7130	7242	7302	7305	7976	8053	8160	8176	8198	8221	8269	8336

SEO 0269

[illegible]

CZR6ND0 RKL11/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 272
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0270

SWREG	000176	2211#	4739	6327	6342	7110	7112	7115*
SWMSG	007466	4306#	7113					
SW0	= 000001	2119#						
SW00	= 000001	2109#	2119					
SW01	= 000002	2108#	2118					
SW02	= 000004	2107#	2117					
SW03	= 000010	2106#	2116					
SW04	= 000020	2105#	2115					
SW05	= 000040	2104#	2114					
SW06	= 000100	2103#	2113					
SW07	= 000200	2102#	2112					
SW08	= 000400	2101#	2111					
SW09	= 001000	2100#	2110					
SW1	= 000002	2118#						
SW10	= 002000	2099#						
SW11	= 004000	2098#						
SW12	= 010000	2097#						
SW13	= 020000	2096#	9208					
SW14	= 040000	2095#						
SW15	= 100000	2094#						
SW2	= 000004	2117#						
SW3	= 000010	2116#						
SW4	= 000020	2115#						
SW5	= 000040	2114#						
SW6	= 000100	2113#						
SW7	= 000200	2112#						
SW8	= 000400	2111#						
SW9	= 001000	2110#						
S. ACLO=	000100	3348#						
S. BRHM=	000100	3363#						
S. BRKE=	040000	3385#						
S. CART=	000400	3365#						
S. DCLO=	010000	3355#						
S. DIB =	002000	3381#						
S. DOOR=	000200	3364#						
S. DRA =	000040	3334#						
S. DROT=	020000	3356#						
S. DRY =	000200	3336#	7476	7563				
S. DSC =	040000	3343#	10141	10289	10306	10369		
S. FLT =	000200	3349#	10284					
S. FORM=	001000	3338#						
S. FWD =	002000	3367#						
S. HDFL=	000200	3378#						
S. HDHM=	000040	3362#	6755	6778	7624			
S. ICYL=	000040	3347#						
S. ILF =	000400	3350#						
S. LIMD=	020000	3384#						
S. LOAD=	010000	3369#						
S. MHD =	000400	3379#						
S. NMOV=	010000	3383#						
S. OFF =	002000	3339#						
S. PAR =	001000	3351#	10144	10621				
S. PIP =	020000	3342#	10152	10347				
S. PLO =	004000	3382#						
S. REV =	004000	3368#						
S. RTZ =	020000	3370#						

MOS

CZR6N00 RK611/06 SS VERIF 2
CZR6N0.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 273
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0271

[illegible]

SEQ 0272

6497	6499	6534	6538	6545	6556	6571	6590	6599	6601	6627	6629	6739
6764	6800	6804	6827	6891	6923	6963	6969	6973	6977	6979	6980	7011
7012	7057	7061	7062	7078	7088	7089	7113	7466	7467	7482	7483	7492
7493	7516	7521	7522	7560	7568	7585	7626	7722	7728	7734	7740	7751
7755	7759	7767	7772	7774	7775	7777	7796	7803	7823	7848	7854	7888
7897	7898	7901	7904	7905	7943	7944	8011	8017	8057	8058	8077	8099
8100	8104	8507	8508	8839	8840	8841	8842	8848	8849	8853	8859	8863
8872	9206	9222	9223	9224	9228	9229	9234	9237	9238	9239	9240	9245
9248	9249	9250	9251	9252	9255	9258	9259	9261	9275	9280	9284	9290
9295	9296	9298	11087	11334	11399	11474	11503	11511	11596	11888#		
5431#												
9199#	11510											
3663#	4992	5433*	5445*	6434*	6444	6661*	6671	6851*	6861	7386	7428*	7433
7452*	7501	7510	7605									
5152	5187#											
4509#	6419											
4503#	6398											
6383	8852	8858	8862	9244	9254	9257	9277	11889#				
5650	11891#											
5639	5644	7075	7800	7805	7851	7856	9231	11890#				
5227	7847#	7903										
5190	5271	7822#										
5043	5087	7796#										
3566#	6230	10109*	10110	10134	10179*	10180	10259*	10260	10266	10297	10331	10339
10363	10514*	10776*	10777									
3563#	10328	10510*										
3558#	6224	6227	10054*	10106*	10107	10120	10131*	10164*	10166	10172	10201*	10223*
10225	10255	10280*	10294*	10324*	10325	10336*	10360*	10406*	10407	10447	10451*	10452*
10454	10458	10467	10507*	10554*	10555	10595*	10602*	10609*	10616*	10770*	10771	10773
3560#	10032*	10033	10048	10051	10063	10092	10192*	10193	10229	10243	10326	10448*
10449*	10450	10460	10469	10508*								
3564#	10329	10511*										
3575#												
3565#	10330	10515*										
3569#	10057*	10058	10333	10512*								
3568#	10066*	10069	10232*	10235	10332	10471	10513*					
3570#	10516*											
3571#	6631*	10141	10152	10288	10305	10347	10369	10517*	10598	10605	10612	10619
3572#	10144	10204	10283	10518*	10599	10606	10613	10620	10621			
3574#	10520*											
3573#	10519*											
3562#	10327	10509*										
3678#	5778	5866										

