

RK611/06

SUBSYSTEM VERIFICATION 1
CZR6MD0

AH-9138D-MC

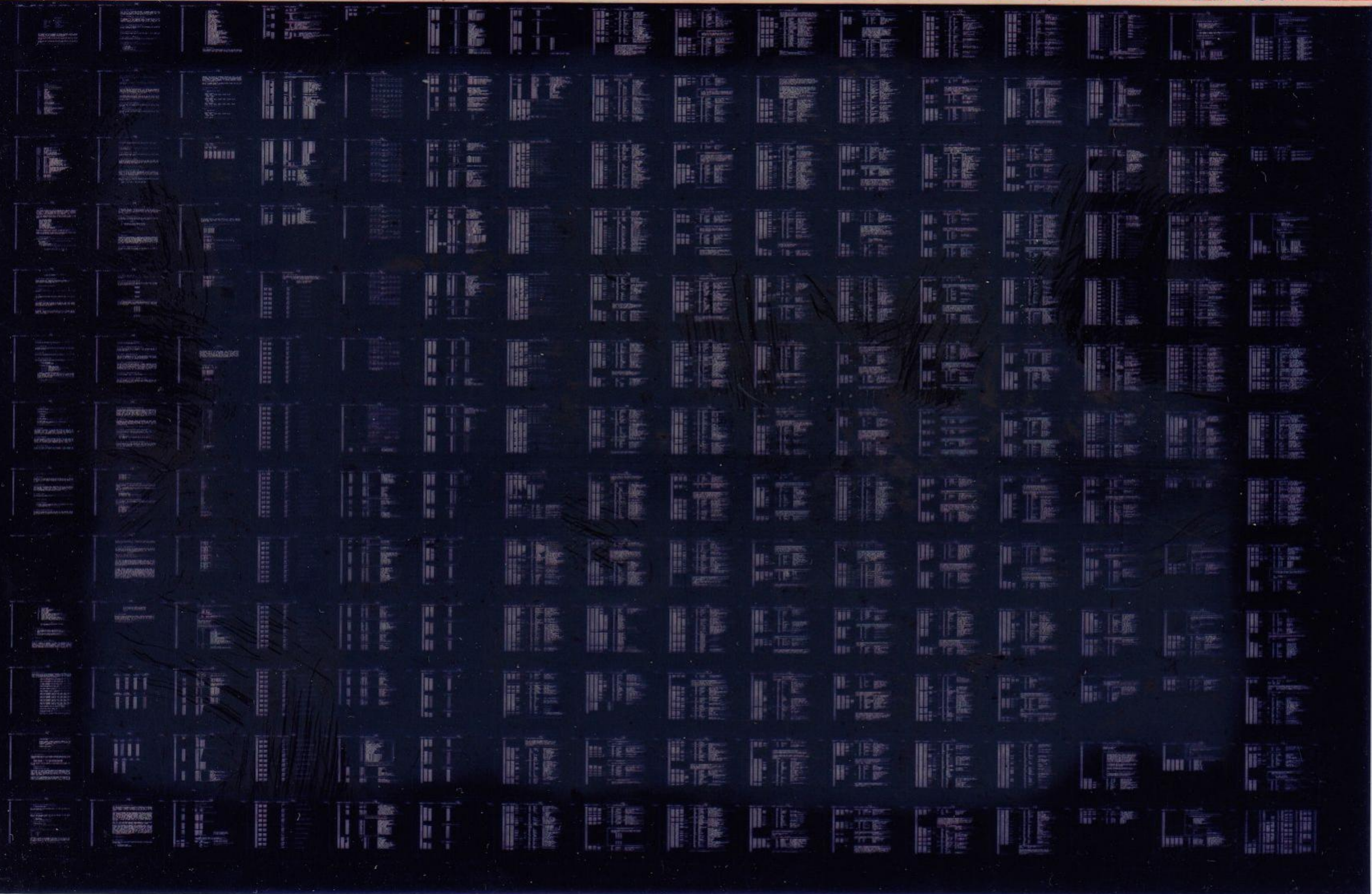
COPYRIGHT © 76-78

MAR 1978

digital

FICHE 1 OF 2

MADE IN USA



B01

EOF1CZR6KESEQ
PDP11 0040280
CZR6MD.P11

00010000 780223
CZR6MDO RK611/06 SS VERIF 1
10-JAN-78 11:48

PDP10 411 HDR1CZR6MDSEQ
MACY11 30A(1052) 10-JAN-78 12:16 PAGE 2

00010000 780223
SEQ 0001

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-9136D-MC
PRODUCT NAME: CZR6MDO RK611/06 SS VERIF 1
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
40
41
42
43
44
45
46

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	HARDWARE REQUIREMENTS
3.0	PRELIMINARY PROGRAM REQUIREMENTS
4.0	GENERAL PROGRAM CONSIDERATIONS
4.1	SYSMAC
4.2	XXDP
4.2.1	CHAIN MODE
4.2.2	DUMP MODE
4.3	ACT/APT
4.3.1	AUTOMATIC MODE
4.3.2	DUMP MODE
4.3.3	APT ETABLE DEFINITIONS
4.4	DUAL-ACCESS
4.5	MEMORY MANAGEMENT
4.6	MEMORY PARITY CHECK
4.7	BAD SECTORS
4.8	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
8.2.1	DRIVE SELECTION
8.2.2	TEST SELECTION
8.2.2.1	LIST TESTS (L)
8.2.2.2	CHANGE TESTS (C)
8.2.2.3	INPUT PARAMETERS AND RUN TESTS (I)
8.2.2.4	CONTROL Z (↑Z) FUNCTION
8.2.2.5	CONTROL C (↑C) FUNCTION

TABLE OF CONTENTS (CONT'D)

8.2.3	PARAMETER LIST ALTERATION
8.2.3.1	TYPE LIST, (T)
8.2.3.2	OPEN LIST, (O)
8.2.3.3	SET INDIVIDUAL PARAMETER, (S)
8.2.3.4	RUN TESTS, (R)
8.2.3.5	CONTROL Z (1Z) FUNCTION
8.2.3.6	CONTROL C (1C) FUNCTION
8.2.4	SPECIAL PARAMETER SPECIFICATIONS
8.2.4.1	PT - DATA PATTERN SELECT WORD
8.2.4.2	CS - CONTROL SWITCH WORD
9.0	DESCRIPTION OF TESTS
9.1	SEEK TESTS
9.1.1	TEST 1 - RECALIBRATE/SEEK TEST
9.1.2	TEST 2 - SEEK/SEEK TEST
9.1.3	TEST 3 - CYLINDER ADDRESSING TEST
9.1.4	TEST 4 - CYLINDER ADDRESS CROSSTALK TEST
9.1.5	TEST 5 - INCREMENT/DECREMENT SEEK TEST
9.1.6	TEST 6 - OSCILLATING SEEK TEST
9.1.7	TEST 7 - CONVERGING/DIVERGING SEEK TEST
9.1.8	TEST 10 - PSEUDO-RANDOM SEEK TEST
9.1.9	TEST 11 - MAXIMUM VELOCITY REVERSAL SEEK TEST
9.1.10	TEST 12 - MECHANICAL VIBRATION SEEK TEST
9.2	TIMING TESTS
9.2.1	TEST 13 - MAXIMUM ROTATIONAL LATENCY MEASUREMENT
9.2.2	TEST 14 - ONE CYLINDER SEEK TIME MEASUREMENT
9.2.3	TEST 15 - AVERAGE SEEK TIME MEASUREMENT
9.2.4	TEST 16 - MAXIMUM SEEK TIME MEASUREMENT
9.3	TEST 17 - SECTOR ADDRESSING TEST
9.4	TEST 20 - TRACK ADDRESSING TEST
9.5	TEST 21 - READ/WRITE DATA TEST
9.5.1	DATA PATTERNS
9.6	TEST 22 - DUAL-ACCESS DATA TEST
10.0	ERROR REPORTING
10.1	COMMON ERRORS
10.2	ERROR HANDLING
10.3	ERROR PRINTOUT EXAMPLES
APPENDIX A	SAMPLE ADDRESS 200 DEFAULT RUN
APPENDIX B	SAMPLE ADDRESS 204 RUN

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 1 OF THE RK06/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 1 EMPLOYS WORST-CASE SITUATIONS INVOLVING MECHANICAL POSITIONING, DISK ADDRESSING AND DATA TRANSFER. IN ADDITION, MEASUREMENTS ARE MADE PERTAINING TO DRIVE OPERATIONAL TIMING.

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
DISK ADDRESS LIMITS
DISK ADDRESS INCREMENTS
DATA PATTERNS USED
PHYSICAL MEMORY ADDRESS ON DATA TRANSFERS
WORD COUNT ON DATA TRANSFERS
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

THE FOLLOWING HARDWARE IS REQUIRED TO RUN PART 1 OF THE SUBSYSTEM VERIFICATION TESTS:

PDP-11/04, (05, 10 MFG. ONLY), 20, 34, 35, 40, 45, 50, 70 OR P30
16 K MEMORY
CONSOLE TELETYPE
KW11-L OR KW11-P CLOCK
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)

NOTE

IF NEITHER KW11-L OR P CLOCK IS PROVIDED, THE TIMING TESTS IN SECTION

9.2 WILL BE BYPASSED, AND A MESSAGE WILL
INFORM THE OPERATOR.

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE SUBSYSTEM VERIFICATION PROGRAM, THE RK611
CONTROLLER DIAGNOSTIC (CZR6A THRU DZR6E AND CZR6K) AND RK06
DRIVE DIAGNOSTIC (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 1-21 MAY BE
RUN IN CHAIN OR DUMP MODE, BUT DUAL-ACCESS TEST 22 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. A MESSAGE WILL INFORM THE OPERATOR
WHEN THIS IS NECESSARY.

4.3 ACT/APT

THE PROGRAM IS COMPATIBLE WITH ACT/APT CONVENTIONS. SUBSYSTEM TESTS 1-21 MAY BE RUN IN AUTOMATIC OR 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 1-21 OR DUAL-ACCESS DATA TEST 26 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 PROGRAMS :

1. SOFTWARE ENVIRONMENT

=1 IF APT SCRIPT MODE
=0 IF STANDALONE MODE

2. ENVIRONMENT MODE BYTE

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

3. SWITCH 1 (SOFTWARE SWITCH REGISTER)

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

4. SWITCH 2 (USER SWITCH REGISTER)
NOT USED

5. MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES

261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316

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.

12. REMAINING ETABLE ENTRIES ARE NOT USED.

4.4 DUAL-ACCESS

THIS PROGRAM IS DESIGNED TO UTILIZE DUAL-ACCESS IN TEST 22 ONLY
(DUAL-ACCESS DATA TEST - SEE SECTION 9.6). FOR THE PURPOSES OF ALL
OTHER TESTS (1-21). THE OPERATOR MUST GUARANTEE THAT THERE IS NO
INTERFERENCE FROM THE UNUSED PORT. IF FAILURES ARE ENCOUNTERED IN
TESTS 1-21 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 21 AND 22 ONLY. IN THESE TESTS, NP
TRANSFERS BETWEEN THE RK06 AND ANY SPECIFIED PHYSICAL MEMORY ABOVE THE
PROGRAM, ARE PERFORMED DURING READ/WRITE TESTING. (NOTE - TESTS 21
AND 22 ALSO SUPPORT 22-BIT ADDRESSING AND THE UNIBUS MAP, IF
INSTALLED). IN ALL OTHER TESTS, MEMORY MANAGEMENT IS DISABLED.

4.6 MEMORY PARITY CHECK

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, ALL
TESTING IS DONE WITH PARITY CHECK ENABLED. PARITY ERRORS ARE
REPORTED, AND TESTING IS ABORTED, UNLESS THE BYTE LABELED MEMABT WAS

PREVIOUSLY LOADED WITH A NON-ZERO VALUE.

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 LESSER DEGREE UPON THE PROCESSOR. HOWEVER, THE "AVERAGE" TIME REQUIRED TO RUN A QUICK-VERIFICATION (FIRST PASS) IS 14 MINUTES PER DRIVE. A SUBSEQUENT PASS WITH PARAMETERS DEFAULTED REQUIRES 21 MINUTES PER DRIVE. 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 1-21 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS, RK06/07 UNIBUS ADDR. AND INTERRUPT VECTOR FOR TESTS 1-21
220 DUAL-ACCESS DATA TEST 22 START ADDRESS

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:

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

429

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 10

J01

BIT OPTION

SEQ 0288
SEQ 0009

K01

```

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
07  DO EXPLICIT SEEKS IN TESTS 1-12
06  REPORT ONE ERROR PER TRANSFER IN TESTS 17,21
05  INHIBIT WRITES IN TEST 21
04  INHIBIT WRITE CHECKS IN TEST 21
03  INHIBIT READS AND SOFTWARE COMPARES IN TEST 21
02  INHIBIT SOFTWARE COMPARES IN TEST 21
01  READ AFTER A WRITE CHECK ERROR IN TEST 21
00  REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 17,21

```

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: "CZR6MD RK611/RK06-RK07 SUBSYSTEM VERIFICATION: PART 1,"
FOLLOWED BY: "LAST PHYS MEM ADR=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
IC	CYLINDER INCREMENT VALUE IN TESTS WHICH STEP THROUGH CYLINDER ADDRESSES, IC=1-632, DEFAULT=1
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
IT	TRACK INCREMENT VALUE IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, IT=1-2, DEFAULT=1
SO	FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMATTED 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 FORMATTED 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
IS	SECTOR INCREMENT VALUE IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, IS=1-25, DEFAULT=1
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=0-000070, DEFAULT=0
WC	WORD COUNT (NO. OF WORDS) USED IN DISK READ, WRITE, OR WRITE CHECK COMMANDS IN DATA TEST 20, WC=0-177777,

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
540
541

542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597

DEFAULT=MAXIMUM BUFFER AVAILABLE, WC=0 IS INTERPRETED
AS 65,536 WORDS

MA PHYSICAL MEMORY ADDRESS (STARTING ADDRESS OF DATA
BUFFER) USED ON ALL DATA TRANSFERS IN TESTS 21, 22. MA
MAY BE GREATER THAN OR EQUAL TO ADDRESS OF RWBUF (NOT
EXCEEDING AVAILABLE MEMORY ON THE SYSTEM).
DEFAULT = ADDRESS OF RWBUF.

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

SM MAXIMUM NUMBER OF UNIT STALL TIMES USED IN TEST 12 ONLY

8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)

8.2.1 DRIVE SELECTION

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

RK06-07 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 AL 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. THE LIST OF DEFAULT ITERATION NUMBERS BEGINS
AT ADDRESS "DFLTST", AND CAN BE ALTERED IN CORE.

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.

THE LIST OF DEFAULT PARAMETERS BEGINS AT ADDRESS "PROFLT", AND CAN BE ALTERED IN CORE.

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
03	INHIBIT TIMING REPORTS IN TESTS 13-16

NOTE

OTHER BITS UNUSED

9.0 DESCRIPTION OF TESTS

9.1 SEEK TESTS

THIS GROUP OF TESTS PERFORMS A VARIETY OF POSITIONING OPERATIONS. THROUGH THE EXECUTION OF READ HEADER COMMANDS, IMPLIED SEEKS ARE DONE TO SELECTED CYLINDERS, AND HEADER WORDS ARE READ AND CHECKED TO VERIFY THAT THE CORRECT CYLINDER WAS ADDRESSED. TESTING BEGINS WITH SIMPLE OPERATIONS WHICH VERIFY CYLINDER ADDRESSING CAPABILITY, AND PROCEEDS TO MORE INVOLVED SEEKING WHICH STRESSES THE SERVO MECHANISM. AT THE COMPLETION OF EACH SUBSYSTEM COMMAND, STATUS INDICATIONS AND ERROR BITS ARE CHECKED TO DETERMINE THE SUCCESS OF THE OPERATION. THROUGHOUT TESTING, SECTOR=FS, AND TRACK=FT.

9.1.1 TEST 1 - RECALIBRATE/SEEK TEST

THIS TEST WILL PERFORM A RECALIBRATE COMMAND (POSITION TO CYLINDER 0), FOLLOWED BY A SEEK TO CYLINDER LC. (AS IN ALL OF THE SEEK TESTS SEEKING IS IMPLIED, VIA THE READ HEADER COMMAND).

872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927

9.1.2 TEST 2 - SEEK/SEEK TEST

THIS TEST WILL PERFORM A SEEK TO CYLINDER FC, FOLLOWED BY A SEEK TO CYLINDER LC.

9.1.3 TEST 3 - CYLINDER ADDRESSING TEST

THIS TEST VERIFIES THE CYLINDER ADDRESS BITS, BY SEEKING TO CYLINDERS 0, 1, 2, 4, 10, 20, 40, 100, 200, AND 400 (OCTAL). THEN, THE SEEKS ARE DONE IN REVERSE BACK TO CYLINDER 0. IN BINARY, THE CYLINDER ADDRESSES SEQUENCE AS FOLLOWS:

```
000000000
000000001
000000010
:
100000000
010000000
001000000
:
000000001
000000000
```

9.1.4 TEST 4 - CYLINDER ADDRESS CROSSTALK TEST

THIS TEST PERFORMS SEEKS TO CYLINDERS WHOSE ADDRESSES ARE SUSCEPTIBLE TO BIT CROSSTALK WITHIN THE CYLINDER ADDRESSING LOGIC. FIRST A SEEK TO CYLINDER 0 IS DONE, FOLLOWED BY A SEEK TO 377 (OCTAL). THEN A SEEK TO 0 IS DONE AGAIN, AND A SEEK TO 376, ETC. THE CYLINDER ADDRESSES SEQUENCE AS FOLLOWS:

```
000 000 000
011 111 111
000 000 000
011 111 110
000 000 000
011 111 101
000 000 000
011 111 011
:
000 000 000
010 111 111
000 000 000
```

101 111 111

BY SEEKING TO 0 BETWEEN CROSSTALK SEEKS, THE CROSSTALK PATTERNS ALSO TEST THE CYLINDER DIFFERENCE LOGIC.

9.1.5 TEST 5 - INCREMENT/DECREMENT SEEK TEST

IN THIS TEST SEEKS ARE DONE IN INCREMENTS OF IC CYLINDERS STARTING AT CYL FC, AND ENDING AT OR BEYOND LC. THEN, THE SEEKS ARE DONE IN REVERSE, BACK TO FC. IF FC IS CHOSEN > LC, THE SEEKS WILL PROCEED IN THE OPPOSITE DIRECTION.

9.1.6 TEST 6 - OSCILLATING SEEK TEST

THIS TEST FIRST SEEKS TO CYLINDER FC, WHICH IS INITIALLY EQUAL TO NC. THEN NC IS INCREMENTED BY IC, AND A SEEK FROM FC TO NC IS MADE, FOLLOWED BY A SEEK BACK TO FC. NC IS INCREMENTED AGAIN, AND SEEKING FROM FC TO NC TO FC IS REPEATED UNTIL NC EXCEEDS LC. THEN, THE REVERSE IS DONE, DECREMENTING NC UNTIL NC EXCEEDS FC.

9.1.7 TEST 7 - CONVERGING/DIVERGING SEEK TEST

THIS TEST PERFORMS SEEKS WHICH EXERCISE CYLINDER DIFFERENCE VALUES. INITIALLY, NCYL1 = FC AND NCYL2 = LC. A SEEK IS DONE TO NCYL1 AND THEN TO NCYL2, THEN, NCYL1 IS INCREMENTED BY IC IN THE DIRECTION OF LC, AND NCYL2 IS INCREMENTED BY IC IN THE DIRECTION OF FC, AND SEEKS ARE DONE TO NCYL1 AND NCYL2 AGAIN. THIS SEEKING IS CONTINUED WITH INCREMENTED VALUES OF NCYL1 AND NCYL2, UNTIL NCYL1 EXCEEDS LC AND NCYL2 EXCEEDS FC. (NOTE: FC > LC IS PERMISSIBLE.) THIS TEST CAUSES SEEKING TO CONVERGING AND THEN DIVERGING CYLINDER VALUES.

9.1.8 TEST 10 - PSEUDO-RANDOM SEEK TEST

THIS TEST PERFORMS A SEEK TO A PSEUDO-RANDOMLY CHOSEN CYLINDER WHICH IS WITHIN THE RANGE (0-632), (0-1456 FOR RK07)

9.1.9 TEST 11 - MAXIMUM VELOCITY REVERSAL SEEK TEST

THIS TEST PERFORMS A SEEK FROM CYLINDER 0 TO CYLINDER 201(OCT), AND BACK TO CYLINDER 0. THIS PARTICULAR SEEK CAUSES THE HEADS TO ACCELERATE TO MAXIMUM VELOCITY, AND THEN IMMEDIATELY DECELERATE, WITH NO APPRECIABLE PLATEAU OF CONSTANT VELOCITY. THIS OPERATION INDUCES HEATING IN THE SERVO MECHANISM.

9.1.10 TEST 12 - MECHANICAL VIBRATION SEEK TEST

THIS TEST PERFORMS GEOMETRICALLY INCREASING SEEKS WITH GEOMETRICALLY DECREASING STALL TIME BETWEEN OPERATIONS WITH INTENT TO INDUCE VARYING VIBRATIONAL MODES UPON THE DRIVE. THE TEST BEGINS WITH LC = 1 AND ST = SM, THEN, THE FOLLOWING SEQUENCE IS PERFORMED: SEEKS ARE DONE BETWEEN 0 AND LC, 10(DEC) TIMES. THEN, ST IS DIVIDED BY 2 AND LC IS DOUBLED, AND SEEKS ARE DONE BETWEEN 0 AND LC AGAIN, 10(DEC) TIMES. THIS PROCESS IS CONTINUED FOR NEW VALUES OF ST AND LC UNTIL LC EXCEEDS CYL 400 (OCT). THEN, THE WHOLE PROCESS IS REVERSED, WITH ST BEING DOUBLED AND LC DIVIDED BY 2, UNTIL LC BECOMES < 1.

9.2 TIMING TESTS

THIS GROUP OF TESTS PERFORMS TIMING MEASUREMENTS OF BASIC DRIVE OPERATIONS - SPINDLE ROTATION AND SEEKING. FOR EACH TEST THE FUNCTION BEING MEASURED IS TIMED FOR A GIVEN NUMBER OF OCCURENCES, AND THE MINIMUM, MAXIMUM, AND AVERAGE VALUES ARE TYPED ALONG WITH THE NUMBER OF OPERATIONS WHICH EXCEEDED THE LIMITS SPECIFIED IN THE RK06/07 DISK DRIVE SPECIFICATION.

NOTE- FOR 50 HZ OPERATION, PATCH 1 INTO LOCATION 166 (LABELED HZ:).

9.2.1 TEST 13 - MAXIMUM ROTATIONAL LATENCY MEASUREMENT

THIS TEST MEASURES THE INTERVAL OF TIME BETWEEN 2 INDEX MARKS IN A SINGLE DISK ROTATION. THE SPECIFIED MAXIMUM ROTATIONAL LATENCY = 25 MILLI-SEC + OR - 2.5%. THIS MEASUREMENT IS MADE 128 TIMES.

SAMPLE PRINTOUT --

ROTATIONAL TIMES :
MIN = 24086 US 103 OF 128 BELOW SPEC'D MIN OF 24375 US
MAX = 25065 US 0 OF 128 ABOVE SPEC'D MAX OF 25625 US
AVG = 24285 US

9.2.2 TEST 14 - ONE CYLINDER SEEK TIME MEASUREMENT

THIS TEST MEASURES THE TIME REQUIRED TO SEEK BETWEEN 2 ADJACENT CYLINDERS, BOTH IN THE FORWARD AND REVERSE DIRECTIONS. SEEKS ARE DONE TO 0, 1, 2, ..., 631, 632 AND THEN TO 631, 630, ..., 2, 1, 0 AND THE RESULTS ARE TYPED FOR EACH DIRECTION. THE SPECIFIED ONE CYL SEEK TIME IS < 8 MILLI-SEC. NOTE: SEEKS ARE TO CYL 1456 FOR THE RK07.

SAMPLE PRINTOUT --

ONE CYL SEEK TIMES :

FORWARD DIRECTION

MIN = 6332 US

MAX = 6749 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US

AVG = 6536 US

REVERSE DIRECTION

MIN = 6314 US

MAX = 6749 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US

AVG = 6532 US

9.2.3 TEST 15 - AVERAGE SEEK TIME MEASUREMENT

THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD AND REVERSE DIRECTIONS. THE AVERAGE TIME IS CALCULATED FROM THE FOLLOWING FORMULA :

$$T_{AVG} = [T_1(410)(2) + T_2(409)(2) + \dots + T_{410}(1)(2)] / (410)(410)$$

WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS. FORWARD AND REVERSE TIMES ARE MEASURED AND TYPED, SEPARATELY. THE AVERAGE SEEK TIME IS SPECIFIED TO BE < 38 MILLI-SEC.

SAMPLE PRINTOUT --

AVERAGE SEEK TIMES :

FORWARD DIRECTION

AVG = 35673 US SPEC'D MAX IS 38000 US

REVERSE DIRECTION

AVG = 35823 US SPEC'D MAX IS 38000 US

9.2.4 TEST 16 - MAXIMUM SEEK TIME MEASUREMENT

THIS TEST MEASURES THE TIME REQUIRED TO SEEK FROM CYL 0 TO CYL 632 OCT. (410 DEC.). THIS TIME REPRESENTS THE MAXIMUM SEEK TIME, AND IT IS MEASURED 128 TIMES IN EACH DIRECTION. THE SPECIFIED MAXIMUM SEEK TIME IS < 75 MILLI-SEC.

SAMPLE PRINTOUT --

MAXIMUM SEEK TIMES :

FORWARD DIRECTION

MIN = 68950 US

MAX = 69286 US 0 OF 128 ABOVE SPEC'D MAX OF 75000 US

AVG = 69122 US

REVERSE DIRECTION

MIN = 70194 US

MAX = 70667 US 0 OF 128 ABOVE SPEC'D MAX OF 75000 US

AVG = 70407 US

9.3 TEST 17 - SECTOR ADDRESSING TEST

IN THIS TEST, ALL SECTORS OF FC, FT ARE WRITTEN, EACH WITH 256(DEC) WORDS OF ITS OWN SECTOR NO. + 100(OCT). NEXT, FOR EACH SECTOR WRITTEN A WRITE CHECK IS DONE, FOLLOWED BY A READ AND SOFTWARE COMPARE OF THE DATA.

9.4 TEST 20 - TRACK ADDRESSING TEST

IN THIS TEST, SECTOR FS OF CYL FC IS WRITTEN WITH 256 (DEC) WORDS OF THE TRACK NO. + 100(OCT), FOR EACH OF TRACKS 0,1,2. THEN, A WRITE CHECK OF EACH SECTOR IS DONE TO VERIFY THE WRITES. THEN, EACH OF THE 3 SECTORS IS RE-WRITTEN, AND AFTER EACH WRITE ALL OF THE THREE SECTORS ARE WRITE-CHECKED TO DETECT TRACK ADDRESSING PROBLEMS.

9.5 TEST 21 - READ/WRITE DATA TEST

THE READ/WRITE DATA TEST HAS 2 DIFFERENT VERSIONS, DEPENDING ON THE VALUE OF THE DATA PATTERN PARAMETER (PT).

FOR PT = 0 :

THIS TEST IS THE QUICK VERIFY DEFAULT DATA TEST, WHICH IS RUN WHEN PT=0, DUE TO PARAMETER INPUT CHOICE, OR ADDRESS 200 START. IN THIS TEST, THE ENTIRE PACK IS TESTED WITH A SINGLE DATA PATTERN. THIS PATTERN IS COMPRISED OF THE DATA PATTERNS 00-15, WHICH REPEATS EVERY 256(DEC) WORDS. EACH TRACK IS WRITTEN IN 4 SEGMENTS, FOLLOWED BY A WRITE-CHECK, READ, AND SOFTWARE COMPARE. THE SEGMENTS ARE 6 SECTORS EACH, WHICH MEANS THAT TRACK SPIRALING OCCURS ON THE LAST SEGMENT WRITTEN ON EACH TRACK.

FOR PT NOT = 0 :

THIS TEST PERFORMS READ/WRITE FUNCTIONS ON THE ENTIRE RANGE OF CYLINDERS (FC-LC), TRACKS (FT-LT), AND SECTORS (FS-LS) SPECIFIED. AT EACH SPECIFIED SECTOR WC WORDS OF THE CURRENT REPEATING DATA PATTERN ARE WRITTEN, AND THEN WRITE-CHECKED, FOLLOWED BY A READ AND SOFTWARE COMPARE. THIS IS DONE FOR ALL THE SPECIFIED SECTORS ON ALL THE TRACKS USING THE SECTOR INCREMENT IS, AND TRACK INCREMENT IT. AND THEN IT IS REPEATED USING EACH OF THE OTHER DATA PATTERNS CHOSEN IN PARAMETER PT. THEN, EACH OF THE ABOVE OPERATIONS ARE REPEATED AT EACH OF THE REMAINING CYLINDERS, IN THE SPECIFIED RANGE, USING THE CYLINDER INCREMENT IC. NOTE THAT FS MUST BE CHOSEN < OR = LS, AND FT MUST BE < OR = LT. HOWEVER, FC MAY BE < =, OR > LC, AND IF FC > LC, THE CYLINDER ADDRESS WILL BE DECREMENTED BY IC (INSTEAD OF INCREMENTED) TO OBTAIN EACH NEW CYLINDER ADDRESS.

NOTE:

THE PACK ADDRESS LIMITS

1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151

(FT,LT,FS,LS,FC,LC) REFER TO THE RANGE
OF DATA SECTORS AT WHICH TRANSFERS MAY
BEGIN, USING THE SPECIFIED WORD COUNT
WC. IF WC IS LARGE ENOUGH, HOWEVER, THE
TRANSFERS MAY EXTEND BEYOND THE ABOVE
LIMITS.

9.5.1 DATA PATTERNS

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:

1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174

L02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 25

SEQ 0303
SEQ 0024

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

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

M02

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 26

SEQ 0304
SEQ 0025

	8 SHIFTED 1 IN FIELD OF 0'S	9 SHIFTED 0 IN FIELD OF 1'S	10 ALTERNATING 0-1	11 ALTERNATING 1-0
1224				
1225				
1226				
1227				
1228				
1229				
1230				
1231				
1232				
1233	000001	177776	052525	125252
1234	000002	177775	052525	125252
1235	000004	177773	052525	125252
1236	000010	177767	052525	125252
1237	000020	177757	052525	125252
1238	000040	177737	052525	125252
1239	000100	177677	052525	125252
1240	000200	177577	052525	125252
1241	000400	177377	052525	125252
1242	001000	176777	052525	125252
1243	002000	175777	052525	125252
1244	004000	173777	052525	125252
1245	010000	167777	052525	125252
1246	020000	157777	052525	125252
1247	040000	137777	052525	125252
1248	100000	077777	052525	125252
1249				
1250				
1251				
1252				
1253				
1254				
1255				
1256				
1257				
1258				
1259				
1260				
1261				
1262				
1263				
1264				
1265				
1266				
1267				
1268				
1269				
1270				

	12 SHIFTING 0'S AND 1'S	13 COMPOSITE ROTATING	14 PSEUDO- RANDOM	15 USER- DEFINED
1251				
1252				
1253				
1254				
1255	000001	072307		
1256	000003	135143		
1257	000007	156461		
1258	000017	167230		
1259	000037	073514		
1260	CJ0077	035646		
1261	000177	016723		
1262	000377	107351		
1263	000777	143564		
1264	001777	061672		
1265	003777	030735		
1266	007777	114356		
1267	017777	046167		
1268	037777	123073		
1269	077777	151453		
1270	177777	164616		

9.6 TEST 22 - DUAL-ACCESS DATA TEST

THIS TEST IS DESIGNED TO RUN ON 2 DIFFERENT PROCESSORS, SIMULTANEOUSLY AND INDEPENDENTLY PERFORMING DYNAMIC DATA OPERATIONS ON THE SAME DRIVE, THROUGH 2 DIFFERENT CONTROLLERS. TEST 22 HAS A SEPARATE STARTING ADDRESS = 220 AND IT IS NEVER RUN WITH THE OTHER TESTS IN AN AUTOMATIC MANNER. ALSO, BOTH PORTS MUST BE SWITCHED ON-LINE, AT THE DRIVE.

THIS TEST IS A MODIFIED VERSION OF READ/WRITE DATA TEST 21, IN WHICH A DRIVE RELEASE COMMAND IS DONE IMMEDIATELY AFTER COMPLETION OF READ WRITE OPERATIONS AT A PARTICULAR PACK ADDRESS. IN ALL OTHER RESPECTS, TESTS 21 AND 22 ARE IDENTICAL, AS FAR AS EACH PROCESSOR IS CONCERNED. THIS MEANS THAT THE TEST PARAMETERS (ADDRESS LIMITS, DATA PATTERNS, ETC.) CAN BE DEFINED DIFFERENTLY FOR EACH PROCESSOR. IF DIFFERENT DATA PATTERNS ARE USED, FOR EXAMPLE, THE DATA WHICH APPEARS ON THE DISK AT TIME OF FAILURE CAN IDENTIFY THE CONTROLLER FROM WHICH IT ORIGINATED. LIKEWISE, THE WORD COUNT MAY BE CHOSEN APPROPRIATELY SMALL (WC=1) TO CAUSE THE GREATEST AMOUNT OF CONTENTION FOR THE USE OF THE DRIVE.

NOTE THAT IF BOTH PROCESSORS ARE PERFORMING TRANSFERS WITH DIFFERENT DATA AT THE SAME PACK ADDRESS, INTERFERENCE MAY OCCUR, IF ONE CONTROLLER SHOULD LOSE THE DRIVE PREMATURELY TO REPORT AN ERROR.

WHEN THE TEST IS STARTED (AT ADDRESS 220) PARAMETER INPUT MODE IS ENTERED, AND OPERATIONAL PARAMETERS AND THE DRIVE NUMBER MAY BE TYPED IN (SEE SECTION 8.1) AS WELL AS RK06 UNIBUS ADDRESS AND INTERRUPT VECTOR. THE TEST NUMBER IS AUTOMATICALLY SET TO 22, AND THE PROGRAM TYPES "ENTER T.O.S. OR R" AS DESCRIBED IN SECTION 8.2.3. AT THIS POINT, THE OPERATION OF THE PROGRAM IS THE SAME AS PREVIOUSLY DESCRIBED IN CONJUNCTION WITH THE OTHER TESTS. HOWEVER, ONLY TEST 22 MAY BE RUN, AND ONLY ON THE SINGLE SELECTED DRIVE. ALSO, THE OPERATOR MUST INDEPENDENTLY LOAD AND START THE TEST IN EACH PROCESSOR AND SELECT INPUT PARAMETERS, INDEPENDENTLY. THE DRIVE NUMBER TYPED AT EACH PROCESSOR CONSOLE MUST BE THE SAME, IN ORDER TO PERFORM THE TESTS ON THE SAME DRIVE, SIMULTANEOUSLY.

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

1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326

1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382

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 ON A GIVEN ERROR. IN THE CASE OF DATA MISCOMPARE ERRORS, FOR INSTANCE, THE WORD 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.

MANY OF THE COMMON ERRORS ARE RETRIED SEVERAL TIMES, AND IF THEY PERSIST, AN ABORT IS MADE. DATA ERRORS AND MISCOMPARES ARE REPORTED, AND THE PROGRAM CONTINUES TESTING.

10.3 ERROR PRINTOUT EXAMPLES

EXAMPLE 1:

** NON-EXISTANT DRIVE ERROR TEST 2

PREVIOUS COMMAND :

DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
000000	000177	000000	000000	000000	000000
HI BA	LO BA				
000000	000000				

CURRENT COMMAND :

ERR PC	DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
037576	000000	000125	000000	000000	000000	000000
HI BA	LO BA					
000000	000000					

RK611 REGISTERS:

RKCS1	RKCS2	RKDC	RKDA	RKWCR	RKBA	RKASOF
140324	010100	000000	000000	000000	000000	000400

REMAINING REGISTERS NOT VALID

EXAMPLE 2:

** DATA MISCOMPARE TEST 21

PREVIOUS COMMAND :

DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
000000	000121	000000	000000	000000	036000
HI BA	LO BA				
000000	074000				

CURRENT COMMAND :

ERR PC	DRIVE	CMND	CYLNR	TRACK	SECTOR	WD CNT
--------	-------	------	-------	-------	--------	--------

D03

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 30

SEQ 0308
SEQ 0029

1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461

034606 000000 000121 000000 000000 000000 036000
HI BA LO BA
000000 074000

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

WD #	GOOD	BAD	HI PHY	LO PHY	VRT AD	KIPAR6
000000	000001	125252	000000	074000	154000	000600
000001	000002	125252	000000	074002	154002	000600
000002	000003	125252	000000	074004	154004	000600
000003	000004	125252	000000	074006	154006	000600
000004	000005	125252	000000	074010	154010	000600
000005	000006	125252	000000	074012	154012	000600
000006	000007	125252	000000	074014	154014	000600
000007	000010	125252	000000	074016	154016	000600
000010	000011	125252	000000	074020	154020	000600
000011	000012	125252	000000	074022	154022	000600

E03

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 31

SEQ 0309
SEQ 0030

APPENDIX A

SAMPLE ADDRESS 200 DEFAULT RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM
VERIFICATION, PART 1, 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.

CZR6M-D - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 1

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

ROTATIONAL TIMES :
MIN = 24153 US 99 OF 128 BELOW SPEC'S MIN OF 24375 US
MAX = 25125 US 0 OF 128 ABOVE SPEC'D MAX OF 25625 US
AVG = 24374 US

ONE CYL SEEK TIMES :
FORWARD DIRECTION
MIN = 6323 US
MAX = 6686 US 0 OF 410 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US
AVG = 6518 US
REVERSE DIRECTION
MIN = 6332 US
MAX = 6713 US 0 OF 410 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US
AVG = 6513 US

AVERAGE SEEK TIMES :

1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517

F03

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:15 PAGE 32

SEQ 0310
SEQ 0031

1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536

FORWARD DIRECTION

AVG = 35673 US SPEC'D MAX IS 38000 US

REVERSE DIRECTION

AVG = 35823 US SPEC'D MAX IS 38000 US

MAXIMUM SEEK TIMES :

FORWARD DIRECTION

MIN = 68950 US

MAX = 69150 US 0 OF 128 ABOVE SPEC'D MAX OF 75000 US

AVG = 69064 US

REVERSE DIRECTION

MIN = 70222 US

MAX = 70530 US 0 OF 128 ABOVE SPEC'D MAX OF 75000 US

AVG = 70379 US

END PASS # 1

APPENDIX B

SAMPLE ADDRESS 204 RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART 1, 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 2 BE RUN 25(OCTAL) TIMES, TEST 7 FOUR TIMES, TEST 14 RUN 2 TIMES, AND TEST 21 ONCE. THESE TESTS WERE SPECIFIED TO RUN FOR 2 PROGRAM PASSES, USING PARAMETER VALUES TYPED BY THE OPERATOR. IN THIS CASE PACK ADDRESS PARAMETERS AND DATA TRANSFER WORD COUNT WERE SPECIFIED, FOR TESTS WHICH UTILIZE THEM.

CZR6M-D - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 1

LAST PHYS MEM ADR = 377776

PARAMETER INPUT MODE

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

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

*

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>

H03

CZR6MD0 RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 34

SEQ 0312
SEQ 0033

```

*
TEST  ITERATIONS
1      10 * 0
2      200 * 25
3      10 * 0
4      10 * 0
5      2 * 0
6      2 * 0
7      2 * 0
10     400 * 0
11     500 * 0
12     2 * 0
13     1 * 0
14     1 * 2
15     1 * 0
16     1 * 0
17     10 * 0
20     10 * 0
21     1 * 0

```

ENTER L, C, OR I
* L

```

TEST  ITERATIONS
1      0
2      25
3      0
4      0
5      0
6      0
7      0
10     0
11     0
12     0
13     0
14     2
15     0
16     0
17     0
20     0
21     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
* O

1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648

TO DEFAULT ALL PARAMETERS TYPE D<CR>, ELSE <CR>

*
FC=0 *
LC=632 * 2
IC=1 *
FT=0 * 1
LT=2 *)
IT+1 *
SO=0 *
S1=23 *
S2=0 * 10
S3=25 * 10
IS=1 *
PT=0 * 20
MA=61566 *
WC=403 * 1100
CS=0 *
ST=0 *
SM=1000 *

ENTER T, O, S, OR R

* T
FC=0
LC=2
IC=1
FT=1
LT=1
IT=1
SO=0
S1=23
S2=10
S3=10
IS=1
PT=20
MA=61566
WC=1100
CS=0
ST=0
SM=1000

ENTER T, O, S, OR R

* R
ENTER NO. OF PASSES (1-77777) :
* 2TESTING DRIVE 0
DRIVE SER. NO. 8
CART. SER. NO. 334ONE CYL SEEK TIMES :
FORWARD DIRECTIONMIN = 6319 US
MAX = 6691 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US

J03

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 36

SEQ 0314
SEQ 0035

AVG = 6502 US
REVERSE DIRECTION
MIN = 6301 US
MAX = 6682 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6498 US

ONE CYL SEEK TIMES :
FORWARD DIRECTION
MIN = 6332 US
MAX = 6695 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6506 US
REVERSE DIRECTION
MIN = 6314 US
MAX = 6686 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6503 US

ONE CYL SEEK TIMES :
FORWARD DIRECTION
MIN = 6319 US
MAX = 6691 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6504 US
REVERSE DIRECTION
MIN = 6301 US
MAX = 6700 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6500 US

ONE CYL SEEK TIMES :
FORWARD DIRECTION
MIN = 6326 US
MAX = 6698 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6511 US
REVERSE DIRECTION
MIN = 6317 US
MAX = 6698 US 0 OF 410 ABOVE SPEC'D MAX OF 8000 US
AVG = 6507 US

END PASS # 2
TO TEST ALL DRIVES TYPE "A"<CR>, ELSE <CR>
*

2

K03

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 37

SEQ 0315
SEQ 0036

1751 000000

PART=0

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

;*** REV 004 ***

.NLIST MC,MD,CND

.LIST ME

.ENABL ABS,AMA

SSWR= 167000

1760 167000

;*****

.SBTTL STARTING ADDRESSES

;*

;* 200 DEFAULT PARAMETERS FOR TESTS 1-21

;* 204 SELECT PARAMETERS FOR TESTS 1-21

;* 220 DUAL-ACCESS DATA TEST

;*****

1769 000001

STN=1

.TITLE CZR6MDO RK611/06 SS VERIF 1

;*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

;* 7

DO EXPLICIT SEEKS IN TESTS 1-12

;* 6

REPORT 1 ERROR PER TRANSFER IN TESTS 17,21

;* 5

INHIBIT WRITES IN TEST 21

;* 4

INHIBIT WRITE CHECKS IN TEST 21

;* 3

INHIBIT READS AND SOFTWARE COMPARES IN TEST 21

;* 2

INHIBIT SOFTWARE COMPARES IN TEST 21

;* 1

READ AFTER A WRITE CHECK ERROR IN TEST 21

;* 0

REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 17,21

;*

.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

;* 03

INHIBIT TIMING REPORTS IN TESTS 13-16

1805

1806

L03

.SBTTL BASIC DEFINITIONS

.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

STACK= 1100

.EQUIV EMT,ERROR ;;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE ;;;BASIC DEFINITION OF SCOPE CALL

.*MISCELLANEOUS DEFINITIONS

HT= 11 ;;;CODE FOR HORIZONTAL TAB

LF= 12 ;;;CODE FOR LINE FEED

CR= 15 ;;;CODE FOR CARRIAGE RETURN

CRLF= 200 ;;;CODE FOR CARRIAGE RETURN-LINE FEED

PS= 177776 ;;;PROCESSOR STATUS WORD

.EQUIV PS,PSW

STKLMT= 177774 ;;;STACK LIMIT REGISTER

PIRQ= 177772 ;;;PROGRAM INTERRUPT REQUEST REGISTER

DSWR= 177570 ;;;HARDWARE SWITCH REGISTER

DDISP= 177570 ;;;HARDWARE DISPLAY REGISTER

.*GENERAL PURPOSE REGISTER DEFINITIONS

R0= %0 ;;;GENERAL REGISTER

R1= %1 ;;;GENERAL REGISTER

R2= %2 ;;;GENERAL REGISTER

R3= %3 ;;;GENERAL REGISTER

R4= %4 ;;;GENERAL REGISTER

R5= %5 ;;;GENERAL REGISTER

R6= %6 ;;;GENERAL REGISTER

R7= %7 ;;;GENERAL REGISTER

SP= %6 ;;;STACK POINTER

PC= %7 ;;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS

PR0= 0 ;;;PRIORITY LEVEL 0

PR1= 40 ;;;PRIORITY LEVEL 1

PR2= 100 ;;;PRIORITY LEVEL 2

PR3= 140 ;;;PRIORITY LEVEL 3

PR4= 200 ;;;PRIORITY LEVEL 4

PR5= 240 ;;;PRIORITY LEVEL 5

PR6= 300 ;;;PRIORITY LEVEL 6

PR7= 340 ;;;PRIORITY LEVEL 7

.*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000

SW14= 40000

SW13= 20000

SW12= 10000

SW11= 4000

SW10= 2000

SW09= 1000

SW08= 400

SW07= 200

SW06= 100

SW05= 40

SW04= 20

SW03= 10

BASIC DEFINITIONS

1863 000004
1864 000002
1865 000001

SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

1877 100000
1878 040000
1879 020000
1880 010000
1881 004000
1882 002000
1883 001000
1884 000400
1885 000200
1886 000100
1887 000040
1888 000020
1889 000010
1890 000004
1891 000002
1892 000001

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

1905 000004
1906 000010
1907 000014
1908 000014
1909 000014
1910 000014
1911 000020
1912 000024
1913 000030
1914 000034
1915 000060
1916 000064
1917 000240
1918

.*BASIC "CPU" TRAP VECTOR ADDRESSES

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

```
1919
1920      ;*KT11 VECTOR ADDRESS
1921
1922      000250      MMVEC= 250
1923
1924      ;*KT11 STATUS REGISTER ADDRESSES
1925
1926      177572      SR0= 177572
1927      177574      SR1= 177574
1928      177576      SR2= 177576
1929      172516      SR3= 172516
1930
1931      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1932
1933      172300      KIPDR0= 172300
1934      172302      KIPDR1= 172302
1935      172304      KIPDR2= 172304
1936      172306      KIPDR3= 172306
1937      172310      KIPDR4= 172310
1938      172312      KIPDR5= 172312
1939      172314      KIPDR6= 172314
1940      172316      KIPDR7= 172316
1941
1942      ;*KERNEL "I" PAGE ADDRESS REGISTERS
1943
1944      172340      KIPAR0= 172340
1945      172342      KIPAR1= 172342
1946      172344      KIPAR2= 172344
1947      172346      KIPAR3= 172346
1948      172350      KIPAR4= 172350
1949      172352      KIPAR5= 172352
1950      172354      KIPAR6= 172354
1951      172356      KIPAR7= 172356
1952
1953      170200      MAPL00=170200
1954      170202      MAPH00=170202
1955      172100      MEMCSR=172100      ; MEMORY CSR REG START ADRS
1956      177740      LOERAD=177740      ; 11/70 MEM LO ERROR ADRS REG
1957      177742      HIERAD=177742      ; 11/70 MEM HI ERROR ADRS REG
1958      177744      MEMSYS=177744      ; 11/70 MEM SYSTEM REG
1959      .SBTTL TRAP CATCHER
1960
1961      000000      .=0
1962      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1963      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1964      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1965      000174      000174      .=174
1966      000174      000000      DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
1967      000176      000000      SWREG: .WORD 0      ;; SOFTWARE SWITCH REGISTER
1968
1969      000200      000137      012522      .SBTTL STARTING ADDRESS(ES)
1970      000166      000166      JMP 0#DFSTRT      ;; JUMP TO STARTING ADDRESS OF PROGRAM
1971      000166      000000      .=166
1972      000204      000204      HZ: .WORD 0      ; LINE FREQ. FLAG - 0 = 60 HZ
1973      000204      000137      012574      .=204
1974      000220      000220      JMP 0#PSTART
1975      .=220
```

1975 000220 000137 012556
 1976 000224 000700
 1977 000226 001100
 1978
 1979
 1980
 1981
 1982 000230
 1983 000046 000046
 1984 000046 025534
 1985 000052 000052
 1986 000052 140000
 1987 000230
 1988 001000
 1989
 1990
 1991
 1992
 1993
 1994 001000
 1995 000024 000024
 1996 000024 000200
 1997 000044 000044
 1998 000044 001000
 1999 001000
 2000
 2001
 2002
 2003
 2004 001000
 2005 001000 000000
 2006 001002 001320
 2007 001004 001320
 2008 001006 001546
 2009 001010 001546
 2010 001012 000030
 2011 120210
 2012 177440
 2013 000377

JMP @DASTRT
 ..LOW: .WORD 700
 ..HIGH: .WORD 1100
 .SBTTL ACT11 400KS
 ;*****
 ;HOOKS REQUIRED BY ACT11
 \$SVPC=.; SAVE PC
 =46
 \$ENDAD ;;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
 =52
 .WORD 140000 ;;;2)SET LOC.52 TO 140000
 .= \$SVPC ;;; RESTORE PC
 =1000
 .SBTTL APT PARAMETER BLOCK
 ;*****
 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
 ;*****
 .SX=.; SAVE CURRENT LOCATION
 =24 ;;;SET POWER FAIL TO POINT TO START OF PROGRAM
 200 ;;;FOR APT START UP
 =44 ;;;POINT TO APT INDIRECT ADDRESS PNTR.
 \$AP1HDR ;;;POINT TO APT HEADER BLOCK
 .= \$X ;;;RESET LOCATION COUNTER
 ;*****
 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
 ;INTERFACE SPEC.
 \$APTHD:
 \$HIBTS: .WORD 0 ;;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 \$MBADR: .WORD \$MAIL ;;;ADDRESS OF APT MAILBOX (BITS 0-15)
 \$TSTM: .WORD 1320 ;;;RUN TIM OF LONGEST TEST
 \$PASTM: .WORD 1546 ;;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
 \$UNITM: .WORD 1546 ;;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 .WORD \$ETEND-\$MAIL/2 ;;;LENGTH MAILBOX-ETABLE(WORDS)
 AVECT1=120210 ;;;RKVEC=210, RKPRI=5
 ABASE=177440 ;;;RKBAS ADRS
 ADEVM=000377 ;;;SET DEVICES 0-7 IN MAP