[illegible]

CZR6ND0 RK611/06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05MACY11 30A(1052) 10-JAN-78 12:02 PAGE 276
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0274

		2329*	2330*	2331*	2332*	2333*	2334*	2335*	2336*	2337*	2338*						
SCM4 =	000011																
SCNTLG	057005	11573*															
SCNTLU	057000	11572*															
SCORE	060102	11764*	11791*														
SCPUOP	001346	2363*															
SCRLF	001315	2342*	4841	4911	5044	5191	5228	5665	6376	6379	6980	7061	7062	7772			
		7777	8017	8077	8104	8842	8849	8863	9228	9238	9240	9248	9251	9261			
		9284	9290	9298	11088	11123	11511	11535									
SCROUT	060132	11791*	11798*														
SDBLK	056454	11440	11474	11482*													
SDB2D	055556	11267*															
SDB20	055436	7059	7830	8015	8102	11226*											
SDECVL	055736	11269	11315*														
SDEVCT	001330	2354*	5416*	6251*													
SDEVN	001376	2390*	4970														
SDOAGN	024056	6261	6270	6276*													
SOTBL	056444	11443	11478*														
SENDAD	024046	2228	6272*	11530													
SENOCT	024014	4721	6263*	6309													
SENOBG	024065	6265	6280*														
SENULL	024062	6268	6279*														
SENV	001340	2359*	6283	11067	11513	11690	11714										
SENVN	001341	2360*	4744	4797	4971	6332	7468	11069	11074	11692							
SEOP	023742	6078	6249*														
SEOPCT	024006	4721*	5266*	5410*	6260*	6264	6309*										
SERFLG	001103	2268*	11498*	11535	11611	11625	11650	11652	11658*	11678							
SERMAX	001115	2274*	4724*	4755*	6312*	7668*	11652	11673*	11678								
SERROR	056464	4715	6303	11497*													
SERRPC	001116	2275*	11505*	11506*	11507	11535	12322										
SERRTB	001400	2407*	9219														
SERTTL	001112	2272*	11504*	11535													
SESCAP	001306	2339*	4723*	6311*	11526	11528	11535	11672*									
SETABL	001340	2358*															
SETEND	001400	2254	2391*														
SFATAL	001322	2351*	11718*														
SFFLG	057672	11681*	11684*	11712	11721*	11729*											
SFILLC	001156	2293*	11092	11123													
SFILLS	001155	2292*	11123														
SGOADR	001120	2276*															
SGDDAT	001124	2278*															
SGET42	024036	6269*															
SGTSWR=	***** U	11894															
SHD =	000000	2029															
SHIBTS	001000	2249*															
SHINUM	054732	4786*	6353*	8203	11024	11032	11037*	11042*									
SHIOCT	054632	5291	5371	7096	7931	10994*	11007*										
SICNT	001104	2269*	11663*	11664	11666*	11677											
SINTAG	001135	2283*															
SITEMB	001114	2273*	9214	11507*	11515	11535											
SKTNEX	060074	11765	11790*														
SKTOUT	060064	11782	11787*														
SKT11	057732	5755	5834	5923	7025*	7027	7316	8291	8362	8451	8460	8562	8572	8580			
		8608	8629	8682	8691	8954	9012	11763*	11767*	11790*							
SLF	001316	2343*	11123	11535													
SLFLG	057671	11722*	11728*														
SLONUM	054734	4787*	6354*	8122	8147	8202	11023	11030	11036*	11043*							