```

2014      .SBTTL  COMMON TAGS
2015
2016      ;*****
2017      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
2018      ;*USED IN THE PROGRAM.
2019
2020      001100      .=1100
2021      001100      SCMTAG:      .WORD      0      ;;START OF COMMON TAGS
2022      001100      000000      ;CONTAINS THE TEST NUMBER
2023      001102      000      ;CONTAINS ERROR FLAG
2024      001103      000      ;CONTAINS SUBTEST ITERATION COUNT
2025      001104      000000      ;CONTAINS SCOPE LOOP ADDRESS
2026      001106      000000      ;CONTAINS SCOPE RETURN FOR ERRORS
2027      001110      000000      ;CONTAINS TOTAL ERRORS DETECTED
2028      001112      000000      ;CONTAINS ITEM CONTROL BYTE
2029      001114      000      ;CONTAINS MAX. ERRORS PER TEST
2030      001115      001      ;CONTAINS PC OF LAST ERROR INSTRUCTION
2031      001116      000000      ;CONTAINS ADDRESS OF 'GOOD' DATA
2032      001120      000000      ;CONTAINS ADDRESS OF 'BAD' DATA
2033      001122      000000      ;CONTAINS 'GOOD' DATA
2034      001124      000000      ;CONTAINS 'BAD' DATA
2035      001126      000000      ;RESERVED--NOT TO BE USED
2036      001130      000000      ;
2037      001132      000000      ;
2038      001134      000      ;AUTOMATIC MODE INDICATOR
2039      001135      000      ;INTERRUPT MODE INDICATOR
2040      001136      000000      ;
2041      001140      177570      ;ADDRESS OF SWITCH REGISTER
2042      001142      177570      ;ADDRESS OF DISPLAY REGISTER
2043      001144      177560      ;TTY KBD STATUS
2044      001146      177562      ;TTY KBD BUFFER
2045      001150      177564      ;TTY PRINTER STATUS REG. ADDRESS
2046      001152      177566      ;TTY PRINTER BUFFER REG. ADDRESS
2047      001154      000      ;CONTAINS NULL CHARACTER FOR FILLS
2048      001155      002      ;CONTAINS # OF FILLER CHARACTERS REQUIRED
2049      001156      012      ;INSERT FILL CHARS. AFTER A "LINE FEED"
2050      001157      000      ;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
2051      001160      000000      ;CONTAINS THE ADDRESS FROM
2052      ;WHICH ($REG0) WAS OBTAINED
2053      001162      000000      ;CONTAINS (($REGAD)+0)
2054      001164      000000      ;CONTAINS (($REGAD)+2)
2055      001166      000000      ;CONTAINS (($REGAD)+4)
2056      001170      000000      ;CONTAINS (($REGAD)+6)
2057      001172      000000      ;CONTAINS (($REGAD)+10)
2058      001174      000000      ;CONTAINS (($REGAD)+12)
2059      001176      000000      ;CONTAINS (($REGAD)+14)
2060      001200      000000      ;CONTAINS (($REGAD)+16)
2061      001202      000000      ;CONTAINS (($REGAD)+20)
2062      001204      000000      ;CONTAINS (($REGAD)+22)
2063      001206      000000      ;CONTAINS (($REGAD)+24)
2064      001210      000000      ;CONTAINS (($REGAD)+26)
2065      001212      000000      ;CONTAINS (($REGAD)+30)
2066      001214      000000      ;CONTAINS (($REGAD)+32)
2067      001216      000000      ;CONTAINS (($REGAD)+34)
2068      001220      000000      ;CONTAINS (($REGAD)+36)
2069      001222      000000      ;CONTAINS (($REGAD)+40)

```

2070	001224	000000	\$REG21: .WORD	0	:: CONTAINS ((\$REGAD)+42)
2071	001226	000000	\$REG22: .WORD	0	:: CONTAINS ((\$REGAD)+44)
2072	001230	000000	\$REG23: .WORD	00	:: CONTAINS ((\$REGAD)+46)
2073	001232	000000	\$REG24: .WORD	00	:: CONTAINS ((\$REGAD)+50)
2074	001234	000000	\$REG25: .WORD	00	:: CONTAINS ((\$REGAD)+52)
2075	001236	000000	\$REG26: .WORD	00	:: CONTAINS ((\$REGAD)+54)
2076	001240	000000	\$REG27: .WORD	00	:: CONTAINS ((\$REGAD)+56)
2077	001242	000000	\$REG30: .WORD	00	:: CONTAINS ((\$REGAD)+60)
2078	001244	000000	\$REG31: .WORD	00	:: CONTAINS ((\$REGAD)+62)
2079	001246	000000	\$REG32: .WORD	00	:: CONTAINS ((\$REGAD)+64)
2080	001250	000000	\$REG33: .WORD	00	:: CONTAINS ((\$REGAD)+66)
2081	001252	000000	\$REG34: .WORD	00	:: CONTAINS ((\$REGAD)+70)
2082	001254	000000	\$REG35: .WORD	00	:: CONTAINS ((\$REGAD)+72)
2083	001256	000000	\$REG36: .WORD	00	:: CONTAINS ((\$REGAD)+74)
2084	001260	000000	\$REG37: .WORD	00	:: CONTAINS ((\$REGAD)+76)
2085	001262	000000	\$TMP0: .WORD	00	:: USER DEFINED
2086	001264	000000	\$TMP1: .WORD	00	:: USER DEFINED
2087	001266	000000	\$TMP2: .WORD	00	:: USER DEFINED
2088	001270	000000	\$TMP3: .WORD	00	:: USER DEFINED
2089	001272	000000	\$TMP4: .WORD	00	:: USER DEFINED
2090	001274	000000	\$TMP5: .WORD	00	:: USER DEFINED
2091	001276	000000	\$TMP6: .WORD	00	:: USER DEFINED
2092	001300	000000	\$TMP7: .WORD	00	:: USER DEFINED
2093	001302	000000	\$TMP10: .WORD	0	:: USER DEFINED
2094	001304	000000	\$TIMES: 0		:: MAX. NUMBER OF ITERATIONS
2095	001306	000000	\$ESCAPE: 0		:: ESCAPE ON ERROR ADDRESS
2096	001310	177607	\$BELL: .ASCIZ <207><377><377>		:: CODE FOR BELL
2097	001314	077	\$QUES: .ASCII /?/		:: QUESTION MARK
2098	001315	015	\$CRLF: .ASCII <15>		:: CARRIAGE RETURN
2099	001316	000012	\$LF: .ASCIZ <12>		:: LINE FEED
2100			;*****		
2101			.SBTTL APT MAILBOX-ETABLE		
2102			;*****		
2103			.EVEN		
2104			\$MAIL: .WORD AMSGTY :: APT MAILBOX		
2105	001320	000000	\$MSGTY: .WORD AFATAL :: MESSAGE TYPE CODE		
2106	001320	000000	\$FATAL: .WORD ATESTN :: FATAL ERROR NUMBER		
2107	001322	000000	\$TESTN: .WORD APASS :: TEST NUMBER		
2108	001324	000000	\$PASS: .WORD ADEVCT :: PASS COUNT		
2109	001326	000000	\$DEVCT: .WORD AUNIT :: DEVICE COUNT		
2110	001330	000000	\$UNIT: .WORD AMSGAD :: I/O UNIT NUMBER		
2111	001332	000000	\$MSGAD: .WORD AMSGLG :: MESSAGE ADDRESS		
2112	001334	000000	\$MSGLG: .WORD \$ETABLE: :: MESSAGE LENGTH		
2113	001336	000000	\$ENV: .BYTE AENV :: APT ENVIRONMENT TABLE		
2114	001340	000	\$ENVM: .BYTE AENV :: ENVIRONMENT BYTE		
2115	001341	000	\$SWREG: .WORD ASWREG :: ENVIRONMENT MODE BITS		
2116	001342	000000	\$USWR: .WORD AUSWR :: APT SWITCH REGISTER		
2117	001344	000000	\$CPUOP: .WORD ACPUOP :: USER SWITCHES		
2118	001346	000000			:: CPU TYPE, OPTIONS
2119					BITS 15-11=CPU TYPE
2120					11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
2121					11/70=06, PQQ=07, Q=10
2122					BIT 10=REAL TIME CLOCK
2123					BIT 9=FLOATING POINT PROCESSOR
2124					BIT 8=MEMORY MANAGEMENT
2125					

E04

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 44
APT MAILBOX-ETABLE

SEQ 0322
SEQ 0043

2126	001350	000	\$MAMS1: .BYTE	AMAMS1	;; HIGH ADDRESS, M.S. BYTE
2127	001351	000	\$MTYP1: .BYTE	AMTYP1	;; MEM. TYPE, BLK#1
2128			.*		MEM. TYPE BYTE -- (HIGH BYTE)
2129			.*		900 NSEC CORE=001
2130			.*		300 NSEC BIPOLAR=002
2131			.*		500 NSEC MOS=003
2132	001352	000000	\$MADR1: .WORD	AMADR1	;; HIGH ADDRESS, BLK#1
2133			.*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
2134	001354	000	\$MAMS2: .BYTE	AMAMS2	;; HIGH ADDRESS, M.S. BYTE
2135	001355	000	\$MTYP2: .BYTE	AMTYP2	;; MEM. TYPE, BLK#2
2136	001356	000000	\$MADR2: .WORD	AMADR2	;; MEM. LAST ADDRESS, BLK#2
2137	001360	000	\$MAMS3: .BYTE	AMAMS3	;; HIGH ADDRESS, M.S. BYTE
2138	001361	000	\$MTYP3: .BYTE	AMTYP3	;; MEM. TYPE, BLK#3
2139	001362	000000	\$MADR3: .WORD	AMADR3	;; MEM. LAST ADDRESS, BLK#3
2140	001364	000	\$MAMS4: .BYTE	AMAMS4	;; HIGH ADDRESS, M.S. BYTE
2141	001365	000	\$MTYP4: .BYTE	AMTYP4	;; MEM. TYPE, BLK#4
2142	001366	000000	\$MADR4: .WORD	AMADR4	;; MEM. LAST ADDRESS, BLK#4
2143	001370	120210	\$VECT1: .WORD	AVECT1	;; INTERRUPT VECTOR#1, BUS PRIORITY#1
2144	001372	000000	\$VECT2: .WORD	AVECT2	;; INTERRUPT VECTOR#2, BUS PRIORITY#2
2145	001374	177440	\$BASE: .WORD	ABASE	;; BASE ADDRESS OF EQUIPMENT UNDER TEST
2146	001376	000377	\$DEVM: .WORD	ADEVM	;; DEVICE MAP
2147	001400		\$ETEND:		
2148			.MEXIT		

```

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

```

```

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

```

SERRTB:

```
;ERROR 1                                ;UNIBUS PARITY ERROR  
EMI  
DH100  
DT100  
DFQI
```

```
:ERROR 2                                ;NON-EXISTANT MEMORY  
EM2  
DH100  
DT100  
DFQ1
```

```
.ERROR 3                                :NON-EXISTANT DRIVE  
EM3  
DH100  
DT100  
DF01
```

```
.ERROR 4                                ;UNIT FIELD ERROR  
EM4  
DH100  
DT100  
DEQ1
```

```
.ERROR 5                                :SUBSYSTEM TIMEOUT  
EM5  
DH100  
DT100  
DF02
```

```
.ERROR 6  
EM6 ;D TO C PARITY ERROR  
DH100  
DT100  
DF03
```

```

:ERROR 7
EM7          ;DRIVE DETECTED PARITY ERROR
DH100
DT100

```

G04

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 46
ERROR POINTER TABLESEQ 0324
SEQ 0045

2205	001466	062704	DF03	
2206				
2207			:ERROR 10	
2208	001470	056530	EM10	;AC LOW
2209	001472	061005	DH100	
2210	001474	062504	DT100	
2211	001476	062650	DF02	
2212				
2213			:ERROR 11	
2214	001500	056537	EM11	;SPEED LOSS
2215	001502	061005	DH100	
2216	001504	062504	DT100	
2217	001506	062650	DF02	
2218				
2219			:ERROR 12	
2220	001510	056552	EM12	;ILLEGAL FUNCTION
2221	001512	061005	DH100	
2222	001514	062504	DT100	
2223	001516	062650	DF02	
2224				
2225			:ERROR 13	
2226	001520	056564	EM13	;PROGRAMMING ERROR
2227	001522	061005	DH100	
2228	001524	062504	DT100	
2229	001526	062620	DF01	
2230				
2231			:ERROR 14	;NON-EXISTANT FUNCTION
2232	001530	056574	EM14	
2233	001532	061005	DH100	
2234	001534	062504	DT100	
2235	001536	062650	DF02	
2236				
2237			:ERROR 15	
2238	001540	056614	EM15	;DRIVE TYPE ERROR
2239	001542	061005	DH100	
2240	001544	062504	DT100	
2241	001546	062650	DF02	
2242				
2243			:ERROR 16	
2244	001550	056630	EM16	;FORMAT ERROR
2245	001552	061005	DH100	
2246	001554	062504	DT100	
2247	001556	062650	DF02	
2248				
2249			:ERROR 17	
2250	001560	056640	EM17	;WRITE LOCK ERROR
2251	001562	061005	DH100	
2252	001564	062504	DT100	
2253	001566	062650	DF02	
2254				
2255			:ERROR 20	
2256	001570	056655	EM20	;DRIVE UNSAFE
2257	001572	061005	DH100	
2258	001574	062504	DT100	
2259	001576	062650	DF02	
2260				

H04

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 47
ERROR POINTER TABLESEQ 0325
SEQ 0046

2261			:ERROR 21	
2262	001600	056670	EM21	;SEEK INCOMPLETE
2263	001602	061005	DH100	
2264	001604	062504	DT100	
2265	001606	062650	DF02	
2266				
2267			:ERROR 22	
2268	001610	056701	EM22	;CYLINDER OVERFLOW
2269	001612	061005	DH100	
2270	001614	062504	DT100	
2271	001616	062650	DF02	
2272				
2273			:ERROR 23	
2274	001620	056714	EM23	;ILLEGAL CYLINDER
2275	001622	061005	DH100	
2276	001624	062504	DT100	
2277	001626	062650	DF02	
2278				
2279			:ERROR 24	
2280	001630	056731	EM24	;DRIVE OFF TRACK
2281	001632	061005	DH100	
2282	001634	062504	DT100	
2283	001636	062650	DF02	
2284				
2285			:ERROR 25	
2286	001640	056745	EM25	;DRIVE TIMING ERROR
2287	001642	061005	DH100	
2288	001644	062504	DT100	
2289	001646	062650	DF02	
2290				
2291			:ERROR 26	
2292	001650	056764	EM26	;DATA LATE
2293	001652	061005	DH100	
2294	001654	062504	DT100	
2295	001656	062650	DF02	
2296				
2297			:ERROR 27	
2298	001660	056776	EM27	;CONTROLLER TIMEOUT
2299	001662	061005	DH100	
2300	001664	062504	DT100	
2301	001666	062650	DF02	
2302				
2303			:ERROR 30	
2304	001670	057013	EM30	;OPERATION INCOMPLETE
2305	001672	061005	DH100	
2306	001674	062504	DT100	
2307	001676	062760	DF05	
2308				
2309			:ERROR 31	
2310	001700	057025	EM31	;HEADER VRC ERROR
2311	001702	061005	DH100	
2312	001704	062504	DT100	
2313	001706	062760	DF05	
2314				
2315			:ERROR 32	
2316	001710	057041	EM32	;DATA CHECK ERROR

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 48
ERROR POINTER TABLESEQ 0326
SEQ 0047

2317	001712	061005	DH100	
2318	001714	062504	DT100	
2319	001716	063014	DF07	
2320				
2321			:ERROR 33	
2322	001720	057056	EM33	;WRITE CHECK ERROR
2323	001722	061005	DH100	
2324	001724	062504	DT100	
2325	001726	063050	DF10	
2326				
2327			:ERROR 34	
2328	001730	057072	EM34	;DATA MISCOMPARE(S)
2329	001732	061005	DH100	
2330	001734	062504	DT100	
2331	001736	062744	DF04	
2332				
2333			:ERROR 35	
2334	001740	057112	EM35	;NO DRIVE RESPONSE-UFE & NXD
2335	001742	061005	DH100	
2336	001744	062504	DT100	
2337	001746	062620	DF01	
2338				
2339			:ERROR 36	
2340	001750	057144	EM36	;DRIVE ERROR WILL NOT CLEAR
2341	001752	000000	0	
2342	001754	000000	0	
2343	001756	000000	0	
2344				
2345			:ERROR 37	
2346	001760	057173	EM37	;DRIVE STATUS CHANGE WILL NOT CLEAR
2347	001762	000000	0	
2348	001764	000000	0	
2349	001766	000000	0	
2350				
2351			:ERROR 40	
2352	001770	057234	EM40	;ATTENTION BUT NO STATUS CHANGE OR FAULT
2353	001772	061005	DH100	
2354	001774	062504	DT100	
2355	001776	062650	DF02	
2356				
2357			:ERROR 41	
2358	002000	057300	EM41	;ATTENTION BUT DRIVE NOT AVAILABLE
2359	002002	061005	DH100	
2360	002004	062504	DT100	
2361	002006	062650	DF02	
2362				
2363			:ERROR 42	
2364	002010	057330	EM42	;ATTENTION WHEN NOT EXPECTED
2365	002012	061005	DH100	
2366	002014	062504	DT100	
2367	002016	062650	DF02	
2368				
2369			:ERROR 43	
2370	002020	057360	EM43	;ERROR WHILE GATHERING DRIVE STATUS
2371	002022	061005	DH100	
2372	002024	062504	DT100	

J04

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 49
ERROR POINTER TABLESEQ 0327
SEQ 0048

2373	002026	063114	DF12	
2374				
2375			:ERROR 44	
2376	002030	057560	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2377	002032	061005	DH100	
2378	002034	062504	DT100	
2379	002036	063114	DF12	
2380				
2381			:ERROR 45	
2382	002040	057616	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2383	002042	061005	DH100	
2384	002044	062504	DT100	
2385	002046	063114	DF12	
2386				
2387			:ERROR 46	
2388	002050	057644	EM65	;UNSOLICITED ATTENTION
2389	002052	061005	DH100	
2390	002054	062504	DT100	
2391	002056	063114	DF12	
2392				
2393			:ERROR 47	
2394	002060	057666	EM66	;UNEXPECTED DATA TYPE ERROR
2395	002062	061005	DH100	
2396	002064	062504	DT100	
2397	002066	063114	DF12	
2398				
2399			:ERROR 50	
2400	002070	057705	EM67	;ATTENTION DID NOT RESET WITH CLEAR
2401	002072	061005	DH100	
2402	002074	062504	DT100	
2403	002076	063114	DF12	
2404				
2405			:ERROR 51	
2406	002100	057744	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2407	002102	061005	DH100	
2408	002104	062504	DT100	
2409	002106	063114	DF12	
2410				
2411			:ERROR 52	
2412	002110	057411	EM52	;MULTIPLE DRIVE SELECT
2413	002112	061005	DH100	
2414	002114	062504	DT100	
2415	002116	063114	DF12	
2416				
2417			:ERROR 53	
2418	002120	057426	EM53	;ABREVIATED HCE ERROR
2419	002122	061005	DH100	
2420	002124	062504	DT100	
2421	002126	063144	DF13	
2422				
2423			:ERROR 54	
2424	002130	057013	EM30	;OPERATION INCOMPLETE ERROR
2425	002132	061005	DH100	
2426	002134	062504	DT100	
2427	002136	063174	DF14	
2428				

K04

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 50
ERROR POINTER TABLESEQ 0328
SEQ 0049C
C

2429			:ERROR 55	
2430	002140	057025	EM31	;ABREVIATED HVRC ERROR
2431	002142	061005	DH100	
2432	002144	062504	DT100	
2433	002146	063144	DF13	
2434				
2435			:ERROR 56	
2436	002150	057443	EM56	;2 TIMEOUT ERROR
2437	002152	061005	DH100	
2438	002154	062504	DT100	
2439	002156	063224	DF15	
2440				
2441			:ERROR 57	;2ND LEVEL IN SUBSYSTEM TIMEOUT
2442	002160	057443	EM56	
2443	002162	061005	DH100	
2444	002164	062504	DT100	
2445	002166	063270	DF16	
2446				
2447			:ERROR 60	
2448	002170	057462	EM60	;ERROR IN RECAL FOR RECOVERY
2449	002172	000000	0	
2450	002174	000000	0	
2451	002176	000000	0	
2452				
2453			:ERROR 61	
2454	002200	057514	EM61	;ABORT MESSAGE
2455	002202	000000	0	
2456	002204	000000	0	
2457	002206	000000	0	
2458				
2459			:ERROR 62	
2460	002210	057544	EM62	;CYLINDER MISCOMPARE
2461	002212	061005	DH100	
2462	002214	062504	DT100	
2463	002216	063334	DF17	
2464				
2465			:ERROR 63	;DATA ERROR WORDS
2466	002220	000000	0	
2467	002222	000000	0	
2468	002224	062576	DT602	
2469	002226	063444	DF25	
2470				
2471			:ERROR 64	
2472	002230	057560	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
2473	002232	061005	DH100	
2474	002234	062504	DT100	
2475	002236	062650	DF02	
2476				
2477			:ERROR 65	
2478	002240	057616	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
2479	002242	061005	DH100	
2480	002244	062504	DT100	
2481	002246	062650	DF02	
2482				
2483			:ERROR 66	
2484	002250	057644	EM65	;UNSOLICITED ATTENTION

L04

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 51
ERROR POINTER TABLESEQ 0329
SEQ 0050

2485	002252	061005	DH100	
2486	002254	062504	DT100	
2487	002256	062650	DF02	
2488				
2489				
2490	002260	057666	EM66	; UNEXPECTED DATA TYPE ERROR
2491	002262	061005	DH100	
2492	002264	062504	DT100	
2493	002266	062650	DF02	
2494				
2495				
2496	002270	057705	EM67	; ATTENTION DID NOT RESET WITH CLEAR
2497	002272	061005	DH100	
2498	002274	062504	DT100	
2499	002276	062650	DF02	
2500				
2501				
2502	002300	057744	EM70	; SUBSYSTEM CLEAR DID NOT CLEAR ATT
2503	002302	061005	DH100	
2504	002304	062504	DT100	
2505	002306	062650	DF02	
2506				
2507				
2508	002310	060011	EM71	; DATA LATE WHEN UNLOADING HEADER
2509	002312	061005	DH100	
2510	002314	062504	DT100	
2511	002316	062650	DF02	
2512				
2513				
2514	002320	060044	EM72	; CONTROLLER ERROR DURING DRIVER SERVICE
2515	002322	061005	DH100	
2516	002324	062504	DT100	
2517	002326	062650	DF02	
2518				
2519				
2520	002330	060077	EM73	; DRIVE DETECTED PARITY ERROR
2521	002332	061005	DH100	
2522	002334	062504	DT100	
2523	002336	062650	DF02	
2524				
2525				
2526	002340	060117	EM74	; UNDEFINED ERROR
2527	002342	061005	DH100	
2528	002344	062504	DT100	
2529	002346	062650	DF02	
2530				
2531				
2532	002350	060131	EM75	; MARKING SECTOR BAD MESSAGE
2533	002352	000000	0	
2534	002354	000000	0	
2535	002356	000000	0	
2536				
2537				
2538	002360	060161	EM76	; BAD DATA VERIFICATION WITH READ
2539	002362	061717	DH605	
2540	002364	062570	DT601	

M04

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 52
ERROR POINTER TABLESEQ 0330
SEQ 0051C
C

2541	002366	063370	DF21	
2542				
2543			; ERROR 100	
2544	002370	060243	EM77	; RETRY SUCCESSFUL MESSAGE
2545	002372	000000	0	
2546	002374	062570	DT601	
2547	002376	063430	DF23	
2548				
2549			; ERROR 101	
2550	002400	060243	EM77	; ANOTHER RETRY SUCCESSFUL MESSAGE
2551	002402	000000	0	
2552	002404	062570	DT601	
2553	002406	063430	DF23	
2554				
2555			; ERROR 102	
2556	002410	060264	EM100	; RETRY UNSUCCESSFUL MESSAGE
2557	002412	000000	0	
2558	002414	062570	DT601	
2559	002416	063430	DF23	
2560				
2561			; ERROR 103	
2562	002420	060307	EM101	; NO VALID HEADERS IN TRACK JUST READ
2563	002422	061701	DH6042	
2564	002424	062570	DT601	
2565	002426	063440	DF24	
2566				
2567			; ERROR 104	
2568	002430	060365	EM102	; BSE ERROR ON SECTOR NOT LISTED AS BAD
2569	002432	061005	DH100	
2570	002434	062504	DT100	
2571	002436	062650	DF02	
2572				
2573			; ERROR 105	
2574	002440	060431	EM103	; TIMED-OUT ON READ HEADER
2575	002442	061005	DH100	
2576	002444	062504	DT100	
2577	002446	062704	DF03	
2578				
2579			; ERROR 106	
2580	002450	060457	EM104	; TIMED-OUT ON SEEK
2581	002452	061005	DH100	
2582	002454	062504	DT100	
2583	002456	062704	DF03	
2584				
2585			; ERROR 107	
2586	002460	060501	EM105	; DRIVE SIEZED BY OTHER PORT
2587	002462	061005	DH100	
2588	002464	062504	DT100	
2589	002466	062650	DF02	
2590				
2591			; ERROR 110	
2592	002470	060532	EM106	; "DATA MISCMPR WHILE BAI SET"
2593	002472	061005	DH100	
2594	002474	062504	DT100	
2595	002476	063460	DF27	
2596				

NO4

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 53
ERROR POINTER TABLE

SEQ 0331
SEQ 0052

C2
C2

2597			:ERROR 111	
2598	002500	060565	EM107	; "NO NEM WHEN EXPECTED"
2599	002502	000000	0	
2600	002504	000000	0	
2601	002506	000000	0	
2602				
2603			:ERROR 112	
2604	002510	060630	EM110	; "INTRPT WHEN CNTRLR NOT READY"
2605	002512	061005	DH100	
2606	002514	062504	DT100	
2607	002516	062620	DF01	
2608				
2609			:ERROR 113	
2610	002520	060663	EM111	; "NO ATT'N ON SEEK"
2611	002522	061005	DH100	
2612	002524	062504	DT100	
2613	002526	062650	DF02	
2614				
2615			:ERROR 114	
2616	002530	060704	EM112	; DRIVE'S CYLINDER INCORRECT
2617	002532	062126	DH702	
2618	002534	062544	DT202	
2619	002536	063450	DF26	
2620				
2621			:ERROR 115	
2622	002540	000000	0	; TYPE ADRS OF DATA MISCOMPARE(S)
2623	002542	000000	0	
2624	002544	062570	DT601	
2625	002546	063074	DF11	
2626				
2627			:ERROR 116	
2628	002550	057072	EM34	; DATA MISCOMPARE (11/70)
2629	002552	061263	DH103	
2630	002554	062614	DT103	
2631	002556	063360	DF20	
2632				
2633			:ERROR 117	
2634	002560	000000	0	; PART OF DATA MISCOMPARE
2635	002562	000000	0	
2636	002564	062504	DT100	
2637	002566	063400	DF22	
2638				
2639			:ERROR 120	
2640	002570	060726	EM113	; ABORT- CAN'T READ BSF
2641	002572	061005	DH100	
2642	002574	062504	DT100	
2643	002576	062620	DF01	
2644				
2645			:ERROR 121	
2646	002600	060754	EM114	; KT11 FAILURE
2647	002602	061005	DH100	
2648	002604	062504	DT100	
2649	002606	063504	DF30	
2650				
2651			:ERROR 122	
2652	002610	060771	EM115	; MEM PARITY ERROR

B05

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 54
ERROR POINTER TABLE

SEQ 0332
SEQ 0053

2653 002612 061005
2654 002614 062504
2655 002616 063530
2656
2657
2658 002620 056416
2659 002622 000000
2660 002624 000000
2661 002626 000000
2662
2663
2664
2665
2666
2667

DH100
DT100
DF31

.ERROR 123
EM3

;NED WHEN SIZING UNDER APT DRIVE TABLE

0
0
0

C05

.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	RO	RO	R/W	RO	R/W	R/W	R/W
R/W							
7	6	5	4	3	2	1	0
RDY	IE	SPARE	F4	F3	F2	F1	G0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RKCS2							
15	14	13	12	11	10	9	8
DLT	WCE	UPE	NED	NEM	PGE	MDS	UFE
RO	RO	RO	RO	RO	RO	RO	RO
7	6	5	4	3	2	1	0
OR	IR	CLR	BAI	DESL	DS2	DS1	DS0
RO	RO	WO	R/W	R/W	R/W	R/W	R/W
RKDC (DESIRED CYLINDER)							
15	14	13	12	11	10	9	8
NU	NU	NU	NU	NU	NU	DC9	DC8
			READ ONLY				
7	6	5	4	3	2	1	0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
			READ ONLY				
RKDA (DESIRED TRACK AND SECTOR)							
15	14	13	12	11	10	9	8
NU	UN	UN	UN	UN	TA2	TA1	TA0
			READ ONLY				
7	6	5	4	3	2	1	0
UN	UN	UN	SA4	SA3	SA2	SA1	SA0
			READ ONLY				

2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 56
BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0334
SEQ 0055

[illegible]

```

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
                                ALWAYS!

```

SEQ 0335
SEQ 0056

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 58
BIT ASSIGNMENTS IN THE RK611 REGISTERS

F05

SEQ 0336
SEQ 0057

2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849

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

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 59
BIT ASSIGNMENTS IN THE RK611 REGISTERS

SEQ 0337
SEQ 0058

G05

2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887

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	FOLOW				
			OK				
LINE B MESSAGE 01							
15	14	13	12	11	10	9	8
PAR	SERVO	LIMIT	SEEK	PLO	DIBIT	INDEX	MULTI
	BRAKE	ON SK	NO MO	ERR	ERR	ERR	HD SEL
7	6	5	4	3	2	1	0
HEAD	WR GTE	WR CUR	SEC	NU	NU	0	1
FAULT	AND NO	AND NO	ERR				
	XISTON	WR GTE					

2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
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

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	RKCS1=	0	; CONTROL AND STATUS REGISTER 1
000002	RKWC=	2	; WORD COUNT REGISTER
000004	RKBA=	4	; BUS ADDRESS REGISTER
000006	RKDA=	6	; DESIRED TRACK SECTOR REGISTER
000010	RKCS2=	10	; CONTROL AND STATUS REGISTER 2

```

2944      000012      RKDS= 12      ;DRIVE STATUS REGISTER
2945      000014      RKER= 14      ;ERROR REGISTER
2946      000016      RKASOF= 16     ;ATTENTION SUMMARY AND OFFSET REGISTER
2947      000020      RKDC= 20      ;DESIRED CYLINDER REGISTER
2948      000020      RKDCYL= 20     ;DESIRED CYLINDER REGISTER
2949      000024      RKDB= 24      ;DATA BUFFER
2950      000026      RKMR1= 26     ;MAINTENANCE REGISTER 1
2951      000034      RKMR2= 34     ;MAINTENANCE REGISTER 2
2952      000036      RKMR3= 36     ;MAINTENANCE REGISTER 3
2953      000030      RKPOS= 30     ;ECC POSITION INFORMATION
2954      000030      RKECPS= 30     ;ECC POSITION INFORMATION
2955      000032      RKPAT= 32     ;ECC PATTERN INFORMATION
2956      000032      RKECPT= 32     ;ECC PATTERN INFORMATION
2957
2958      .SBTTL  DRIVE COMMANDS
2959
2960      000101      SELDRV= 101     ;SELECT DRIVE
2961      000103      PACK= 103      ;PACK ACKNOWLEDGE
2962      000105      CLEAR= 105     ;DRIVE CLEAR
2963      000107      UNLOAD= 107    ;UNLOAD
2964      000111      SRTSPL= 111    ;START SPINDLE
2965      000113      RECAL= 113     ;RECALIBRATE
2966      000115      OFFSET= 115    ;OFFSET
2967      000117      SEEK= 117     ;SEEK
2968      000121      RDDATA= 121    ;READ DATA
2969      000123      WRDATA= 123    ;WRITE DATA
2970      000125      RDHEAD= 125    ;READ HEADER
2971      000127      WRHEAD= 127    ;WRITE HEADER AND DATA
2972      000131      WRCHK= 131     ;WRITE CHECK
2973
2974      ;          THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
2975      ;          TO SIMULATE A SPECIFIC DESIRED OPERATION
2976
2977      000140      RELEAS= 140     ;RELEASE DRIVE
2978      000141      ROSTAT= 141     ;GET ALL STATUS FROM DRIVE
2979      000164      RDALHD= 164     ;READ ALL HEADERS
2980      000176      CONCLR= 176     ;CONTROLLER CLEAR (BIT 15 OF CS1)
2981      000177      SUBCLR= 177     ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
2982      000300      INTR= 300       ;GENERATE INTERRUPT TO CPU
2983
2984      ;          DRIVER ISSUED SERVICE COMMANDS
2985
2986      000001      DR.SEL= 001      ;DRIVE SELECT
2987      000005      DR.CLR= 005     ;DRIVE CLEAR
2988
2989      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
2990
2991      000001      GO= BIT0         ;GO BIT
2992      000100      IE= BIT6         ;INTERRUPT ENABLE
2993      000200      RDY= BIT7        ;CONTROLLER READY
2994      000400      BA16= BIT8       ;BUS ADDRESS BIT 16
2995      001000      BA17= BIT9       ;BUS ADDRESS BIT 17
2996      002000      CDT= BIT10      ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
2997      004000      CTO= BIT11      ;CONTROLLER TIMED OUT WAITING FOR
2998      ;          DRIVE RESPONSE
2999      010000      CFMT= BIT12      ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

```

3000	020000	SPAR=	BIT13	;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
3001	040000	DI=	BIT14	;DRIVE INTERRUPT
3002	100000	CERR=	BIT15	;CONTROLLER ERROR
3003	100000	CCLR=	BIT15	;CONTROLLER CLEAR
3004				
3005				
3006				
3007				
3008	000001	B.BA16=	BIT0	;BUS ADDRESS BIT 16
3009	000002	B.BA17=	BIT1	;BUS ADDRESS BIT 17
3010	000004	B.CDT=	BIT2	;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
3011	000020	B.CFMT=	BIT4	;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
3012				
3013				
3014				
3015	000007	.SBTTL	CONTROL AND STATUS REGISTER 2 BITS	
3016	000010	DRVMSK=	7	;MASK FOR DRIVE SELECTION CODE
3017	000010	DESL=	BIT3	;DESELECT OR RELEASE DRIVE IN BITS 0-2
3018	000020	RLS=	BIT3	;DESELECT OR RELEASE DRIVE IN BITS 0-2
3019	000040	BAI=	BIT4	;BUS ADDRESS INCREMENT INHIBIT
3020	000040	CLR=	BIT5	;CLEAR CONTROLLER AND ALL DRIVES
3021	000100	SCLR=	BIT5	;CLEAR CONTROLLER AND ALL DRIVES
3022	000200	IR=	BIT6	;INPUT READY
3023	000400	OR=	BIT7	;OUTPUT READY
3024	001000	UFE=	BIT8	;UNIT FIELD ERROR
3025	002000	MDS=	BIT9	;MULTIPLE DRIVE SELECT
3026	004000	PGE=	BIT10	;PROGRAMMING ERROR
3027	010000	NEM=	BIT11	;NON-EXISTENT MEMORY
3028	020000	NED=	BIT12	;NON-EXISTENT DRIVE
3029	040000	UPE=	BIT13	;UNIBUS PARITY ERROR
3030	100000	WCE=	BIT14	;WRITE CHECK ERROR
3031		DLT=	BIT15	;DATA LATE ERROR
3032				
3033		.SBTTL	ERROR REGISTER BIT DEFINITION	
3034	000001	ILC=	BIT0	;ILLEGAL FUNCTION CODE
3035		*ILF=	BIT0	;ILLEGAL FUNCTION CODE
3036	000002	SKI=	BIT1	;SEEK INCOMPLETE
3037	000004	ILF=	BIT2	;ILLEGAL DRIVE FUNCTION
3038	000004	NXF=	BIT2	;ILLEGAL DRIVE FUNCTION
3039	000010	DRPAR=	BIT3	;DRIVE DETECTED DRIVE BUS PARITY ERROR
3040	000020	FMTE=	BIT4	;FORMAT ERROR
3041	000040	DTYE=	BIT5	;DRIVE TYPE ERROR
3042	000100	ECH=	BIT6	;ECC HARD
3043	000200	BSE=	BIT7	;BAD SECTOR ERROR
3044	000400	HCRC=	BIT8	;HEADER CRC ERROR
3045	000400	HVRC=	BIT8	;HEADER VRC ERROR
3046	001000	COE=	BIT9	;CYLINDER ADDRESS OVERFLOW ERROR
3047	002000	IDAE=	BIT10	;INVALID DISK ADDRESS ERROR
3048	004000	WLE=	BIT11	;WRITE LOCK ERROR
3049	010000	DTE=	BIT12	;DRIVE TIMING ERROR
3050	020000	OPI=	BIT13	;OPERATION (SEARCH) INCOMPLETE
3051	040000	UNS=	BIT14	;DRIVE UNSAFE
3052	100000	DCK=	BIT15	;DATA CHECK
3053				
3054		.SBTTL	STATUS REGISTER BIT DEFINITION	
3055				

K05

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 63
STATUS REGISTER BIT DEFINITIONSEQ 0341
SEQ 0062

3056	000001	DRA=	BIT0	; DRIVE AVAILABLE (CONTROLLER IS SET IF
3057				; THIS BIT IS RESET)
3058	000004	OFST=	BIT2	; DRIVE OFFSET
3059	000010	ACLO=	BIT3	; AC LOW
3060	000020	SPDLSS=	BIT4	; SPEED LOSS
3061	000020	DCLO=	BIT4	; DC LOW
3062	000040	DROT=	BIT5	; DRIVE OFF TRACK
3063	000100	VV=	BIT6	; VOLUME VALID
3064	000200	DRY=	BIT7	; DRIVE READY
3065	000200	CRDY=	BIT7	; DRIVE READY
3066	000400	ODT=	BIT8	; DRIVE TYPE (0=RK06, 1=RK07)
3067	004000	WRL=	BIT11	; WRITE LOCK
3068	020000	PIP=	BIT13	; POSITIONING IN PROGRESS
3069	040000	DSC=	BIT14	; DRIVE STATUS CHANGE
3070	100000	SVAL=	BIT15	; STATUS VALID
3071				
3072		.SBTTL	MAINTENANCE REGISTER 1 BIT DEFINITION	
3073				
3074	000017	MESMSK=	17	; MESSAGE MASK
3075				
3076	000020	PAT=	BIT4	; FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
3077	000040	DMD=	BIT5	; DIAGNOSTIC MODE
3078	000100	MSP=	BIT6	; MAINTENANCE SECTOR PULSE
3079	000200	MIND=	BIT7	; MAINTENANCE INDEX
3080	000400	MCLK=	BIT9	; MAINTENANCE CLOCK
3081	001000	MERD=	BIT9	; MAINTENANCE ENCODED READ DATA
3082	002000	MEWD=	BIT10	; MAINTENANCE ENCODED WRITE DATA
3083	004000	PCA=	BIT11	; PRECOMPENSATION ADVANCE
3084	010000	PCD=	BIT12	; PRECOMPENSATION DELAY
3085	020000	ECCW=	BIT13	; ECC WORD IS BEING READ OR WRITTEN
3086	040000	WRTGAT=	BIT14	; WRITE GATE
3087	100000	RDGATE=	BIT15	; READ GATE
3088				
3089		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A	
3090				
3091	000040	S.DRA=	BIT5	; DRIVE AVAILIABLE
3092	000100	S.VV=	BIT6	; VOLUME VALID
3093	000200	S.DRY=	BIT7	; DRIVE READY
3094	000400	S.TYPE=	BIT8	; DRIVE TYPE
3095	001000	S.FORM=	BIT9	; DRIVE FORMAT
3096	002000	S.OFF=	BIT10	; OFFSET
3097	004000	S.WRL=	BIT11	; WRITE LOCK
3098	010000	S.SPIN=	BIT12	; SPINDLE ON
3099	020000	S.PIP=	BIT13	; POSITIONING IN PROGRESS
3100	040000	S.DSC=	BIT14	; DRIVE STATUS CHANGE
3101				
3102		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B	
3103				
3104	000040	S.ICYL=	BIT5	; ILLEGAL CYLINDER ADDRESS
3105	000100	S.ACLO=	BIT6	; AC LOW
3106	000200	S.FLT=	BIT7	; DRIVE FAULT
3107	000400	S.ILF=	BIT8	; ILLEGAL FUNCTION
3108	001000	S.PAR=	BIT9	; DRIVE DETECTED DRIVE BUS PARITY ERROR
3109	002000	S.SKI=	BIT10	; SEEK INCOMPLETE
3110	004000	S.WLE=	BIT11	; WRITE LOCK ERROR
3111	010000	S.SPLS=	BIT12	; SPEED LOSS

L05

CZR6MDO RK611/06 SS VERIF 1
CZR6MC.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 64
DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

SEQ 0342
SEQ 0063

3112	010000	S.DCLO= BIT12	;DC LOW
3113	020000	S.DROT= BIT13	;DRIVE OFF TRACK
3114	040000	S.UNS= BIT14	;DRIVE UNSAFE
3115			
3116		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3117			
3118	000020	S.XDOK= BIT4	;TRANSDUCER OK
3119	000040	S.HDAM= BIT5	;HEADS HOME
3120	000100	S.BRHM= BIT6	;BRUSHES HOME
3121	000200	S.DOOR= BIT7	;DOOR INTERLOCKED
3122	000400	S.CART= BIT8	;CARTRAGE INTERLOCK
3123	001000	S.SPOK= BIT9	;SPEED OK
3124	002000	S.FWD= BIT10	;FORWARD
3125	004000	S.REV= BIT11	;REVERSE
3126	010000	S.LOAD= BIT12	;HEADS LOADING
3127	020000	S.RTZ= BIT13	;RETURN TO ZERO
3128	040000	S.UNLD= BIT14	;HEADS UNLOADING
3129			
3130		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3131			
3132	000020	S.SECT= BIT4	;SECTOR ERROR
3133	000040	S.WCLK= BIT5	;WRITE CLOCK AND NO WRITE GATE
3134	000100	S.WGAT= BIT6	;WRITE GATE AND NO TRANSISTIONS
3135	000200	S.HOFL= BIT7	;HEAD FAULT
3136	000400	S.MHO= BIT8	;MULTIPLE HEAD SELECT
3137	001000	S.XERR= BIT9	;INDEX ERROR
3138	002000	S.DIB= BIT10	;DIBIT ERROR
3139	004000	S.PLO= BIT11	;PLO ERROR
3140	010000	S.NMOV= BIT12	;SEEK AND NO MOTION
3141	020000	S.LIMD= BIT13	;LIMIT DETECT ON SEEK
3142	040000	S.BRKE= BIT14	;SERVO-BRAKE
3143			
3144		.SBTTL	COMMON MASKS
3145			
3146	000007	M.DRV= 7	;DRIVE CODE
3147	100000	M.PAR= BIT15	;PARITY
3148	000003	M.ID= 3	;BYTE ID
3149	017760	M.COIF= 17760	;CYLINDER DIFFERENCE/OFFSET
3150	017760	M.CADD= 17760	;CYLINDER ADDRESS
3151	077770	M.SER= 77770	;DRIVE SERIAL NUMBER
3152	000760	M.SECT= 760	;SECTOR COUNT
3153	007000	M.HEAD= 7000	;HEAD DECODE

.SBTTL PARAMETER BLOCK ALLOCATION

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

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

.SBTTL PARAMETERS PASSED TO THE DRIVER

THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
TO THE RK06/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	DRV PDT= BIT2	; DRIVE POSITIONED FOR DATA TRANSFER
000010	UEXATT= BIT3	; UNEXPECTED ATTENTION
000020	DRVHRD= BIT4	; DRIVE HAS HARD ERROR
000040	DRV DSC= BITS	; DRIVE STATUS CHANGE DID NOT CLEAR

N05

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 66
PROGRAM DEVICE STATUS REGISTER DEFINITION

SEQ 0344
SEQ 0065

C2
C2

3210	000100	CMDT0= BIT6	; NO TERMINATION TO COMMAND FOR AT
3211			; LEAST 1 SECOND
3212	000200	W.WCK= BIT7	; WRITE FOR WRITE WRITE CHECK
3213	000400	NOCHK= BIT8	; NO CHECK, DO NOT SET INTERRUPT ENABLE
3214	001000	PBSVAL= BIT9	; PARAMETER STATUS WORDS VALID
3215			; (SET WHEN ERROR TERMINATION OR
3216			; READ STATUS COMMAND)
3217	002000	DRPDRV= BIT10	; DROP DRIVE FROM TEST SEQUENCE
3218	004000	NODSC= BIT11	; ATTENTION SET BUT DCS AND FAULT RESET
3219	010000	DRVSZD= BIT12	; DRIVE SEIZED BY OTHER PORT
3220	020000	E.UNLD= BIT13	; DRIVE UNLOADED DUE TO ERROR
3221	040000	Q.INIT= BIT14	; PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
3222	100000	DTBAII= BIT15	; INHIBIT BUS ADDRESS INCREMENT
3223			
3224		.SBTTL	PARAMETERS PASSED FROM DRIVER TO PROGRAM
3225			
3226			THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
3227			FROM THE DRIVER TO THE CALLING PROGRAM
3228			
3229	000016	P.CS1= 16	; COMMAND AND STATUS REGISTER 1
3230	000020	P.CS2= 20	; COMMAND AND STATUS REGISTER 2
3231	000022	P.WCR= 22	; WORD COUNT REGISTER
3232	000024	P.BAR= 24	; BUS ADDRESS REGISTER
3233	000026	P.DTS= 26	; DESIRED TRACK SECTOR REGISTER
3234	000030	P.DCYL= 30	; DESIRED CYLINDER REGISTER
3235	000032	P.ASOF= 32	; ATTENTION SUMMARY/OFFSET REGISTER
3236	000034	P.ER= 34	; ERROR REGISTER
3237	000036	P.OS= 36	; STATUS REGISTER
3238	000040	P.A00= 40	; MESSAGE A STATUS BYTE 00
3239	000042	P.B00= 42	; MESSAGE B STATUS BYTE 00
3240	000044	P.A01= 44	; MESSAGE A STATUS BYTE 01
3241	000046	P.B01= 46	; MESSAGE B STATUS BYTE 01
3242	000050	P.A10= 50	; MESSAGE A STATUS BYTE 10
3243	000052	P.B10= 52	; MESSAGE B STATUS BYTE 10
3244	000054	P.A11= 54	; MESSAGE A STATUS BYTE 11
3245	000056	P.B11= 56	; MESSAGE B STATUS BYTE 11
3246	000060	P.EPOS= 60	; ECC POSITION INFORMATION
3247	000062	P.EPAT= 62	; ECC PATTERN INFORMATION
3248			
3249		.SBTTL	PARAMETER BLOCK 0 FOR DRIVE
3250			
3251	002630	000	PARMO: .BYTE 0 ; DRIVE NUMBER
3252	002631	000	.BYTE 0 ; COMMAND
3253	002632	000000	.WORD 0 ; CYLINDER ADDRESS
3254	002634	000	.BYTE 0 ; SECTOR ADDRESS
3255	002635	000	.BYTE 0 ; TRACK ADDRESS
3256	002636	000	.BYTE 0 ; OFFSET VALUE
3257	002637	000	.BYTE 0 ; BUS ADDRESS (BITS 16 AND 17)
3258	002640	000000	.WORD 0 ; BUS ADDRESS (BITS 0 - 15)
3259	002642	000000	.WORD 0 ; WORD COUNT (2'S COMPLEMENT)
3260	002644	000000	.WORD 0 ; PROGRAM DRIVE STATUS INFORMATION
3261	002646	000000	.WORD 0 ; COMMAND AND STATUS REGISTER 1
3262	002650	000000	.WORD 0 ; COMMAND AND STATUS REGISTER 2
3263	002652	000000	.WORD 0 ; WORD COUNT REGISTER
3264	002654	000000	.WORD 0 ; BUS ADDRESS REGISTER
3265	002656	000000	.WORD 0 ; DESIRED TRACK AND SECTOR REGISTER

3266	002660	000000	.WORD	0	; DESIRED CYLINDER REGISTER
3267	002662	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
3268	002664	000000	.WORD	0	; ERROR REGISTER
3269	002666	000000	.WORD	0	; STATUS REGISTER
3270	002670	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
3271	002672	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
3272	002674	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
3273	002676	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
3274	002700	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
3275	002702	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
3276	002704	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
3277	002706	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
3278	002710	000000	.WORD	0	; ECC POSITION INFORMATION
3279	002712	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL PARAMETER BLOCK 1 FOR DRIVE

3283	002714	000	PARM1: .BYTE	0	; DRIVE NUMBER
3284	002715	000	.BYTE	0	; COMMAND
3285	002716	000000	.WORD	0	; CYLINDER ADDRESS
3286	002720	000	.BYTE	0	; SECTOR ADDRESS
3287	002721	000	.BYTE	0	; TRACK ADDRESS
3288	002722	000	.BYTE	0	; OFFSET VALUE
3289	002723	000	.BYTE	0	; BUS ADDRESS (BITS 16 AND 17)
3290	002724	000000	.WORD	0	; BUS ADDRESS (BITS 0 - 15)
3291	002726	000000	.WORD	0	; WORD COUNT (2'S COMPLEMENT)
3292	002730	000000	.WORD	0	; PROGRAM DRIVE STATUS INFORMATION
3293	002732	000000	.WORD	0	; COMMAND AND STATUS REGISTER 1
3294	002734	000000	.WORD	0	; COMMAND AND STATUS REGISTER 2
3295	002736	000000	.WORD	0	; WORD COUNT REGISTER
3296	002740	000000	.WORD	0	; BUS ADDRESS REGISTER
3297	002742	000000	.WORD	0	; DESIRED TRACK AND SECTOR REGISTER
3298	002744	000000	.WORD	0	; DESIRED CYLINDER REGISTER
3299	002746	000000	.WORD	0	; ATTENTION SUMMARY/OFFSET REGISTER
3300	002750	000000	.WORD	0	; ERROR REGISTER
3301	002752	000000	.WORD	0	; STATUS REGISTER
3302	002754	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 00
3303	002756	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 00
3304	002760	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 01
3305	002762	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 01
3306	002764	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 10
3307	002766	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 10
3308	002770	000000	.WORD	0	; MESSAGE LINE A STATUS BYTE 11
3309	002772	000000	.WORD	0	; MESSAGE LINE B STATUS BYTE 11
3310	002774	000000	.WORD	0	; ECC POSITION INFORMATION
3311	002776	000000	.WORD	0	; ECC PATTERN INFORMATION

.SBTTL TEMPORARY CONTROLLER REGISTER STORAGE

3315	003000	000000	T.CS1: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 1
3316					
3317	003002	000000	T.CS2: .WORD	0	; TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 2
3318					
3319	003004	000000	T.WCR: .WORD	0	; TEMPORARY STORAGE FOR WORD COUNT REGISTER
3320	003006	000000	T.BA: .WORD	0	; TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
3321	003010	000000	T.DA: .WORD	0	; TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

3322	003012	000000	T.DC: .WORD	0	; TEMPORARY STORAGE FOR DRIVE CYLINDER
3323	003014	000000	T.ASOF: .WORD	0	; TEMPORARY STORAGE FOR ATTENTION SUMMARY
3324					AND OFFSET
3325	003016	000000	T.ER: .WORD	0	; TEMPORARY STORAGE FOR ERROR REGISTER
3326	003020	000000	T.DS: .WORD	0	; TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3327	003022	000000	T.MR1: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3328	003024	000000	T.MR2: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3329	003026	000000	T.MR3: .WORD	0	; TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3330	003030	000000	T.POS: .WORD	0	; TEMPORARY STORAGE FOR ECC POSITION
3331	003032	000000	T.PAT: .WORD	0	; TEMPORARY STORAGE FOR ECC PATTERN
3332	003034	000000	T.DB: .WORD	0	; TEMPORARY STORAGE FOR DATA BUFFER REGISTER
3333					
3334			.SBTTL DRIVER PARAMETERS		
3335					
3336	003036	177440	RKBAS: .WORD	177440	; ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3337	003040	000210	RKVEC: .WORD	210	; ADDRESS OF R611 VECTOR
3338	003042	000240	RKPRI: .WORD	PR5	; RK611 INTERRUPT PRIORITY
3339	003044	040716	A.NORM: ERRFRE		; ADDRESS OF NORMAL RETURN FROM DRIVER
3340	003046	042164	A.ABNL: ERRHDL		; ADDRESS OF ABNORMAL RETURN FROM DRIVER
3341	003050	041460	A.CONT: CONERR		; ADDRESS OF CONTROLLER ERROR RETURN
3342	003052	000000	E.CONT: .WORD	0	; CONTROLLER ERROR STATUS
3343					THIS LOCATION IS CLEARED WHEN EVERY COMMAND
3344					IS INITIATED. IF A CONTROLLER ERROR
3345					OCCURS THE FOLLOWING BIT ASSIGNMENT IS
3346					USED:
3347					
3348		000001	E.CCLR= BIT0		; CLEAR CONTROLLER DID NOT CLEAR ERROR
3349		000002	E.NOAT= BIT1		; NO ATTENTION IN ATTENTION SUMMARY REG
3350		000004	E.UATT= BIT2		; UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3351		000010	E.UDAT= BIT3		; UNEXPECTED DATA TYPE ERROR
3352		000020	E.CLAT= BIT4		; ATTENTION DID NOT RESET WITH CLEAR
3353		000040	E.SCLR= BIT5		; SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
3354					ATTENTION
3355		000100	E.ILLD= BIT6		; ILLEGAL DRIVER COMMAND
3356		000400	E.DLT= BIT8		; DATA LATE WHEN UNLOADING HEADER
3357		001000	E.CERR= BIT9		; CONTROLLER ERROR DURING DRIVER SERVICING
3358		002000	E.OPAR= BIT10		; DRIVE DETECTED PARITY ERROR
3359		040000	E.CMTO= BIT14		; CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3360		100000	E.MDS= BIT15		; MULTIPLE DRIVE SELECT
3361					
3362	003054	000000	O.WAIT: .WORD	0	; PARAMETER BLOCK OF THE DRIVE
3363					; WAITING FOR COMMAND COMPLETION
3364	003056	000400	W.MTIM: .WORD	400	; LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
3365	003060	000400	W.MILI: .WORD	400	; 16 MILLISECOND TIME FOR PROGRAM
3366					
3367					
3368					
3369					
3370					
3371					
3372					
3373					
3374					
3375					
3376					
3377					

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

3378	003062	000300			W.SEC: .WORD	300	; SECOND COUNT COUNT FOR ALL COMMANDS
3379							; EXCEPT START SPINDLE
3380	003064	003000			W.BSEC: .WORD	3000	; 8 SECOND FOR DRIVE CYCLE DOWN
3381	003066	030000			W.MIN: .WORD	30000	; MINUTE TIME FOR START SPINDLE
3382	003070	000000			HDR.AD: .WORD	0	; ADDRESS USED FOR READ ALL HEADERS
3383	003072	000000			HDR.CT: .WORD	0	; NUMBER OF HEADERS LEFT TO READ FOR READ
3384							; ALL HEADERS
3385	003074	000			I.ISRL: .BYTE	0	; INTERRUPT OR RELEASED COMMAND ISSUED
3386	003075	002	004	010	H.HEAD: .BYTE	2,4,10	; HEAD DECODES
3387	003100	000			W.TIME: .BYTE	0	; DRIVES BEING WATCH-DOG TIMED
3388					.SBTTL	INTERRUPT MASKS	
3389							
3390							
3391	003101	000			INTMSK: .BYTE	0	; INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
3392							