CZR6ND0 RK611 06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 277
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0275

SLPADR	001106	2270*	4725*	6313*	11598	11656*	11670*	11675	11677								
SLPERR	001110	2271*	4726*	5762*	5937*	6314*	11525	11615	11656	11671*	11677						
SLSTAD	060176	7029	7042	11803*	11810*												
SLSTBK	060200	7033	11804*	11811*													
SMADR1	001352	2376*															
SMADR2	001356	2380*															
SMADR3	001362	2383*															
SMADR4	001366	2386*															
SMAIL	001320	2250	2254	2349*	4743	5537	5726	5803	5891	5992	6075	6331	11067	11513			
		11669															
SMAMS1	001350	2370*															
SMAMS2	001354	2378*															
SMAMS3	001360	2381*															
SMAMS4	001364	2384*															
SMBADR	001002	2250*															
SMFLG	057670	11682*	11688	11723*	11727*												
SMNEW	057023	11576*															
SMGAD	001334	2356*	11698*	11701													
SMGLG	001336	2357*	11703*														
SMGTY	001320	2350*	11696	11704*	11716	11720*											
SMGWR	057012	11574*															
SM*YP1	001351	2371*															
SM*YP2	001355	2379*															
SM*YP3	001361	2382*															
SM*YP4	001365	2385*															
SMXCNT	057424	11667	11677*														
SMULL	001154	2291*	11094	11123													
SMWST=	000001	5508*	5510	5705*	5707	5785*	5787	5873*	5875	5961*	5963	6047*	6049				
SOCNT	056234	11371*	11400*	11413*													
SOCTVL	055540	8060	11228	11253*													
SOMODE	056236	11366*	11370*	11375	11378*	11389*	11415*										
SOVER	057410	11637	11657	11665	11674*												
SPASS	001326	2353*	4743*	5413*	5489	6257*	6258*	6266	6279	6285	6331*	7673					
SPASTM	001006	2252*															
SPWACT	057120	11589*	11590*	11599*													
SPWRDN	057034	4719	6307	11584*	11592												
SPWRUP	057046	11584	11589*														
SQUES	001314	2341*	4844	4879	4940	5027	5073	5132	5148	5256	5305	5331	5405	7012			
		7089	7775	7898	7944	11123	11535										
		8121	8146	8201	11019*												
SRAND	054634	8121	8146	8201	11019*												
SRDCHR	056660	11551*	11895														
SRDECI=	*****	11896															
SRDLIN=	*****	11896															
SRDOCT=	*****	11896															
SRDSZ =	000000	11572*															
SREGAD	001160	2295*															
SREGO	001162	2297*	9371	12322													
SREG1	001164	2298*	12322														
SREG10	001202	2305*	7158*	7203*	8394*	8395*	8396	8434*	8437	8617*	8618*	8619	8663*	8666			
		9596*	9621*	9656*	12326	12334											
SREG11	001204	2306*	8435*	8664*	9597*	9622*	9657*	12326	12334								
SREG12	001206	2307*	8436*	8665*	9598*	9623*	9650*	9652*	12326	12334							
SREG13	001210	2308*	8438*	8440*	8442*	8443*	8667*	8669*	8671*	8672*	12326	12334					
SREG14	001212	2309*	6204*	8437*	8439*	8441*	8666*	8668*	8670*	12329	12334						
SREG15	001214	2310*	6205*	8444*	8445*	8673*	8674*	12329	12334								
SREG16	001216	2311*	8373*	8634*	12330	12334											

U
U
U

CZR6ND0 RK611/06 SS VERIF 2 MACY11 30A(1052) 10-JAN-78 12:02 PAGE 278
CZR6ND.P11 10-JAN-78 10:05 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

[illegible]