3393						INTERRUPT MASK TABLE	
3394							
3395	003102	001			I.DRV: .BYTE	1	; INTERRUPT MASK FOR DRIVE 0
3396	003103	002			.BYTE	2	; INTERRUPT MASK FOR DRIVE 1
3397	003104	004			.BYTE	4	; INTERRUPT MASK FOR DRIVE 2
3398	003105	010			.BYTE	10	; INTERRUPT MASK FOR DRIVE 3
3399	003106	020			.BYTE	20	; INTERRUPT MASK FOR DRIVE 4
3400	003107	040			.BYTE	40	; INTERRUPT MASK FOR DRIVE 5
3401	003110	100			.BYTE	100	; INTERRUPT MASK FOR DRIVE 6
3402	003111	200			.BYTE	200	; INTERRUPT MASK FOR DRIVE 7
3403							
3404					.SBTTL	PARAMETER BLOCK TABLE	
3405							
3406	003112	002630			PBLKT: PARM0		; ADDRESS OF PARAMETER BLOCK GIVEN WITH
3407							; DRIVE CALL. MUST BE LOADED INTO PBLKT
3408							
3409							
3410					.SBTTL	TIME FOR WATCH-DOG TIMER	
3411							
3412	003114	000000			W.DRV: .WORD	0	; TIME FOR INSTRUCTION IN PARAMETER BLOCK
3413					.SBTTL	PROGRAM SPECIFIC RESERVED LOCATIONS	
3414	003116	000			MDFLAG: .BYTE	0	; FLAG TO INDIC. DEFLT OR PARAM MODE
3415	003117	000			XXDPCH: .BYTE	0	; XXDP CHAIN MODE FLAG
3416	003120	000			TSTING: .BYTE	0	; CURRENTLY RUNNING TESTS IF = 1
3417	003121	000			DERCNT: .BYTE	0	; DATA ERROR COUNT
3418	003122	000			OPCOMP: .BYTE	0	; OPERATION COMPLETE FLAG
3419	003123	000			DONE: .BYTE	0	; DONE SWITCH
3420	003124	000			TYPFMT: .BYTE	0	; DRIVE TYPE & FORMAT CONTROL
3421	003125	000			FORMAT: .BYTE	0	; DRIVE FORMAT IN BIT 4 OF BYTE
3422	003126	000			ERRCNT: .BYTE	0	; ERROR COUNT
3423	003127	004			ERRLMT: .BYTE	4	; ERROR LIMIT
3424	003130	000			DRVERS: .BYTE	0	; ERROR COUNT FOR CURRENT DRIVE
3425	003131	000			OPCONT: .BYTE	0	; OPERATION CONTROL SWITCHES
3426	003132	000			PCLKF: .BYTE	0	; IF BYTE=1, KW11-P CLOCK IS PRESENT
3427	003133	000			DOTIM: .BYTE	0	; IF BYTE=1, DO TIMING TESTS
3428	003134	000			XOVLAD: .BYTE	0	; FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
3429	003135	000			XDPSVD: .BYTE	0	; FLAG = 1 IF XXDP IS SAVED
3430	003136	000			DULACS: .BYTE	0	; FLAG=1 IF DUAL ACCESS TEST
3431	003137	000			ORNAFG: .BYTE	0	; =1 INDICATES DRIVE SIEZED BY OTHER PORT
3432	003140	000			REISSU: .BYTE	0	; DUAL-ACC FLAG TO RE-ISSUE COMMAND
3433	003141	000			WCEFLG: .BYTE	0	; WRITE CHECK ERROR FLAG

E06

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 70
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0348
SEQ 0069

3434	003142	000	DLTFLG: .BYTE	0	; DATA LATE ERROR FLAG
3435	003143	000	NORTRY: .BYTE	0	; "NO-RETRY" FLAG
3436	003144	000	UBMPRS: .BYTE	0	; UNIBUS MAP PRESENT IF = 1
3437	003145	000	MEMABT: .BYTE	0	; SET BYTE = 1 FOR NO ABORT
3438					; ON MEMORY PARITY ERRORS
3439	003146	000	HLPOVL: .BYTE	0	; HELP FILE OVERLAID INDICATOR
3440	003147	000	NOTYPE: .BYTE	0	; INDICATES PROGRAM JUST LOADED IF 0
3441		000001	WHDSW=BIT0		; WRITE HEADER & DATA SWITCH
3442		000002	VFHDSW=BIT1		; VERIFY HEADERS SWITCH
3443		000004	WCDASW=BIT2		; WRITE CHECK DATA SWITCH
3444		000010	RCDASW=BIT3		; READ CHECK DATA SWITCH
3445		000020	OREQSW=BIT4		; OFFSET REQUIRED SWITCH
3446			.EVEN		
3447					
3448	003150	177546	LKS: .WORD	177546	; KW11-L CLOCK STATUS REGISTER
3449	003152	172540	PKS: .WORD	172540	; KW11-P CONTROL AND STATUS REGISTER
3450	003154	172542	PKSB: .WORD	172542	; KW11-P COUNT SET BUFFER REGISTER
3451	003156	172544	PKRB: .WORD	172544	; KW11-P COUNTER REGISTER
3452	003160	000100	LCVEC: .WORD	100	; KW11-L VECTOR STORAGE
3453	003162	000104	PCVEC: .WORD	104	; KW11-P VECTOR STORAGE
3454		000114	MEMVEC=114		; MEMORY PARITY TRAP VECTOR
3455	003164	000000	TCONLO: .WORD	0	; LO BITS OF CALIBRATION TIME CONSTANT
3456	003166	000000	TCONHI: .WORD	0	; HI BITS OF CALIB TIME CONST
3457	003170	000000	BLWMIN: .WORD	0	; COUNT OF TIMES BELOW MIN
3458	003172	000000	ABVMX1: .WORD	0	; COUNT OF FORWARD TIMES ABOVE MAX
3459	003174	000000	ABVMX2: .WORD	0	; COUNT OF REVERSE TIMES ABOVE MAX
3460	003176	000000	SUML01: .WORD	0	; LO BITS OF FORWARD TIME SUM
3461	003200	000000	SUMHI1: .WORD	0	; HI BITS OF FORWARD TIME SUM
3462	003202	000000	SUML02: .WORD	0	; LO BITS OF REVERSE TIME SUM
3463	003204	000000	SUMHI2: .WORD	0	; HI BITS OF REVERSE TIME SUM
3464	003206	000000	SAVPAR: .WORD	0	; SAVE WORD FOR PAR CONSTANT
3465	003210	000000	SAVWRD: .WORD	0	; SCRATCH WORD
3466	003212	000000	MINIL1: .WORD	0	; LO BITS OF MIN MEAS'D TIME (FORWARD)
3467	003214	000000	MINIH1: .WORD	0	; HI BITS OF MIN MEAS'D TIME (FORWARD)
3468	003216	000000	MAXIL1: .WORD	0	; LO BITS OF MAX MEAS'D TIME (FORWARD)
3469	003220	000000	MAXIH1: .WORD	0	; HI BITS OF MAX MEAS'D TIME (FORWARD)
3470	003222	000000	MINIL2: .WORD	0	; LO BITS OF MIN MEAS'D TIME (REVERSE)
3471	003224	000000	MINIH2: .WORD	0	; HI BITS OF MIN MEAS'D TIME (REVERSE)
3472	003226	000000	MAXIL2: .WORD	0	; LO BITS OF MAX MEAS'D TIME (REVERSE)
3473	003230	000000	MAXIH2: .WORD	0	; HI BITS OF MAX MEAS'D TIME (REVERSE)
3474	003232	000400	BSSOFT: .BLKW	↑D256	; RECORD OF BAD SECTORS FROM SOFTWARE
3475	004232	000400	BSFACT: .BLKW	↑D256	; RECORD OF BAD SECTORS FROM FACTORY
3476	005232	000000	PRVCMD: .WORD	0,0,0,0,0,0	; PREVIOUS COMMAND STORAGE
3477	005240	000000			
3478	005246	000000	COMSTR: .WORD	0,0,0,0,0,0	; CURRENT COMMAND STORAGE
3479	005254	000000			
3480	005262	000000	PRMPLO: .WORD	0	; PREV. U.B. MAP REG 0
3481	005264	000000	PRMPHO: .WORD	0	
3482	005266	000000	CRMPLO: .WORD	0	; CURRENT U.B. MAP REG 0
3483	005270	000000	CRMPHO: .WORD	0	
3484	005272	000102	BUFFO: .BLKW	↑D66	; OUTPUT BUFFER 1
3485	005476	000000	LOWOCT: .WORD	0	; LOW 16 BITS OF CONVERTED BINARY NUMBER
3486	005500	000000	HIGOCT: .WORD	0	; HIGH BITS OF CONVERTED BINARY NO.
3487	005502	000000	BUFPRT: .WORD	0	; BUFFER POINTER
3488	005504	000000	RECODE: .WORD	0	; RECOVERY CODE WORD
3489	005506	000000	ERRCOM: .WORD	0	; ERROR COMMAND

3490	005510	000000	DRIVE:	.WORD	0	;NO. OF DRIVE IN USE
3491	005512	000000	STALLS:	.WORD	0	;CURRENT NO. OF UNIT STALLS TO APPLY
3492	005514	000000	CYLNDR:	.WORD	0	;CURRENT CYLINDER NUMBER
3493	005516	000000	FS:	.WORD	0	;FIRST SECTOR LIMIT
3494	005520	000000	LS:	.WORD	0	;LAST SECTOR LIMIT
3495	005522	000000	NCYL1:	.WORD	0	;NEXT CYL SCRATCH WORD
3496	005524	000000	NCYL2:	.WORD	0	;NEXT CYL SCRATCH WORD
3497	005526	000000	OFINUS:	.WORD	0	;OFFSET IN USE
3498	005530	000000	TRACK:	.WORD	0	;TRACK IN USE
3499	005532	000000	INTCHR:	.WORD	0	;TTY INTERRUPT INPUT WORD
3500	005534	000000	SELECT:	.WORD	0	;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3501	005536	000000	ONLINE:	.WORD	0	;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3502	005540	000000	NEWON:	.WORD	0	;NEW LOOK AT ONLINE DRIVES
3503	005542	000000	SCRACH:	.WORD	0	;ALL-PURPOSE SCRATCH WORD
3504	005544	000000	PATRN:	.WORD	0	;DATA PATTERN LIST WORD
3505	005546	000000	XXDPAD:	.WORD	0	;STARTING ADDRESS OF XXDP LOADER
3506	005550	000002	XDPSAV:	.BLKW	2	;XXDP PHYSICAL SAVE ADDRESS
3507	005554	000000	PLOFST:	.WORD	0	; (+) OFFSET VALUE
3508	005556	000000	NGOFST:	.WORD	0	; (-) OFFSET VALUE
3509	005560	000005	SAVPRS:	.BLKW	5	;SAVE INITIAL PARAMS FOR XFER
3510	005572	000000	WDSXFR:	.WORD	0	;NO. OF WORDS ACTUALLY XFERRED
3511	005574	000000	FINCYL:	.WORD	0	;FINAL CYLINDER ADRS
3512	005576	000	FINTRK:	.BYTE	0	;FINAL TRACK ADRS
3513	005577	000	FINSEC:	.BYTE	0	;FINAL SECTOR ADRS
3514	005600	000000	LASTWC:	.WORD	0	;ACTUAL FINAL WC
3515	005602	000002	PMA:	.BLKW	2	;PARTIAL TRANSFER MEMORY START ADRS

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

.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
DTYLST:

.BYTE	0	;DRV 0
.BYTE	0	;DRV 1
.BYTE	0	;DRV 2
.BYTE	0	;DRV 3
.BYTE	0	;DRV 4
.BYTE	0	;DRV 5
.BYTE	0	;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)
†STLST:

3545 005626

G06

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 72
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0350
SEQ 0071

3546	005626	000010	.WORD	10	;TEST	01
3547	005630	000200	.WORD	200	;TEST	02
3548	005632	000010	.WORD	10	;TEST	03
3549	005634	000010	.WORD	10	;TEST	04
3550	005636	000002	.WORD	2	;TEST	05
3551	005640	000002	.WORD	2	;TEST	06
3552	005642	000002	.WORD	2	;TEST	07
3553	005644	000400	.WORD	400	;TEST	10
3554	005646	000500	.WORD	500	;TEST	11
3555	005650	000002	.WORD	2	;TEST	12
3556	005652	000001	.WORD	1	;TEST	13
3557	005654	000001	.WORD	1	;TEST	14
3558	005656	000001	.WORD	1	;TEST	15
3559	005660	000001	.WORD	1	;TEST	16
3560	005662	000010	.WORD	10	;TEST	17
3561	005664	000010	.WORD	10	;TEST	20
3562	005666	000001	.WORD	1	;TEST	21

;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
DFLTST:

3563						
3564						
3565						
3566						
3567	005670					
3568	005670	000010	.WORD	10	;TEST	01
3569	005672	000200	.WORD	200	;TEST	02
3570	005674	000010	.WORD	10	;TEST	03
3571	005676	000010	.WORD	10	;TEST	04
3572	005700	000002	.WORD	2	;TEST	05
3573	005702	000002	.WORD	2	;TEST	06
3574	005704	000002	.WORD	2	;TEST	07
3575	005706	000400	.WORD	400	;TEST	10
3576	005710	000500	.WORD	500	;TEST	11
3577	005712	000002	.WORD	2	;TEST	12
3578	005714	000001	.WORD	1	;TEST	13
3579	005716	000001	.WORD	1	;TEST	14
3580	005720	000001	.WORD	1	;TEST	15
3581	005722	000001	.WORD	1	;TEST	16
3582	005724	000010	.WORD	10	;TEST	17
3583	005726	000010	.WORD	10	;TEST	20
3584	005730	000001	.WORD	1	;TEST	21

NMTSTS=(DFLTST-TSTLST)/2

;TOTAL NO. OF AUTOMATIC TESTS

;OPERATING PARAMETER LIST
PRMLST:

3585						
3586						
3587		000021				
3588						
3589						
3590						
3591	005732					
3592	005732	000000	FC:	.WORD	0	;FIRST CYLINDER
3593	005734	000000	LC:	.WORD	0	;LAST CYLINDER. 632 FOR RK06, 1456 FOR RK07
3594	005736	000001	IC:	.WORD	1	;CYLINDER INCREMENT
3595	005740	000000	FT:	.WORD	0	;FIRST TRACK
3596	005742	000002	LT:	.WORD	2	;LAST TRACK
3597	005744	000001	IT:	.WORD	1	;TRACK INCREMENT
3598	005746	000000	SO:	.WORD	0	;FIRST SECTOR IF 20(DEC) SECTOR FMT
3599	005750	000023	S1:	.WORD	23	;LAST SECTOR IF 20(DEC) SECTOR FMT
3600	005752	000000	S2:	.WORD	0	;FIRST SECTOR IF 22(DEC) SECTOR FMT
3601	005754	000025	S3:	.WORD	25	;LAST SECTOR IF 22(DEC) SECTOR FMT

H06

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 73
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0351
SEQ 0072

3602 005756 000001
3603 005760 000000
3604 005762 063554
3605 005764 000000
3606 005766 000403
3607 005770 000000
3608 005772 000000
3609 005774 001000

IS: .WORD 1 ;SECTOR INCREMENT
PT: .WORD 0 ;DATA PATTERN SELECT WORD
MA: .WORD RWBUF ;LO BITS OF PHYS. MEM. ADDR. (BITS 0-15)
;HI BITS OF PHYS. MEM. ADDR. (BITS 16-21)
WC: .WORD 403 ;WORD COUNT (IF WC=0, WRD CNT IS 65,536 DEC)
CS: .WORD 0 ;CONTROL SWITCH WORD
ST: .WORD 0 ;NUMBER OF UNIT STALLS
SM: .WORD 1000 ;MAX. STALLS, TEST 11

:DEFAULT OPERATING PARAMETER VALUES

PRDFLT:

3612
3613 005776 000000
3614 005776 000000
3615 006000 000000
3616 006002 000001
3617 006004 000000
3618 006006 000002
3619 006010 000001
3620 006012 000000
3621 006014 000023
3622 006016 000000
3623 006020 000025
3624 006022 000001
3625 006024 000000
3626 006026 063554
3627 006030 000000
3628 006032 000403
3629 006034 000000
3630 006036 000000
3631 006040 001000

.WORD 0 ;FC DEFAULT
.WORD 0 ;LC DEFAULT
.WORD 1 ;IC DEFAULT
.WORD 0 ;FT DEFAULT
.WORD 2 ;LT DEFAULT
.WORD 1 ;IT DEFAULT
.WORD 0 ;SO DEFAULT
.WORD 23 ;S1 DEFAULT
.WORD 0 ;S2 DEFAULT
.WORD 25 ;S3 DEFAULT
.WORD 1 ;IS DEFAULT
.WORD 0 ;PT DEFAULT
.WORD RWBUF ;LOW MA (0-15) DEFAULT
.WORD 0 ;HIGH MA (16-21) DEFAULT
.WORD 403 ;WC DEFAULT
.WORD 0 ;CS DEFAULT
.WORD 0 ;ST DEFAULT
.WORD 1000 ;SM DEFAULT

:OPERATING PARAMETER VALUE LOW AND HIGH LIMITS

PRMLIM:

3632
3633
3634
3635
3636 006042 000000
3637 006042 000000
3638 006044 000000
3639 006046 000000
3640 006050 000000
3641 006052 000001
3642 006054 000000
3643 006056 000000
3644 006060 000002
3645 006062 000000
3646 006064 000002
3647 006066 000001
3648 006070 000002
3649 006072 000000
3650 006074 000023
3651 006076 000000
3652 006100 000023
3653 006102 000000
3654 006104 000025
3655 006106 000000
3656 006110 000025
3657 006112 000001

.WORD 0 ;FC LIMITS
.WORD 0
.WORD 0 ;LC LIMITS
.WORD 0
.WORD 1 ;IC LIMITS
.WORD 0
.WORD 0 ;FT LIMITS
.WORD 2
.WORD 0 ;LT LIMITS
.WORD 2
.WORD 1 ;IT LIMITS
.WORD 2
.WORD 0 ;SO LIMITS
.WORD 23
.WORD 0 ;S1 LIMITS
.WORD 23
.WORD 0 ;S2 LIMITS
.WORD 25
.WORD 0 ;S3 LIMITS
.WORD 25
.WORD 1 ;IS LIMITS

3658	006114	000025	.WORD	25	
3659	006116	000000	.WORD	0	;PT LIMITS
3660	006120	177777	.WORD	177777	
3661	006122	063554	.WORD	RWBUF	;MA LOWER LIMIT
3662	006124	000000	.WORD	0	
3663	006126	157776	MAHILM: .WORD	157776	;MA UPPER LIMIT
3664	006130	000077	.WORD	77	
3665	006132	000000	.WORD	0	;WC LIMITS
3666	006134	177777	.WORD	177777	
3667	006136	000000	.WORD	0	;CS LIMITS
3668	006140	000070	.WORD	000070	
3669	006142	000000	.WORD	0	;ST LIMITS
3670	006144	177777	.WORD	177777	
3671	006146	000000	.WORD	0	;SM LIMITS
3672	006150	177777	.WORD	177777	

.ASCII PARAMETER MNEMONICS
PRMNEM:

3677	006152		.ASCII	/FC/	
3678	006152	041506	.ASCII	/LC/	
3679	006154	041514	.ASCII	/IC/	
3680	006156	041511	.ASCII	/FT/	
3681	006160	052106	.ASCII	/LT/	
3682	006162	052114	.ASCII	/IT/	
3683	006164	052111	.ASCII	/SO/	
3684	006166	030123	.ASCII	/S1/	
3685	006170	030523	.ASCII	/S2/	
3686	006172	031123	.ASCII	/S3/	
3687	006174	031523	.ASCII	/IS/	
3688	006176	051511	.ASCII	/PT/	
3689	006200	052120	.ASCII	/MA/	
3690	006202	040515	.WORD	0	;TABLE FILLER FOR MA
3691	006204	000000	.ASCII	/WC/	
3692	006206	041527	.ASCII	/CS/	
3693	006210	051503	.ASCII	/ST/	
3694	006212	052123	.ASCII	/SM/	
3695	006214	046523			

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

3701	006216		177777	
3702	006216	177777	177777	
3703	006220	177777	177777	
3704	006222	177777	177777	
3705	006224	052525	052525	
3706	006226	052525	052525	
3707	006230	052525	052525	
3708	006232	177777	177777	
3709	006234	177777	177777	
3710	006236	052525	052525	
3711	006240	052525	052525	
3712	006242	177777	177777	
3713	006244	052525	052525	

CZRM00 RK611/06 SS VERIF 1
CZRM00.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 75
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0353
SEQ 0074

3714	006246	177252	177252
3715	006250	177252	177252
3716	006252	172765	172765
3717	006254	172765	172765

3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769

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

006256	000000	000000
006256	000000	000000
006260	000000	000000
006262	000000	000000
006264	177777	177777
006266	177777	177777
006270	177777	177777
006272	000000	000000
006274	000000	000000
006276	177777	177777
006300	177777	177777
006302	000000	000000
006304	177777	177777
006306	000000	000000
006310	177777	177777
006312	000000	000000
006314	177777	177777

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

006316	052525	052525
006316	052525	052525
006320	052525	052525
006322	052525	052525
006324	125252	125252
006326	125252	125252
006330	125252	125252
006332	052525	052525
006334	052525	052525
006336	125252	125252
006340	125252	125252
006342	052525	052525
006344	125252	125252
006346	052525	052525
006350	125252	125252
006352	052525	052525
006354	125252	125252

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

006356	133333	133333
006356	066666	066666

K06

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 76
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0354
SEQ 0075

3770	006362	155555	155555
3771	006364	155555	155555
3772	006366	133333	133333
3773	006370	066666	066666
3774	006372	066666	066666
3775	006374	155555	155555
3776	006376	155555	155555
3777	006400	133333	133333
3778	006402	133333	133333
3779	006404	133333	133333
3780	006406	133333	133333
3781	006410	133333	133333
3782	006412	133333	133333
3783	006414	133333	133333
3784			
3785			
3786			
3787			
3788			
3789	006416		
3790	006416	121105	121105
3791	006420	150442	150442
3792	006422	064221	064221
3793	006424	132110	132110
3794	006426	055044	055044
3795	006430	026422	026422
3796	006432	013211	013211
3797	006434	105504	105504
3798	006436	042642	042642
3799	006440	021321	021321
3800	006442	110550	110550
3801	006444	044264	044264
3802	006446	022132	022132
3803	006450	011055	011055
3804	006452	104426	104426
3805	006454	042213	042213
3806			
3807			
3808			
3809			
3810			
3811	006456		
3812	006456	026455	026455
3813	006460	113226	113226
3814	006462	045513	045513
3815	006464	122645	122645
3816	006466	151322	151322
3817	006470	064551	064551
3818	006472	132264	132264
3819	006474	055132	055132
3820	006476	026455	026455
3821	006500	113226	113226
3822	006502	045513	045513
3823	006504	122645	122645
3824	006506	151322	151322
3825	006510	064551	064551

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

121105
150442
064221
132110
055044
026422
013211
105504
042642
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

L06

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A.1052) 10-JAN-78 12:16 PAGE 77
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0355
SEQ 0076

C
C

3826 006512 132264
3827 006514 055132

132264
055132

3828
3829

3830

3831

3832

3833

3834

3835

3836

3837

3838

3839

3840

3841

3842

3843

3844

3845

3846

3847

3848

3849

3850

3851

3852

3853

3854

3855

3856

3857

3858

3859

3860

3861

3862

3863

3864

3865

3866

3867

3868

3869

3870

3871

3872

3873

3874

3875

3876

3877

3878

3879

3880

3881

006516
006516 000000
006520 000000
006522 000000
006524 000000
006526 000000
006530 000000
006532 000000
006534 000000
006536 000000
006540 000000
006542 000000
006544 000000
006546 000000
006550 000000
006552 000000
006554 000000

;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:

006556
006556 177777
006560 177777
006562 177777
006564 177777
006566 177777
006570 177777
006572 177777
006574 177777
006576 177777
006600 177777
006602 177777
006604 177777
006606 177777
006610 177777
006612 177777
006614 177777

177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777
177777

;DATA PATTERN 08
SHIFTED 1 IN FIELD OF ZEROS
PAT08:

006616
006616 000001
006620 000002
006622 000004
006624 000010
006626 000020
006630 000040

000001
000002
000004
000010
000020
000040

M06

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 78
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0356
SEQ 0077

C
C.

3882	006632	000100	000100
3883	006634	000200	000200
3884	006636	000400	000400
3885	006640	001000	001000
3886	006642	002000	002000
3887	006644	004000	004000
3888	006646	010000	010000
3889	006650	020000	020000
3890	006652	040000	040000
3891	006654	100000	100000

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

3897	006656		
3898	006656	177776	177776
3899	006660	177775	177775
3900	006662	177773	177773
3901	006664	177767	177767
3902	006666	177757	177757
3903	006670	177737	177737
3904	006672	177677	177677
3905	006674	177577	177577
3906	006676	177377	177377
3907	006700	176777	176777
3908	006702	175777	175777
3909	006704	173777	173777
3910	006706	167777	167777
3911	006710	157777	157777
3912	006712	137777	137777
3913	006714	077777	077777

;DATA PATTERN 10
PAT10: ALTERNATING 0-1

3919	006716		
3920	006716	052525	052525
3921	006720	052525	052525
3922	006722	052525	052525
3923	006724	052525	052525
3924	006726	052525	052525
3925	006730	052525	052525
3926	006732	052525	052525
3927	006734	052525	052525
3928	006736	052525	052525
3929	006740	052525	052525
3930	006742	052525	052525
3931	006744	052525	052525
3932	006746	052525	052525
3933	006750	052525	052525
3934	006752	052525	052525
3935	006754	052525	052525
3936			
3937			

3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993006756
006756 125252
006760 125252
006762 125252
006764 125252
006766 125252
006770 125252
006772 125252
006774 125252
006776 125252
007000 125252
007002 125252
007004 125252
007006 125252
007010 125252
007012 125252
007014 125252;DATA PATTERN 11
; ALTERNATING 1-0
PAT11:125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252
125252;DATA PATTERN 12
; SHIFTING ZEROS AND ONES
PAT12:007016
007016 000001
007020 000003
007022 000007
007024 000017
007026 000037
007030 000077
007032 000177
007034 000377
007036 000777
007040 001777
007042 003777
007044 007777
007046 017777
007050 037777
007052 077777
007054 177777000001
000003
000007
000017
000037
000077
000177
000377
000777
001777
003777
007777
017777
037777
077777
177777;DATA PATTERN 13
; COMPOSITE ROTATING
PAT13:007056
007056 072307
007060 135143
007062 156461
007064 167230
007066 073514
007070 035646
007072 016723
007074 107351072307
135143
156461
167230
073514
035646
016723
107351

143564
061672
030735
114356
046167
123073
151453
164616

[illegible]

072307
135143
156461
167230
073514
035646
016723
107351
143564
061672
030735
114356
046167
123073
151453
164616

```
;WORD 00
;WORD 01
;WORD 02
;WORD 03
;WORD 04
;WORD 05
;WORD 06
;WORD 07
;WORD 10
;WORD 11
;WORD 12
;WORD 13
;WORD 14
;WORD 15
;WORD 16
;WORD 17
```

```
MINLO1=↑D24375
MINHI1=0
MAXLO1=↑D25625
```

```

;LO BITS OF SPEC'D MIN ROT. LATENCY
;HI BITS OF SPEC'D MIN ROT. LATENCY
;LO BITS OF SPEC'D MAX ROT. LATENCY

```

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 81
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0359
SEQ 0080

4050		000000			MAXHI1=0			; HI BITS OF SPEC'D MAX ROT. LATENCY
4051					; MAXLO2=↑D8000			; LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
4052					; MAXHI2=0			; HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
4053	007216	000000			MAXLO2: .WORD	0		; LOAD FROM NEW DRIVE ROUTINE
4054	007220	000000			MAXHI2: .WORD	0		; LOAD FROM NEWDRIVE ROUTINE
4055		112160			MAXLO3=↑D38000			; LO BITS OF SPEC'D MAX AVG SEEK TIME
4056		000000			MAXHI3=0			; HI BITS OF SPEC'D MAX AVG SEEK TIME
4057					; MAXLO4=↑D9464			; LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
4058					; MAXHI4=1			; HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
4059	007222	000000			MAXLO4: .WORD	0		; LOAD FROM THE DEWDRIE ROUTINE
4060	007224	000000			MAXHI4: .WORD	0		; LOAD FROM THE NEWDRIVE ROUTINE
4061		002734			RNDSDT=↑D1500			; SHORT RANDOM SEEKS = 1500(10)
4062		016514			RNDLNG=↑D7500			; LONG RANDOM SEEKS = 7500(10)
4063		000002			LSTTRK=2			; LAST TRACK = 2
4064		000365			ALNCYL=365			; ALIGNMENT CYLINDER =245(10)
4065		000002			BSERR=BIT1			; BSE ERROR
4066		000004			HVRCER=BIT2			; HVRC ERROR
4067		000010			OPIERR=BIT3			; OPI ERROR
4068		000020			DCKERR=BIT4			; DATA CHECK ERROR
4069		000040			ECCNC=BIT5			; ECC NON-CORRECTABLE
4070		000100			WCERR=BIT6			; WRITE CHECK ERROR
4071		000200			ABORT=BIT7			; ABORT
4072		000400			LEVZER=BIT8			; LEVEL TWO ERROR
4073		001000			BADSEC=BIT9			; BAD SECTOR FLAG
4074		002000			TWOTOS=BIT10			; TWO TIME OUTS
4075		004000			RCLREQ=BIT11			; RECALIBRATE REQUIRED
4076		010000			DRNAVL=BIT12			; DRIVE NOT RELSD BY OTHER PORT
4077		100000			ANYDER=BIT15			; ANY ERROR DETECTED FLAG
4078								
4079	007226	005015	055103	033122	DZR6M: .ASCIZ	<15><12>/CZR6MD0/		
4080	007234	042115	000060					
4081	007240	005015	044103	047101	XXDPMG: .ASCII	<CR><LF>/CHANGE XXDP PACK/		
4082	007246	042507	054040	042130				
4083	007254	020120	040520	045503				
4084	007262	005015	046103	040505	.ASCIZ	<CR><LF>/CLEAR LOC 40,RESTART PROGRAM/		
4085	007270	020122	047514	020103				
4086	007276	030064	051054	051505				
4087	007304	040524	052122	050040				
4088	007312	047522	051107	046501				
4089	007320	000						
4090	007321	040	020055	045522	SUBVER: .ASCIZ	& - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 3		
4091	007326	030466	027461	045522				
4092	007334	033060	051055	030113				
4093	007342	020067	052523	051502				
4094	007350	051531	042524	020115				
4095	007356	042526	044522	044506				
4096	007364	040503	044524	047117				
4097	007372	035040	050040	051101				
4098	007400	020124	000					
4099	007403	061	005015	000	PART1: .ASCIZ	/1/<15><12>		
4100	007407	062	005015	000	PART2: .ASCIZ	/2/<15><12>		
4101	007413	015	005012	040514	LSTMEM: .ASCIZ	<15><12><12>/LAST PHYS MEM ADR = /		
4102	007420	052123	050040	054510				
4103	007426	020123	042515	020115				
4104	007434	042101	020122	020075				
4105	007442	000						

007

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 82
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0360
SEQ 0081

4106	007443	117	042526	046122	OVLODR: .ASCIZ /OVERLAY LOADER ? (Y OR N) * /
4107	007450	054501	046040	040517	
4108	007456	042504	020122	020077	
4109	007464	054450	047440	020122	
4110	007472	024516	025040	000040	
4111	007500	005015	040520	040522	PRMINP: .ASCIZ <15><12>/PARAM INPUT MODE/<15><12><12>
4112	007506	020115	047111	052520	
4113	007514	020124	047515	042504	
4114	007522	005015	000012		
4115	007526	045522	033060	030055	RKBADR: .ASCIZ /RK06-07 BUS ADR = /
4116	007534	020067	052502	020123	
4117	007542	042101	020122	020075	
4118	007550	000			
4119	007551	122	030113	026466	RKVADR: .ASCIZ /RK06-07 VEC ADR = /
4120	007556	033460	053040	041505	
4121	007564	040440	051104	036440	
4122	007572	000040			
4123	007574	045522	033060	030055	RKPRTY: .ASCIZ /RK06-07 PRIOR =/
4124	007602	020067	051120	047511	
4125	007610	020122	000075		
4126	007614	053523	020122	020075	SWRMSG: .ASCIZ /SWR = /
4127	007622	000			
4128	007623	040	047040	053505	NEWMMSG: .ASCIZ / NEW = /
4129	007630	036440	000040		
4130	007634	005015	051104	024126	DRVSEQ: .ASCIZ <15><12>/DRV(S) = /
4131	007642	024523	036440	000040	
4132	007650	051104	053111	020105	BADDRV: .ASCIZ /DRIVE /
4133	007656	020040	000		
4134	007661	116	047117	042455	NXDRIV: .ASCIZ /NON-EXIST/<15><12>
4135	007666	044530	052123	005015	
4136	007674	000			
4137	007675	116	052117	051040	NTREDY: .ASCIZ /NOT RDY/<15><12>
4138	007702	054504	005015	000	
4139	007707	127	052122	046055	WRTLOK: .ASCIZ /WRT-LOCK/<15><12>
4140	007714	041517	006513	000012	
4141	007722	047514	042101	042105	ALNPAK: .ASCIZ /LOADED WITH ALIGN PACK/<15><12>
4142	007730	053440	052111	020110	
4143	007736	046101	043511	020116	
4144	007744	040520	045503	005015	
4145	007752	000			
4146	007753	111	020106	054130	REPLPK: .ASCIZ /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", & REPLACE IT : /
4147	007760	050104	050040	041501	
4148	007766	020113	047117	042040	
4149	007774	053122	030040	020054	
4150	010002	054524	042520	021040	
4151	010010	020131	041474	037122	
4152	010016	026042	023040	020040	
4153	010024	042522	046120	041501	
4154	010032	020105	052111	035040	
4155	010040	000040			
4156	010042	005015	025012	020052	NODRTS: .ASCII <15><12><12>/** NO DRVS TO TEST/<15><12>
4157	010050	047516	042040	053122	
4158	010056	020123	047524	052040	
4159	010064	051505	006524	012	
4160	010071	120	042522	051523	CNTRDY: .ASCIZ /PRESS "CONT" WHEN RDY/<15><12>
4161	010076	021040	047503	052116	

E07

CZR6MDO RK611/06 SS VERIF 1
CZR6MDO.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 83
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0361
SEQ 0082

4162	010104	020042	044127	047105	
4163	010112	051040	054504	005015	
4164	010120	000			
4165	010121	110	046101	020124	HLTRQD: .ASCIZ /HALT REQ/<15><12>
4166	010126	042522	006521	000012	
4167	010134	051104	020126	020060	DROXDP: .ASCIZ /DRV D IS LOAD MEDIUM/<15><12>
4168	010142	051511	046040	040517	
4169	010150	020104	042515	044504	
4170	010156	046525	005015	000	
4171	010163	015	005012	047524	ALDRVS: .ASCIZ <15><12><12>/TO TEST ALL DRVS TYPE "A" <CR>, ELSE <CR>/<15><12> /* /
4172	010170	052040	051505	020124	
4173	010176	046101	020114	051104	
4174	010204	051526	052040	050131	
4175	010212	020105	040442	020042	
4176	010220	041474	037122	020054	
4177	010226	046105	042523	036040	
4178	010234	051103	006476	025012	
4179	010242	000040			
4180	010244	005015	020114	020075	TSTMDS: .ASCII <15><12>/L = LIST TESTS/<15><12>
4181	010252	044514	052123	052040	
4182	010260	051505	051524	005015	
4183	010266	020103	020075	044103	.ASCII /C = CHANGE TESTS/<15><12>
4184	010274	047101	042507	052040	
4185	010302	051505	051524	005015	
4186	010310	020111	020075	047111	.ASCIZ /I = INPUT PARAMS, RUN TESTS/<15><12>
4187	010316	052520	020124	040520	
4188	010324	040522	051515	020054	
4189	010332	052522	020116	042524	
4190	010340	052123	006523	000012	
4191	010346	005015	047105	047524	ENTLCI: .ASCIZ <15><12>/ENTER L,C, OR I/<15><12> /* /
4192	010354	020122	026114	026103	
4193	010362	047440	020122	006511	
4194	010370	025012	000040		
4195	010374	047524	042040	043105	DFTEST: .ASCIZ /TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12> /* /
4196	010402	052501	052114	052040	
4197	010410	051505	051524	052040	
4198	010416	050131	020105	020104	
4199	010424	041474	037122	020054	
4200	010432	046105	042523	036040	
4201	010440	051103	006476	025012	
4202	010446	000040			
4203	010450	005015	042524	052123	TLSTHD: .ASCIZ <15><12>/TEST ITERATIONS/<15><12>
4204	010456	020040	020040	044440	
4205	010464	042524	040522	044524	
4206	010472	047117	006523	000012	
4207	010500	005015	020124	020075	EXPLAN: .ASCII <15><12>/T = TYPE PARAM LIST/<15><12>
4208	010506	054524	042520	050040	
4209	010514	051101	046501	046040	
4210	010522	051511	006524	012	
4211	010527	117	036440	047440	.ASCII /O = OPEN PARAM LIST/<15><12>
4212	010534	042520	020116	040520	
4213	010542	040522	020115	044514	
4214	010550	052123	005015		
4215	010554	020123	020075	042523	.ASCII /S = SET INDIVIDUAL PARAM/<15><12>
4216	010562	020124	047111	044504	
4217	010570	044526	052504	046101	

F07

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 84
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0362
SEQ 0083

4218	010576	050040	051101	046501	
4219	010604	005015			
4220	010606	020122	020075	052522	.ASCIZ /R = RUN TESTS/<15><12>
4221	010614	020116	042524	052123	
4222	010622	006523	000012		
4223	010626	005015	047105	042524	PARMDE: .ASCIZ <15><12>/ENTER T,O,S, OR R/<15><12>
4224	010634	020122	026124	026117	
4225	010642	026123	047440	020122	
4226	010650	006522	000012		
4227	010654	005015	020052	052504	DUACES: .ASCIZ <15><12>/* DUAL-PORT DATA TEST */<15><12>
4228	010662	046101	050055	051117	
4229	010670	020124	040504	040524	
4230	010676	052040	051505	020124	
4231	010704	006452	000012		
4232	010710	040515	020130	047527	MAWRDC: .ASCIZ /MAX WORD CT = /
4233	010716	042122	041440	020124	
4234	010724	020075	000		
4235	010727	052	020052	051440	SECNL1: .ASCII /** S0>S1/
4236	010734	037060	030523		
4237	010740	047040	052117	040440	NOTALD: .ASCIZ / NOT ALLOWED/<15><12>
4238	010746	046114	053517	042105	
4239	010754	005015	000		
4240	010757	052	020052	051440	SECNL2: .ASCIZ /** S2>S3/
4241	010764	037062	031523	000	
4242	010771	052	020052	043040	TRKNLW: .ASCIZ /** FT>LT/
4243	010776	037124	052114	000	
4244	011003	052	020052	053440	WC2BIG: .ASCIZ /** WC OR MA TOO LARGE/<15><12>
4245	011010	020103	051117	046440	
4246	011016	020101	047524	020117	
4247	011024	040514	043522	006505	
4248	011032	000012			
4249	011034	047524	042040	043105	DFQUES: .ASCIZ /TO DEFAULT ALL PARAMS. TYPE D <CR>, ELSE <CR>/<15><12>/* /
4250	011042	052501	052114	040440	
4251	011050	046114	050040	051101	
4252	011056	046501	026123	052040	
4253	011064	050131	020105	020104	
4254	011072	041474	037122	020054	
4255	011100	046105	042523	036040	
4256	011106	051103	006476	025012	
4257	011114	000040			
4258	011116	005015	051525	051105	PFIFTN: .ASCIZ <15><12>/USER-DEF PATT 15 :/<15><12>
4259	011124	042055	043105	050040	
4260	011132	052101	020124	032461	
4261	011140	035040	005015	000	
4262	011145	015	046412	042117	SELP15: .ASCIZ <15><12>/MOD PATT 15 :/<15><12>
4263	011152	050040	052101	020124	
4264	011160	032461	035040	005015	
4265	011166	000			
4266	011167	124	020117	047515	MODFY15: .ASCIZ /TO MOD PATT 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
4267	011174	020104	040520	052124	
4268	011202	030440	026065	052040	
4269	011210	050131	020105	020115	
4270	011216	041474	037122	020054	
4271	011224	046105	042523	036040	
4272	011232	051103	006476	025012	
4273	011240	000040			

G07

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 85
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0363
SEQ 0084

4274	011242	005015	047105	042524	ENTPAS: .ASCIZ <15><12>/ENTER NO. OF PASSES (1-77777) :/'15','12','* /
4275	011250	020122	047516	020056	
4276	011256	043117	050040	051501	
4277	011264	042523	020123	030450	
4278	011272	033455	033467	033467	
4279	011300	020051	006472	025012	
4280	011306	000040			
4281	011310	005015	025052	020040	DROPDR: .ASCIZ <15><12>/** DROPPING DRV - 20 ERRORS/'15'<12>
4282	011316	051104	050117	044520	
4283	011324	043116	042040	053122	
4284	011332	026440	031040	020060	
4285	011340	051105	047522	051522	
4286	011346	005015	000		
4287	011351	015	005012	042524	TSTDNR: .ASCII <15><12><12>/TESTING DRV /
4288	011356	052123	047111	020107	
4289	011364	051104	020126		
4290	011370	006440	000012		DRVNO: .ASCIZ //'15'<12>
4291	011374	051104	020126	000	DRIV: .ASCIZ /DRV /
4292	011401	103	051101	027124	CART: .ASCIZ /CART. /
4293	011406	000040			
4294	011410	042523	027122	047040	SERNM: .ASCIZ /SER. NO. /
4295	011416	027117	020040	000	
4296	011423	015	043012	041501	FACTBS: .ASCIZ <15><12>/FACTORY /
4297	011430	047524	054522	000040	
4298	011436	051412	043117	053524	SOFTBS: .ASCII <12>/SOFTWARE /
4299	011444	051101	020105		
4300	011450	040502	020104	042523	BDSECT: .ASCIZ /BAD SECTORS :/'15'<12>
4301	011456	052103	051117	020123	
4302	011464	006472	000012		
4303	011470	020040	047516	042516	NOFALS: .ASCIZ / NONE/
4304	011476	000			
4305					
4306					MESSAGES USED IN TIMING TESTS
4307	011477	052	020052	047040	NOCLKS: .ASCII /** NO CLOCK/
4308	011504	020117	046103	041517	
4309	011512	113			
4310	011513	040	020055	052040	TIMSKP: .ASCIZ / - TIMING TESTS WILL BE SKIPPED/'15'<12>
4311	011520	046511	047111	020107	
4312	011526	042524	052123	020123	
4313	011534	044527	046114	041040	
4314	011542	020105	045523	050111	
4315	011550	042520	006504	000012	
4316	011556	025052	020040	053513	CLKFAL: .ASCIZ /** KW11 FAILURE/
4317	011564	030461	043040	044501	
4318	011572	052514	042522	000	
4319	011577	015	051012	052117	ROTIMS: .ASCIZ <15><12>/ROT TIMES :/'15'<12>
4320	011604	052040	046511	051505	
4321	011612	035040	005015	000	
4322	011617	015	040412	043526	AVGSEK: .ASCIZ <15><12>/AVG SEEK TIMES :/'15'<12>
4323	011624	051440	042505	020113	
4324	011632	044524	042515	020123	
4325	011640	006472	000012		
4326	011644	005015	047117	020105	ONECYL: .ASCIZ <15><12>/ONE CYL SEEK TIMES :/'15'<12>
4327	011652	054503	020114	042523	
4328	011660	045505	052040	046511	
4329	011666	051505	035040	005015	

H07

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 86
PROGRAM SPECIFIC RESERVED LOCATIONSSEQ 0364
SEQ 0085

4330	011674	000			
4331	011675	015	046412	054101	MAXSEK: .ASCIZ <15><12>/MAX SEEK TIMES :/<15><12>
4332	011702	051440	042505	020113	
4333	011710	044524	042515	020123	
4334	011716	006472	000012		
4335	011722	025052	053506	020104	FORWRD: .ASCIZ /**FWD DIR**/<15><12>
4336	011730	044504	025122	006452	
4337	011736	000012			
4338	011740	025052	042522	020126	REVRSE: .ASCIZ /**REV DIR**/<15><12>
4339	011746	044504	025122	006452	
4340	011754	000012			
4341	011756	044515	020116	020075	MINEQ: .ASCIZ /MIN = /
4342	011764	000			
4343	011765	115	054101	036440	MAXEQ: .ASCIZ /MAX = /
4344	011772	000040			
4345	011774	053101	020107	020075	AVGEQ: .ASCIZ /AVG = /
4346	012002	000			
4347	012003	040	051525	000	MICROS: .ASCIZ / US/
4348	012007	040	043117	030440	BELOW: .ASCIZ / OF 128 BELOW SPEC'D MIN OF /
4349	012014	034062	041040	046105	
4350	012022	053517	051440	042520	
4351	012030	023503	020104	044515	
4352	012036	020116	043117	000040	
4353	012044	047440	020106	031061	ABOVE: .ASCIZ / OF 128 ABOVE SPEC'D MAX OF /
4354	012052	020070	041101	053117	
4355	012060	020105	050123	041505	
4356	012066	042047	046440	054101	
4357	012074	047440	020106	000	
4358	012101	040	043117	032040	ABOVE1: .ASCII / OF 410 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US/
4359	012106	030061	024040	030470	
4360	012114	020064	047506	020122	
4361	012122	045522	033460	020051	
4362	012130	041101	053117	020105	
4363	012136	050123	041505	042047	
4364	012144	046440	054101	047440	
4365	012152	020106	030070	030060	
4366	012160	052440	123		
4367	012163	050	032466	030060	.ASCIZ /((6500-RK07))/<CR><LF>
4368	012170	051055	030113	024467	
4369	012176	005015	000		
4370	012201	040	043117	030440	ABOVE3: .ASCII / OF 128 ABOVE SPEC'D MAX OF 75000 US/
4371	012206	034062	040440	047502	
4372	012214	042526	051440	042520	
4373	012222	023503	020104	040515	
4374	012230	020130	043117	033440	
4375	012236	030065	030060	052440	
4376	012244	123			
4377	012245	050	030467	030060	.ASCIZ /((71000-RK07))/<CR><LF>
4378	012252	026460	045522	033460	
4379	012260	006451	000012		
4380	012264	020040	051440	042520	SPCDMX: .ASCIZ / SPEC'D MAX IS 38000(36500-RK07) US/<15><12>
4381	012272	023503	020104	040515	
4382	012300	020130	051511	031440	
4383	012306	030070	030060	031450	
4384	012314	032466	030060	051055	
4385	012322	030113	024467	052440	

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 87
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0365
SEQ 0086

4386	012330	006523	000012			
4387	012334	032062	033463	020065	LIM1:	.ASCIZ /24375 US/<15><12>
4388	012342	051525	005015	000		
4389	012347	062	033065	032462	LIM2:	.ASCIZ /25625 US/<15><12>
4390	012354	052440	006523	000012		
4391						
4392	012362	025052	020040	040503	BAD632:	.ASCIZ /** CANNOT READ BAD SECTOR TRACK!/<15><12>
4393	012370	047116	052117	051040		
4394	012376	040505	020104	040502		
4395	012404	020104	042523	052103		
4396	012412	051117	052040	040522		
4397	012420	045503	006441	000012		
4398	012426	041536	005015	000	CNTRLC:	.ASCIZ /!C/<15><12>
4399	012433	136	006532	000012	CNTRLZ:	.ASCIZ /!Z/<15><12>
4400	012440	051136	005015	000	CNTRLR:	.ASCIZ /!R/<15><12>
4401	012445	136	006525	000012	CNTRLU:	.ASCIZ /!U/<15><12>
4402	012452	043536	005015	000	CNTRLG:	.ASCIZ /!G/<15><12>
4403	012457	015	005012	000	CR2LF:	.ASCIZ <15><12><12>
4404	012463	134	000		BKSLSH:	.ASCIZ /\
4405	012465	054	000		COMMA:	.ASCIZ /,
4406	012467	040	040		SPACE6:	.ASCII /\
4407	012471	040			SPACE4:	.ASCII /\
4408	012472	040			SPACE3:	.ASCII /\
4409	012473	040			SPACE2:	.ASCII /\
4410	012474	000040			SPACE1:	.ASCIZ /\
4411					.EVEN	
4412	012476	020040	000075		PRMBUF:	.ASCIZ / = /
4413	012502	047527	042122	000040	WORDSP:	.ASCIZ /WORD /
4414	012510	036440	000040		EQUALS:	.ASCIZ / = /
4415	012514	020052	000		PROMPT:	.ASCIZ /* /
4416	012517	076	000040		PRMPSP:	.ASCIZ /> /
4417						
4418						
4419						
4420						
4421						
4422						
4423	012522	105037	003116		DFSTRT:	CLRB MDFLAG ; SET FLAG FOR DEFAULT MODE
4424	012526	105037	003136		CLRB DULACS ; CLEAR DUAL-ACCESS FLAG	
4425	012532	105037	003126		CLRB ERRCNT ; CLEAR ERROR COUNT FOR RESTARTS	
4426	012536	022737	013732	000042	CMP #DRVTST, 2#42 ; SEE IF EOP RETURN ADRS = DRVTST	
4427	012544	001003			BNE 4\$; BR IF NOT DRVTST	
4428	012546	012737	017014	000042	MOV #NEWPAS, 2#42 ; SET RETURN ADRS = NEWPAS	
4429	012554	000414			BR CMSTRT ; PROCEED	
4430						
4431	012556	112737	000001	003136	DASTRT:	MOVB #1, DULACS ; SET FLAG FOR DUAL-ACCESS DATA TEST
4432	012564	112737	000001	003116	MOVB #1, MDFLAG ; SET FLAG FOR PARAMETER MODE	
4433	012572	000405			BR CMSTRT ; PROCEED	
4434						
4435	012574	112737	000001	003116	PSTART:	MOVB #1, MDFLAG ; SET FLAG FOR PARAMETER MODE
4436	012602	105037	003136		CLRB DULACS ; CLEAR DUAL-ACCESS TEST FLAG	
4437						
4438	012606	012737	000340	177776	CMSTRT:	MOV #PR7, 2#PS ; BLOCK ALL INTERRUPTS
4439					.SBTTL	INITIALIZE THE COMMON TAGS
4440					;;CLEAR	THE COMMON TAGS (\$CMTAG) AREA
4441	012614	012706	001100		MOV #CMTAG, R6 ; FIRST LOCATION TO BE CLEARED	