B4EOPS	6241#	6250													
84SCOP	11620#	11634													
CKEXIT	2910#	8729	9116												
COMMEN	2162#														
ENDCOM	2162#														
ERROR	2056#	6038	6193	6207	6229	7163	7206	7226	7447	7470	8412	8413	8415	8433	8446
	8641	8642	8644	8662	8675	8745	8800	9184	9322	9326	9333	9337	9341	9345	9349
	9352	9355	9358	9359	9460	9465	9472	9474	9480	9482	9491	9493	9495	9499	9503
	9508	9513	9517	9521	9524	9528	9532	9536	9540	9544	9548	9553	9558	9563	9568
	9573	9578	9588	9601	9603	9613	9626	9628	9641	9658	9667	9671	9676	9681	9686
	9712	9714	9748												
ESCAPE	2162#														
GETPRI	2162#	11759													
GETSWR	2162#														
MSG	5508#	5510	5705#	5707	5785#	5787	5873#	5875	5961#	5963	6047#	6049			
MULT	2162#														
NEWTST	2162#	5503	5705	5785	5873	5961	6047								
NEXTAD	2909#														
NXTAD	2908#														
NXTST	2906#	5540	5689	5729	5806	5894	5995	6078							
PARMBK	1#	3491	3523												
PARMOD	1#	3397													
POP	2162#	11038	11469	11724	11725	11850									
PUSH	2162#	11019	11428	11685	11687	11708	11830								
P.DRVR	1#	3555													
REGDEF	1#	3179													
REPORT	2162#														
R.DRIV	1#	9909													
R.QDRV	1#														
SCOPE	2057#	5536	5725	5802	5890	5991	6074	6250							
SETPRI	2162#														
SETTRA	11880#	11889	11890	11891	11892	11895	11896	11897	11898						
SETUP	2162#	4705	6293												
SKIP	2162#	5497													
SLASH	2162#														
SPACE	2162#														
STARS	2010	2016	2162#	2224	2235	2237	2244	2260	2344	2347	4854	4862	5001	5015	5508
	5535	5705	5724	5785	5801	5873	5889	5961	5990	6047	6073	6243	6948	6950	6985
	6993	7002	7009	7017	7023	7068	7073	7103	7109	7120	7129	7149	7151	7168	7170
	7190	7192	7236	7241	7274	7277	7291	7301	7365	7373	7391	7396	7411	7426	7530
	7546	7598	7600	7616	7619	7640	7644	7657							

CZR6ND0 RK611 06 SS VERIF 2
CZR6ND.P11 10-JAN-78 10:05

MACY11 30A(1052) 10-JAN-78 12:02 PAGE 282
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0279

TYPNUM	2162#														
TYPOCS	2162#														
TYPOCT	2162#														
*YPTXT	2162#														
\$DECBN	1#														
\$OCTBN	1#	10952													
\$SCMRE	2258#	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	2308	2309	2310
	2311	2312	2313	2314	2315	2316	2317	2318	2319	2320	2321	2322	2323	2324	2325
	2326	2327	2328												
\$SCMTM	2258#	2329	2330	2331	2332	2333	2334	2335	2336	2337					
\$SESCA	2162#														
\$SNEW	2162#	5508	5705	5785	5873	5961	6047								
\$SSET	11880#	11889	11890	11891	11892	11895	11896	11897	11898						
\$SSETM	4743#	6331#													
\$SSKIP	2162#	5497													
.EQUAT	2009#	2052													
.HEADE	2009#	2019													
.KT11	2009#	2162													
.SETUP	2009#	4688													
.SWRHI	2009#	2029													
.SWRLO	2009#	2041#	2042	2043											
.SACT1	2009#	2222													
.SAPT8	2345#														
.SAPTH	2009#	2233													
.SAPTY	2009#	11678													
.SCATC	2009#	2203													
.SCMTA	2009#	2258													
.SDB20	2009#	11254													
.SDB20	2009#	11215													
.SEOP	2009#	6241													
.SERRO	2009#	11483													
.SPOWE	2009#														
.SRAND	2009#	11008													
.SREAD	2009#	11535													
.SSAVE	2009#	11812													
.SSCOP	2009#	11620													
.SSIZE	2009#	11735													
.SSUPR	2009#	11316													
.STRAP	2009#	11857													
.STYPD	2009#	11416													
.STYPE	2009#	11044													
.STYPO	2009#	11339													

. ABS. 070713 000

ERRORS DETECTED: 0

DSKZ:CZR6ND,DSKZ:CZR6ND,SEQ/SOL/CRF/NL:TOC/EQ:QNEWSW/DOC=DSKM:DRIV10.P11,DSKM:CZR6ND.P11

RUN-TIME: 46 40 3 SECONDS

RUN-TIME RATIO: 460/90=5.1

CORE USED: 49K (97 PAGES)

DOCUMENT PAGES: 279