```

4447 012620 005026          CLR      (R6)+          ;; CLEAR MEMORY LOCATION
4448 012622 022706 001140    CMP      #SWR,R6 ;; DONE?
4449 012626 001374          BNE      -6          ;; LOOP BACK IF NO
4450 012630 012706 001100    MOV      #STACK,SP      ;; SETUP THE STACK POINTER
4451          ;; INITIALIZE A FEW VECTORS
4452 012634 012737 055154 000020 MOV      #SCOPE,2#IOTVEC      ;; IOT VECTOR FOR SCOPE ROUTINE
4453 012642 012737 000340 000022 MOV      #340,2#IOTVEC+2      ;; LEVEL 7
4454 012650 012737 054450 000030 MOV      #ERROR,2#EMTVEC      ;; EMT VECTOR FOR ERROR ROUTINE
4455 012656 012737 000340 000032 MOV      #340,2#EMTVEC+2      ;; LEVEL 7
4456 012664 012737 056262 000034 MOV      #TRAP,2#TRAPVEC      ;; TRAP VECTOR FOR TRAP CALLS
4457 012672 012737 000340 000036 MOV      #340,2#TRAPVEC+2      ;; LEVEL 7
4458 012700 012737 055020 000024 MOV      #SPWRDN,2#PWRVEC      ;; POWER FAILURE VECTOR
4459 012706 012737 000340 000026 MOV      #340,2#PWRVEC+2      ;; LEVEL 7
4460 012714 013737 025502 025474 MOV      #ENDCT,SEOPCT      ;; SETUP END-OF-PROGRAM COUNTER
4461 012722 005037 001304          CLR      $TIMES      ;; INITIALIZE NUMBER OF ITERATIONS
4462 012726 005037 001306          CLR      $ESCAPE      ;; CLEAR THE ESCAPE ON ERROR ADDRESS
4463 012732 112737 000001 001115 MOVB     #1,$SERMAX      ;; ALLOW ONE ERROR PER TEST
4464 012740 012737 012740 001106 MOV      #,$SLPADR      ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
4465 012746 012737 012746 001110 MOV      #,$SLPERR      ;; SETUP THE ERROR LOOP ADDRESS
4466          ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
4467          ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
4468 012754 013746 000004          MOV      2#ERRVEC, -(SP)      ;; SAVE ERROR VECTOR
4469 012760 012737 013014 000004 MOV      #64,$2#ERRVEC      ;; SET UP ERROR VECTOR
4470 012766 012737 177570 001140 MOV      #DSWA,SWR      ;; SETUP FOR A HARDWARE SWICH REGISTER
4471 012774 012737 177570 001142 MOV      #DDISP,DISPLAY      ;; AND A HARDWARE DISPLAY REGISTER
4472 013002 022777 177777 166130 CMP      #-1,$SWR      ;; TRY TO REFERENCE HARDWARE SWR
4473 013010 001012          BNE      66$      ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
4474          ;; AND THE HARDWARE SWR IS NOT = -1
4475 013012 000403          BR      65$      ;; BRANCH IF NO TIMEOUT
4476 013014 012716 013022 64$: MOV      #65$, (SP)      ;; SET UP FOR TRAP RETURN
4477 013020 000002          RTI
4478 013022 012737 000176 001140 65$: MOV      #SWREG,SWR      ;; POINT TO SOFTWARE SWR
4479 013030 012737 000174 001142 MOV      #DISPREG,DISPLAY
4480 013036 012637 000004 66$: MOV      (SP)+,2#ERRVEC      ;; RESTORE ERROR VECTOR
4481          ;; CLEAR PASS COUNT
4482 013042 005037 001326          CLR      $PASS
4483 013046 132737 000200 001341 BITB     #APTSIZE,$ENVM      ;; TEST USER SIZE UNDER APT
4484 013054 001403          BEQ      67$      ;; YES, USE NON-APT SWITCH
4485 013056 012737 001342 001140 MOV      #SSWREG,SWR      ;; NO, USE APT SWITCH REGISTER
4486          ;; CLEAR THE UNIBUS
4487 013064 000005 67$: RESET
4488 013066 012737 000006 000004 MOV      #6,2#ERRVEC      ;; SET TIME-OUT VECTOR
4489 013074 005037 000006          CLR      2#ERRVEC+2
4490 013100 112737 000144 001115 MOVB     #100,$SERMAX      ;; SET MAX ERROR CNT TO 100 FOR $SCOPE
4491 013106 105037 003130          CLRB     DRVERS      ;; CLEAR ERROR COUNT FOR CURRENT DRIVE
4492 013112 112737 000001 003146 MOVB     #1,HLPVOL      ;; SET HLP FILE OVLD INDICTR
4493 013120 104401 007226          TYPE     ,DZR6M      ;; TYPE PROGRAM I.D. FOR PART 1
4494 013124 104401 007321          TYPE     ,SUBVER
4495 013130 104401 007403          TYPE     ,PART1
4496 013134 105737 003147          TSTB     $NOTYPE      ;; SEE IF OPERATOR NOTE SHOULD BE TYPED
4497 013140 001004          BNE      39$      ;; BR IF NOT
4498 013142 104401 063554          TYPE     ,NOTMSG      ;; TYPE OPERATOR NOTE
4499 013146 105237 003147          INCB     $NOTYPE      ;; SET FLAG FOR NEXT TIME
4500 013152 105737 003136 39$: TSTB     $DULACS      ;; SEE IF DUAL-ACCESS DATA TEST
4501 013156 001402          BEQ      40$      ;; BR IF NOT
4502 013160 104401 010654          TYPE     ,DUACES      ;; TYPE "* DUAL-ACCESS DATA TEST *"

```

K07

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 89
INITIALIZE THE COMMON TAGSSEQ 0367
SEQ 0088

4498	013164	012737	025700	000060	40\$:	MOV	#KBDHDL, @TKVEC	; LOAD VECTOR FOR TTY KBD
4499	013172	012737	000200	000062		MOV	#PR4, @TKVEC+2	; SET KBD PRIORITY = 4
4500	013200	013701	003040			MOV	RKVEC, R1	; ADDR. OF RK06 VECTOR STORAGE
4501	013204	012721	045270			MOV	#I, INTR, (R1)+	; SET IT TO RK06 HANDLER
4502	013210	013711	003042			MOV	RKPRI, (R1)	; SET RK06 PRIORITY
4503	013214	012737	026510	000250		MOV	#KTERAD, @MMVEC	; VECT FOR KT11 FAILURE
4504	013222	012737	000340	000252		MOV	#PR7, @MMVEC+2	; SET PRIOR. = 7 FOR HANDLER
4505	013230	105037	003120			CLRB	TSTING	; CLEAR "RUNNING TESTS" FLAG
4506	013234	012737	000000	177776		MOV	#PRO, @PS	; ALLOW ALL INTERRUPTS AGAIN
4507	013242	012701	005232			MOV	#PRVCMO, R1	; ZERO OUT PREVIOUS COMMAND
4508	013246	012700	000006			MOV	#6, RO	; SET COUNTER
4509	013252	005021			42\$:	CLR	(R1)+	; ZERO A WORD
4510	013254	005300				DEC	RO	
4511	013256	001375				BNE	42\$	; BR IF NOT DONE YET
4512	013260	005037	005512			CLR	STALLS	; DON'T ALLOW STALLS YET
4513	013264	023727	000042	017014		CMP	@42, #NEWPAS	; SEE IF CHAIN MODE
4514	013272	101002				BHI	44\$	; BR IF YES
4515	013274	004737	026370			JSR	PC, GTSWRG	; OPEN SOFTWARE SWR FOR MODIFICATION
4516	013300	012737	176543	052716	44\$:	MOV	#176543, \$HINUM	; INIT. PSEUDO-RANDOM NOS.
4517	013306	012737	123456	052720		MOV	#123456, \$LONUM	
4518							; CHECK FOR PRESENCE OF KW11-L OR	P CLOCK, AND SET FLAGS
4519	013314	105037	003132			CLRB	PCLKF	; INIT. P CLOCK FLAG
4520	013320	105037	003133			CLRB	DOTIM	; INIT. TIMING TESTS FLAG
4521	013324	012737	013364	000004		MOV	#6\$ @4	; SET TIME-OUT ERROR VECTOR
4522	013332	005777	167614			TST	@PK\$	; SEE IF P-CLOCK IS PRESENT
4523	013336	105237	003132			INCB	PCLKF	; PRESENT, SET FLAG
4524	013342	013700	003162			MOV	PCVEC, RO	; LOAD KW11-P VECTOR ADDRESS
4525	013346	012720	032664		4\$:	MOV	#CLOCK, (RO)+	; ADDR OF CLOCK SERVICE ROUTINE
4526	013352	012710	000340			MOV	#PR7, (RO)	; SET CLOCK HANDLER PRIORITY = 7
4527	013356	105237	003133			INCB	DOTIM	; SET FLAG TO ALLOW TIMING TESTS
4528	013362	000414				BR	8\$	; BR TO CONTINUE
4529	013364	022626			6\$:	CMP	(SP)+, (SP)+	; P-CLK NOT THERE, RESET THE STACK
4530	013366	012737	013406	000004		MOV	#7\$ @4	; SET TIME-OUT ERROR VECTOR
4531	013374	005777	167550			TST	@LK\$	; SEE IF L-CLK PRESENT
4532	013400	013700	003160			MOV	LCVEC, RO	; LOAD KW11-L VECTOR ADDRESS
4533	013404	000760				BR	4\$	; BR TO SET UP L-CLK VECTOR
4534	013406	022626			7\$:		(SP)+, (SP)+	; L-CLK NOT THERE, RESET THE STACK
4535	013410	104401	011477				#NOCLK\$	; SAY TIMING TESTS WON'T BE RUN
4536	013414	012737	000006	000004	8\$:	MOV	#6, @4	; SET TIME-OUT VECTOR TO LOCATION 6
4537	013422	004737	026072			JSR	PC, SIZMEM	; SIZE MEMORY, FIX MA LIMIT IN PRMLIM
4538	013426	105737	003135			TSTB	XDPSVD	; SEE IF XXDP PREVIOUSLY SAVED
4539	013432	001404				BEQ	9\$	; BR IF NOT
4540	013434	004737	027262			JSR	PC, GETXDP	; RESTORE SAVED XXDP
4541	013440	105037	003135			CLRB	XDPSVD	; CLEAR THE FLAG
4542	013444				9\$:			
4543	013444	122737	000013	000041		CMPB	#13, @41	; LOAD FROM XXDP ?
4544	013452	001003				BNE	99\$	; BRANCH IF NOT
4545	013454	104401	007240			TYPE	, XXDPMG	
4546	013460	000000				HALT		
4547	013462				99\$:			
4548	013462	105737	003116			TSTB	MDFLAG	; SEE IF DEFAULT MODE
4549	013466	001031				BNE	10\$	; BR IF NOT DEFAULT MODE
4550	013470	013700	001370			MOV	\$VECT1, RO	; GET APT RK06 VECTOR AND PRY
4551	013474	013737	001374	003036		MOV	\$BASE, RKBAS	; GET APT RK06 BASE ADDRESS
4552	013502	132737	000200	001341		BITB	#BIT7, \$ENVM	; SEE IF NO SIZING
4553	013510	001005				BNE	18\$	; BR IF NO SIZING

```

4554 013512 012700 120210      MOV      #AVECT1,RO      ;GET DEFAULT VECTOR AND PRIORITY
4555 013516 012737 177440      MOV      #ABASE,RKBAS      ;GET DEFAULT BASE ADDRESS
4556 013524 110037 003040      MOV      RO,RKVEC      ;STORE VECTOR
4557 013530 105037 003041      CLRB      RKVEC+1      ;CLEAR HI BYTE
4558 013534 000300      SWAB      RO      ;GET PRTY INTO BITS 5-7
4559 013536 042700 177437      BIC      #177437,RO      ;CLEAR OTHER BITS
4560 013542 010037 003042      MOV      RO,RKPRI      ;STORE RK06 PRIORITY
4561 013546 000137 014444      JMP      ALLDRV      ;GO CHECK ALL DRIVES
4562
4563
4564
4565      ;BEGIN PARAMETER INPUT MODE
4566 013552 104401 007500      10$:      TYPE      ,PRMINP      ;TYPE "PARAMETER INPUT MODE"
4567
4568      ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
4569      MOV      RKBAS,-(SP)      ;PUT OLD VALUE ON STACK
4570 013562 104401 007526      TYPE      RKBADR      ;TYPE "RK06 BUS ADR = "
4571 013566 004737 026260      JSR      PC,GETPRM      ;TYPE OLD, GET NEW RKBAS VALUE
4572 013572 012637 003036      MOV      (SP)+,RKBAS      ;STORE NEW VALUE
4573
4574      ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
4575      MOV      RKVEC,-(SP)      ;PUT OLD VALUE ON STACK
4576 013602 104401 007551      TYPE      RKVADR      ;TYPE "RK06 VEC ADR = "
4577 013606 004737 026260      JSR      PC,GETPRM      ;TYPE OLD, GET NEW RKVEC VALUE
4578      MOV      (SP)+,RKVEC      ;STORE NEW VALUE
4579      ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
4580 013616 013746 003042      11$:      MOV      RKPRI,-(SP)      ;GET OLD VALUE OF PRIORITY
4581      ASL      (SP)      ;GET IT INTO BITS 0-2
4582      ASL      (SP)
4583      ASL      (SP)
4584      SWAB      (SP)
4585 013632 104401 007574      TYPE      RKPRTY      ;TYPE "RK06 PRIORITY = "
4586 013636 004737 026260      JSR      PC,GETPRM      ;TYPE OLD, GET NEW RKPRI VALUE
4587      MOV      (SP)+,RO
4588 013642 012600 000004      CMP      RO,#4      ;SEE IF AT LEAST LEVEL 4
4589 013644 020027 000007      BLT      12$      ;BR IF NEW VALUE TOO SMALL
4590      CMP      RO,#7      ;SEE IF LEVEL 7 OR LESS
4591      BGT      12$      ;BR IF NEW VALUE TOO LARGE
4592      SWAB      RO      ;GET PRIORITY INTO BITS 5-7
4593      ASR      RO
4594      ASR      RO
4595      ASR      RO
4596 013670 010037 003042      MOV      RO,RKPRI      ;STORE NEW VALUE
4597 013674 104401 001315      TYPE      $CRLF
4598 013700 000405      BR      16$
4599 013702 104401 005272      12$:      TYPE      ,BUFFO      ;ECHO BAD INPUT
4600 013706 104401 001314      TYPE      $QUES
4601      BR      11$      ;GO ASK AGAIN
4602 013714 105037 003126      16$:      CLRB      ERRCNT      ;CLEAR ERROR CNT FOR RESTARTS
4603 013720 012737 013732 000042      MOV      #DRVST,2#42      ;SET UP DUMP MODE RETURN FROM $EOP
4604      ;UPON COMPLETION OF REQUESTED PASSES
4605 013726 000137 014522      JMP      CHKLST      ;GO CHECK STATUS OF MARKED DRIVES
4606
4607
4608
4609      ;*****

```

4610
4611
4612
4613
4614
4615
4616
4617
4618
4619 013732
4620 013732 012706 001100
4621 013736 005037 005512
4622 013742 104401 010163
4623 013746 004737 030624
4624 013752 013732
4625 013754 013732
4626 013756 013732
4627 013760 005700
4628 013762 001413
4629 013764 022737 000101 005272
4630 013772 001002
4631 013774 000137 014444
4632 014000 104401 005272
4633 014004 104401 001314
4634 014010 000750
4635
4636 014012 104401 007634
4637 014016 005001
4638 014020 005000
4639 014022 012703 000001
4640 014026 105761 005606
4641 014032 001414
4642 014034 005700
4643 014036 001402
4644 014040 104401 012465
4645 014044 010137 005542
4646 014050 052737 000060 005542
4647 014056 104401 005542
4648 014062 005200
4649 014064 005201
4650 014066 006303
4651 014070 022701 000010
4652 014074 001354
4653 014076 005700
4654 014100 001016
4655 014102 104401 011470
4656 014106 104401 010042
4657
4658 014112 012703 000401
4659 014116 012701 005606
4660 014122 010321
4661 014124 010321
4662 014126 010321
4663 014130 010311
4664 014132 000137 044134
4665 014136 104401 001315

SBTTL TO - DESIRED DRIVE INPUT ROUTINE
;THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
;WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
;CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
;DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS.
;AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
;LIST (DRVLST).
;*****
DRVTST:
MOV #STACK, SP ;RESET THE STACK
CLR STALLS ;INHIBIT STALL BETWEEN OPERATIONS
TYPE ALDRVS ;ASK IF ALL DRIVES DESIRED
JSR PC, RDCHRS ;READ RESPONSE
DRVTST ;(IC) RETURN ADDRESS
DRVTST ;(IZ) RETURN ADDRESS
DRVTST ;(IU) OR ERROR RETURN ADDRESS
TST R0 ;SEE IF NULL INPUT
BEQ TELDRV ;BR IF NULL INPUT
CMP #A, BUFFO ;SEE IF ALL DRIVES REQUESTED
BNE 4\$;BR IF NOT ALL DRIVES
JMP ALLDRV ;CHECK ALL DRIVES
4\$: TYPE ,BUFFO ;ECHO BAD INPUT
TYPE ,SQUES ;TYPE <?> AND <CR>, <LF>
BR DRVTST ;GO ASK AGAIN
;TYPE CURRENT DRIVE LIST
TELDRV: TYPE ,DRVSEQ ;TYPE "DRIVE(S) = "
CLR R1 ;INITIALIZE DRIVE NUMBER
CLR R0 ;INIT. COUNT OF LISTED DRIVES
MOV #BITO, R3 ;INIT BIT POINTER
4\$: TSTB DRVLST(R1) ;SEE IF THIS DRIVE IS LISTED
BEQ 8\$;BR IF DRIVE NOT LISTED
TST R0 ;SEE IF FIRST TIME HERE
BEQ 6\$;BR IF FIRST TIME HERE
6\$: TYPE ,COMMA ;TYPE A COMMA
MOV R1, SCRACH ;USE SCRACH FOR BUFFER
BIS #0, SCRACH ;CONVERT DRIVE NO. TO ASCII
TYPE ,SCRACH ;TYPE DRIVE NO.
8\$: INC R0 ;INCREMENT NO. OF LISTED DRIVES
INC R1 ;INCREMENT DRIVE NO.
ASL R3 ;SHIFT BIT POINTER
CMP #10, R1 ;SEE IF DONE YET
BNE 4\$;BR IF NOT DONE
TST R0 ;SEE IF ANY DRIVES WERE LISTED
BNE 26\$;BR IF YES
TYPE ,NOFALS ;TYPE " NONE"
TYPE ,NODRTS ;TYPE "** NO DRIVES TO TEST"
; "PRESS 'CONT' WHEN RDY"
;PATTERN TO MARK DRIVES
;DRIVE LIST ADDRESS
;MARK ALL DRIVES IN LIST
MOV #401, R3
MOV #DRVLST, R1
MOV R3, (R1)+
MOV R3, (R1)+
MOV R3, (R1)+
MOV R3, (R1)+
JMP HLTPRG ;HALT PROGRAM
26\$: TYPE ,SCRLF ;TYPE <CR>, <LF>

NO7

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 92
TO - DESIRED DRIVE INPUT ROUTINESEQ 0370
SEQ 0091C
C

4666	014142	105737	003116		TSTB	MDFLAG	;SEE IF DEFAULT MODE
4667	014146	001002			BNE	20\$	;BR IF NOT DEFAULT MODE
4668	014150	000137	015062		JMP	LODFLS	;GO LOAD DEFLT ITER. COUNTS
4669	014154	122737	000013	000041	CMPB	#13,2#41	;SEE IF RK06 IS XXDP MEDIUM
4670	014162	001032			BNE	19\$	;BR IF NOT
4671	014164	105737	005606		TSTB	DPVLST	;SEE IF DRIVE 0 LISTED TO TEST
4672	014170	001427			BEQ	19\$	;BR IF NOT
4673	014172	104401	007753	24\$:	TYPE	,REPLPK	;TYPE "IF XXDP PACK ON DRV 0, TYPE Y <CR>
4674							AND REPLACE IT"
4675	014176	004737	030624		JSR	PC, RDCHRS	;READ RESPONSE
4676	014202	014172			24\$		; (↑C) RETURN
4677	014204	014172			24\$		; (↑Z) RETURN
4678	014206	014172			24\$		; (↑U) RETURN
4679	014210	005700			TST	RO	;SEE IF NULL INPUT
4680	014212	001416			BEQ	19\$	;BR IF JUST <CR> TYPED
4681	014214	022737	000131	005272	CMP	#'Y,BUFFO	;SEE IF "Y <CR>" TYPED
4682	014222	001363			BNE	24\$	;BR IF NOT TO ASK AGAIN
4683	014224	104401	010071		TYPE	,CNTRDY	;TYPE "PRESS CONT WHEN DRV RDY"
4684	014230	042762	000100	000000	BIC	#IE,RKCS1(R2)	;DISABLE RK06 INTERRUPT
4685	014236	000000			HALT		;HALT FOR PACK CHANGE
4686	014240	004737	027522		JSR	PC,INITSS	;INIT THE SUB-SYS
4687	014244	000137	014522		JMP	CHKLST	;GO CHECK STATUS OF LISTED DRIVES
4688	014250	104401	012514	19\$:	TYPE	,PROMPT	;TYPE ASTERISK AND SPACE
4689					;READ AND CHECK	NEW DRIVE NUMBERS	
4690	014254	004737	030624		JSR	PC, RDCHRS	;READ IN INPUT STRING
4691	014260	013732			DRVTST		; (↑C) RETURN ADDRESS
4692	014262	013732			DRVTST		; (↑Z) RETURN ADDRESS
4693	014264	014012			TELDV		; (↑U) OR ERROR RETURN ADDRESS
4694	014266	005700			TST	RO	;SEE IF NULL INPUT
4695	014270	001007			BNE	9\$	;BR IF NEW DRIVES WILL BE SELECTED
4696	014272	105737	003136		TSTB	DULACS	;SEE IF DUAL-ACCESS FLAG SET
4697	014276	001002			BNE	22\$	;BR IF YES
4698	014300	000137	014574		JMP	ASKTST	;JUMP TO THE TEST INPUT ROUTINE
4699	014304	000137	015340	22\$:	JMP	INPUTP	;GO ASK FOR INPUT PARAMETERS
4700	014310	022700	000017	9\$:	CMP	#15.,RO	;SEE IF TOO MANY CHARACTERS TYPED
4701	014314	002005			BGE	12\$	;BR IF NOT TOO MANY
4702	014316	104401	005272	10\$:	TYPE	,BUFFO	;ECHO BAD INPUT
4703	014322	104401	001314		TYPE	,SQUES	;TYPE <?> AND <CR> <LF>
4704	014326	000631			BR	TELDV	;GO TYPE DRIVES AND ASK AGAIN
4705	014330	005001		12\$:	CLR	R1	;CLEAR CHAR. POINTER
4706	014332	126127	005272	000060	14\$:	CMPB	BUFFO(R1),#'0
4707	014340	002766			BLT	10\$	;SEE IF DRIVE NO. < 0
4708	014342	126127	005272	000067	CMPB	BUFFO(R1),#'7	;BR TO ECHO BAD INPUT
4709	014350	003362			BGT	10\$	;SEE IF > 7
4710	014352	005201			INC	R1	;BR TO ECHO BAD INPUT
4711	014354	020100			CMP	R1,RO	;INCREMENT CHAR POINTER
4712	014356	001406			BEQ	16\$	;SEE IF MORE CHARS TO CHECK
4713	014360	126127	005272	000054	CMPB	BUFFO(R1),#',	;BR IF ALL CHARS CHECKED
4714	014366	001353			BNE	10\$	;SEE IF THIS IS COMMA
4715	014370	005201			INC	R1	;BR IF NOT COMMA
4716	014372	000757			BR	14\$	;INCREMENT CHAR POINTER
4717					;GET NEW DRIVE NUMBERS INTO LIST		;BR TO CONTINUE CHECKING CHARS
4718	014374	012701	005606	16\$:	MOV	#DRVLST,R1	;GET DRIVE LIST ADDRESS
4719	014400	005021			CLR	(R1)+	;CLEAR OUT THE DRIVE LIST
4720	014402	005021			CLR	(R1)+	
4721	014404	005021			CLR	(R1)+	

```

4722 014406 005011          CLR      (R1)
4723 014410 005000          CLR      RO
4724 014412 116001 005272 18$:  MOVB   BUFFO(RO),R1  ;CLEAR BUFFER POINTER
4725 014416 042701 000060      BIC      #'0,R1      ;GET A DRIVE NUMBER
4726 014422 112761 000001 005606  MOVB   #1,DRVLST(R1)  ;STRIP ASCII BITS
4727 014430 062700 000002      ADD      #2,RO      ;MARK THIS DRIVE IN DRIVE LIST
4728 014434 105760 005271      TSTB   BUFFO-1(RO)  ;POINT TO NEXT DRIVE NUMBER
4729 014440 001364          BNE      18$      ;SEE IF NULL CHAR YET
4730 014442 000427          BR       CHKLST    ;BR IF NOT DONE YET
4731          ;COME HERE IF ALL DRIVES REQUESTED
4732 014444 005000          ALLDRV: CLR      RO      ;GO CHECK NEW DRIVES FOR VALID STATUS
4733 014446 013703 001376      MOV      $DEVN,R3    ;CLEAR DRIVE NUMBER
4734 014452 132737 000200 001341  BITB   #BIT7,$ENVN  ;GET APT DEVICE MAP
4735 014460 001002          BNE      1$      ;SEE IF NO SIZING
4736 014462 012703 000377      MOV      #377,R3    ;BR IF NO SIZING
4737 014466 012701 000001      MOV      #BIT0,R1    ;SET UP FOR SIZING
4738 014472 105060 005606 1$:  CLRB   DRVLST(RO)  ;SET BIT POINTER
4739 014476 030103          BIT      R1,R3      ;INITIALIZE DRIVE ENTRY
4740 014500 001403          BEQ      4$      ;SEE IF THIS DRIVE IS REQUESTED
4741 014502 112760 000001 005606  MOVB   #1,DRVLST(RO) ;BR IF NOT
4742 014510 006301          ASL      R1      ;MARK THIS DRIVE IN DRIVE LIST
4743 014512 005200          INC      RO      ;SHIFT BIT POINTER
4744 014514 022700 000010      CMP      #10,RO    ;INCREMENT DRIVE NUMBER
4745 014520 001364          BNE      2$      ;SEE IF DONE MARKING ALL DRIVES
4746          ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
4747 014522 005000          CHKLST: CLR      RO      ;CLEAR DRIVE NUMBER
4748 014524 105760 005606 2$:  TSTB   DRVLST(RO)  ;SEE IF THIS DRIVE IS MARKED IN LIST
4749 014530 001410          BEQ      4$      ;BR IF NOT MARKED
4750 014532 010037 005510      MOV      RO,DRIVE  ;SET DRIVE NUMBER FOR SCNDRV
4751 014536 004737 027650      JSR      PC,SCNDRV ;CHECK STATUS OF THIS DRIVE
4752          ;IF NOT VALID, TYPE MESSAGE
4753          ;AND REMOVE IT FROM DRIVE LIST
4754          ;ERROR RETURN ADDRESS FOR SCNDRV
4755 014542 014566          6$:  MOVB   TYPFMT,DTYLIST(RO) ;SET DRV TYPE. 0=RK06, 4=RK07
4756 014544 113760 003124 005616 4$:  INC      RO      ;RETURN HERE IF DRIVE IS USEABLE
4757 014552 005200          CMP      #10,RO    ;SEE IF DONE CHECKING LIST
4758 014554 022700 000010      BNE      2$      ;BR IF NOT DONE YET
4759 014562 000137 014012      JMP      TELDRV  ;GO BACK TO LIST SELECTED DRIVES
4760 014566 105060 005606 6$:  CLRB   DRVLST(RO)  ;REMOVE INVALID DRIVE FROM LIST
4761 014572 000767          BR       4$      ;CONTINUE CHECKING THE LIST

```

```

4762
4763
4764 *****
4765 ;SBTTL TO - DESIRED TEST INPUT ROUTINE
4766 ;THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
4767 ;COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
4768 ;AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
4769 ;TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
4770 ;THAT THE TEST IS NOT TO BE RUN.
4771 ;THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
4772 ;CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
4773 ;TYPED. (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
4774 ;AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
4775 ;OF OPERATING PARAMETERS, AND RUN TESTS.
4776 ;TYPING (↑) CAUSES IMMEDIATE RETURN TO DRVST. TYPING
4777 ;* (↑Z) CAUSES RETURN TO ASKTMD.

```

```

4778                                     ;*****
4779                                     ;
4780 : DETERMINE L,C, OR I MODE
4781 ASKTST: TYPE ,TSTMD5                : ASK FOR TEST INPUT MODE
4782 ASKTMD: TYPE ,ENTLCI                : TYPE "ENTER L,C, OR I"
4783 JSR PC,ROCHRS                      : READ RESPONSE
4784 DRVTST                               : (↑C) RETURN ADDRESS
4785 ASKTMD                               : (↑Z) RETURN ADDRESS
4786 ASKTMD                               : (↑U) OR ERROR RETURN ADDRESS
4787 TST RO                             : SEE IF ANY INPUT
4788 BNE 4$                             : BR IF ANY INPUT
4789 2$: TYPE ,BUFFO                    : ECHO BAD INPUT
4790 TYPE ,SQUES                       : TYPE "<?> AND <CR>,<LF>"
4791 BR ASKTMD                          : GO ASK AGAIN
4792 4$: CMP #L,BUFFO                   : SEE IF (L) TYPED
4793 BNE 6$                             : BR IF NOT (L)
4794 JMP LSTTST                         : JUMP TO LIST TESTS
4795 6$: CMP #C,BUFFO                   : SEE IF (C) TYPED
4796 BNE 8$                             : BR IF NOT (C)
4797 JMP CHGTST                         : JUMP TO CHANGE TESTS
4798 8$: CMP #I,BUFFO                   : SEE IF (I) TYPED
4799 BNE 2$                             : BR IF NOT (I), TO ECHO BAD INPUT
4800 JMP INPUTP                         : GO INPUT PARAMS AND RUN TESTS
4801
4802 : LIST CURRENT TESTS AND ITERATION COUNTS
4803 LSTTST: TYPE TLSTHD                : TYPE HEADING "TEST ITERATIONS"
4804 CLR R1                             : INITIALIZE TEST INDEX
4805 2$: JSR PC,PREPKB                   : PREPARE FOR POSSIBLE KBD INPUT
4806 JSR PC,TYPTST                      : TYPE CURRENT TEST AND ITERATION NUMBER
4807 TYPE ,SCLF                         : TYPE "<CR>,<LF>"
4808 TST INTCHR                         : SEE IF ANY INPUT
4809 BEQ 8$                             : BR IF NO INPUT
4810 4$: CMPB #003,INTCHR                : SEE IF (↑C) TYPED
4811 BNE 4$                             : BR IF NOT (↑C)
4812 JMP DRVTST                         : JUMP TO ASK FOR DRIVES AGAIN
4813 4$: CMPB #032,INTCHR                : SEE IF (↑Z) TYPED
4814 BNE 6$                             : BR IF NOT (↑Z)
4815 JMP ASKTMD                         : JUMP TO ASK FOR NEW TEST INPUT MODE
4816 6$: JSR PC,ECOBAD                   : ECHO BAD INPUT
4817 8$: ADD #2,R1                       : INCREMENT TEST INDEX
4818 MOV R1,R2                         : GET COPY OF INDEX
4819 ADD #TSTLST,R2                     : GET POSITION IN LIST
4820 CMP #DFLTST,R2                     : SEE IF DONE WITH LIST
4821 BNE 2$                             : BR IF NOT DONE YET
4822 BIC #BIT6,ISTKS                     : DISABLE KBD INTERRUPT
4823 JMP ASKTMD                         : GO ASK FOR NEW TEST INPUT MODE
4824
4825 : CHANGE CURRENT TESTS AND ITERATION COUNTS
4826 CHGTST: TYPE OFTEST                : ASK IF TESTS SHOULD BE DEFAULTED
4827 JSR PC,ROCHRS                      : READ RESPONSE
4828 DRVTST                               : (↑C) RETURN ADDRESS
4829 ASKTMD                               : (↑Z) RETURN ADDRESS
4830 CHGTST                               : (↑U) OR ERROR RETURN ADDRESS
4831 TST RO                             : SEE IF NULL INPUT
4832 BEQ NULINP                         : BR IF NULL INPUT
4833 000104 005272 CMP #D,BUFFO         : SEE IF (D) TYPED

```

4834	015046	001405		BEQ	LODFLS		;BR IF DEFAULTS REQUESTED
4835	015050	104401	005272	TYPE	,BUFFO		;ECHO BAD INPUT
4836	015054	104401	001314	TYPE	,SQUES		;TYPE <?> AND <CR>,<LF>
4837	015060	000756		BR	CHGTST		;GO ASK AGAIN
4838				;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST			
4839	015062	012700	005670	LODFLS:	MOV	#DFLTST,R0	;LOAD DEFAULT LIST ADDRESS
4840	015066	012702	005626		MOV	#TSTLST,R2	;LOAD TEST LIST ADDRESS
4841	015072	012022		4\$:	MOV	(R0)+(R2)+	;LOAD A DEFAULT VALUE
4842	015074	022702	005670		CMP	#DFLTST,R2	;SEE IF DONE YET
4843	015100	001374			BNE	4\$	;BR IF NOT DONE YET
4844	015102	105737	003116		TSTB	MOFLAG	;SEE IF DEFAULT MODE
4845	015106	001234			BNE	ASKTMD	;BR IF NOT DEFAULT MODE
4846	015110	000137	016244		JMP	LODFPT	;GO LOAD DEFAULT PARAMETERS
4847	015114	005001		NULINP:	CLR	R1	;INITIALIZE TEST INDEX
4848	015116	104401	010450		TYPE	TLSTHD	;TYPE HEADING "TEST ITERATIONS"
4849				;TYPE CURRENT TEST AND ITERATION NUMBER			
4850	015122	004737	031136	8\$:	JSR	PC,TYPTST	;TYPE A TEST NO. AND ITERATION NO.
4851	015126	104401	012474		TYPE	,SPACE1	;TYPE A SPACE
4852	015132	104401	012514		TYPE	,PROMPT	;TYPE ASTERISK AND SPACE
4853				;READ AND CHECK INPUT IF ANY			
4854	015136	004737	030624		JSR	PC,RDCHRS	;READ OCTAL DIGITS TYPED
4855	015142	013732			DRVTST		;(+C) RETURN ADDRESS
4856	015144	014600			ASKTMD		;(+Z) RETURN ADDRESS
4857	015146	015122			8\$		;(+U) OR ERROR RETURN ADDRESS
4858	015150	005700			TST	R0	;SEE IF ANY INPUT
4859	015152	001012			BNE	12\$	;BR IF ANY INPUT
4860	015154	062701	000002	10\$:	ADD	#2,R1	;INCREMENT INDEX
4861	015160	010102			MOV	R1,R2	;COPY INDEX
4862	015162	062702	005626		ADD	#TSTLST,R2	;GET POSITION IN LIST
4863	015166	022702	005670		CMP	#DFLTST,R2	;SEE IF DONE WITH LIST
4864	015172	001353			BNE	8\$	;BR IF NOT DONE YET
4865	015174	000137	014600		JMP	ASKTMD	;GO ASK FOR NEW TEST INPUT MODE
4866	015200	022737	000041	005272	12\$:	CMP	#',BUFFO
4867	015206	001431			BEQ	16\$	;SEE IF (!) TYPED
4868	015210	005002			CLR	R2	;BR TO PROPAGATE CURRENT ITER. NO.
4869	015212	122760	000041	005271	CMPB	#',BUFFO-1(R0)	;INITIALIZE (!) INDICATOR
4870	015220	001004			BNE	14\$	;SEE IF LAST CHAR IN BUF IS (!)
4871	015222	105060	005271		CLRB	BUFFO-1(R0)	;BR IF NOT (!)
4872	015226	005300			DEC	R0	;INSERT TERMINATOR BYTE
4873	015230	005202			INC	R2	;DECREMENT CHAR COUNT
4874	015232	022700	000005	14\$:	CMP	#5,R0	;SET (!) INDICATOR
4875	015236	002433			BLT	20\$	;SEE HOW MANY CHARS NOW
4876	015240	012746	005272		MOV	#BUFFO, -(SP)	;BR IF TOO MANY (MAX ITER. NO. = 77776)
4877	015244	004737	052464		JSR	PC,OCTBIN	;GET BUF ADDR. ON STACK FOR OCTBIN
4878	015250	015326			20\$		;CHECK DIGITS AND CONVERT TO BINARY
4879	015252	012600			MOV	(SP)+,R0	;ERROR RETURN ADDRESS FOR OCTBIN
4880	015254	020027	077776		CMP	R0,77776	;GET ITERATION NUMBER
4881	015260	101022			BHI	20\$	;SEE IF > 77776
4882	015262	010061	005626		MOV	R0,TSTLST(R1)	;BR IF TOO BIG
4883	015266	005702			TST	R2	;PUT ITERATION NUMBER INTO TEST LIST
4884	015270	001731			BEQ	10\$	;SEE IF (!) WAS TYPED
4885				;PROPAGATE CURRENT ITERATION NO.			;BR IF NOT (!)
4886	015272	010102		16\$:	MOV	R1,R2	;TO THE END OF TSTLST
4887	015274	062702	005626		ADD	#TSTLST,R2	;COPY INDEX
4888	015300	022702	005666		CMP	#DFLTST-2,R2	;GET POSITION IN LIST
4889	015304	001002			BNE	17\$	;SEE IF AT END OF LIST
							;BR IF NOT DONE

E08

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 96
TO - DESIRED TEST INPUT ROUTINESEQ 0374
SEQ 0095

4890	015306	000137	014600			JMP	ASKTMD	:GO ASK FOR NEW TEST INPUT MODE
4891	015312	016161	005626	005630	17\$:	MOV	TSTLST(R1),TSTLST	:+2(R1) ;PROPAGATE TO NEXT WORD
4892	015320	062701	000002			ADD	#2,R1	:INCREMENT INDEX
4893	015324	007752				BR	16\$	:BR TO CONTINUE
4894	015326	104401	005272		20\$:	TYPE	,BUFFO	:ECHO BAD INPUT
4895	015332	104401	001314			TYPE	,SQUES	:TYPE '?' AND <CR>,<LF>
4896	015336	00J671				BR	8\$	:GO ASK AGAIN
4897								
4898								:ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
4899	015340	104401	010500			INPUTP: TYPE	EXPLAN	:EXPLAIN INPUT MODES
4900	015344	005037	005512			ASKMDE: CLR	STALLS	:INHIBIT STALL BETWEEN OPERATIONS
4901	015350	104401	010626			TYPE	,PARMDE	:ASK FOR DESIRED INPUT MODE
4902	015354	104401	012514			TYPE	PROMPT	:TYPE "*"
4903	015360	004737	030624			JSR	PC,ROCHRS	:READ RESPONSE TO MODE QUESTION
4904	015364	013732				DRVTST		:(:C) RETURN ADDRESS
4905	015366	015344				ASKMDE		:(:Z) RETURN ADDRESS
4906	015370	015344				ASKMDE		:(:U) OR ERROR RETURN ADDRESS
4907	015372	005700				TST	RO	:SEE IF NULL INPUT
4908	015374	001005				BNE	4\$	:BR IF ANY INPUT
4909	015376	104401	005272		2\$:	TYPE	,BUFFO	:ECHO BAD INPUT
4910	015402	104401	001314			TYPE	,SQUES	:TYPE '?' AND <CR>,<LF>
4911	015406	000756				BR	ASKMDE	:GO ASK AGAIN
4912	015410	022737	000124	005272	4\$:	CMP	#'T,BUFFO	:SEE IF (T) TYPED
4913	015416	001002				BNE	6\$	:BR IF NOT (T)
4914	015420	000137	015726			JMP	TYPLST	:JUMP TO THE TYPE LIST ROUTINE
4915	015424	022737	000117	005272	6\$:	CMP	#'O,BUFFO	:SEE IF (O) TYPED
4916	015432	001002				BNE	8\$	:BR IF NOT (O)
4917	015434	000137	016200			JMP	OPNLST	:JUMP TO THE OPEN LIST ROUTINE
4918	015440	022737	000123	005272	8\$:	CMP	#'S,BUFFO	:SEE IF (S) TYPED
4919	015446	001002				BNE	10\$	:BR IF NOT (S)
4920	015450	000137	016464			JMP	SETPRM	:JUMP TO THE SET INDIV. PARAM. ROUTINE
4921	015454	022737	000122	005272	10\$:	CMP	#'R,BUFFO	:SEE IF (R) TYPED
4922	015462	001345				BNE	2\$	:BR IF NOT (R) TO ECHO BAD INPUT
4923	015464	023737	005746	005750		CMP	S0,S1	:COMPARE 20(DEC) SECTOR LIMITS
4924	015472	003403				BLE	12\$	:BR IF S0 NOT > S1
4925	015474	104401	010727			TYPE	SECNL1	:TYPE "S0>S1 NOT ALLOWED"
4926	015500	000721				BR	ASKMDE	:GO ASK AGAIN FOR PARAMETERS
4927	015502	023737	005752	005754	12\$:	CMP	S2,S3	:COMPARE 22(DEC) SECTOR LIMITS
4928	015510	003405				BLE	14\$	:BR IF S2 NOT > S3
4929	015512	104401	010757			TYPE	SECNL2	:TYPE "S2>S3"
4930	015516	104401	010740			TYPE	NOTALD	:TYPE "NOT ALLOWED"
4931	015522	000710				BR	ASKMDE	:GO ASK AGAIN FOR PARAMETERS
4932	015524	023737	005740	005742	14\$:	CMP	FT,LT	:COMPARE CHOSEN TRACK LIMITS
4933	015532	003405				BLE	20\$	:BR IF FT NOT > LT
4934	015534	104401	010771			TYPE	TRKNLW	:TYPE "FT>LT"
4935	015540	104401	010740			TYPE	NOTALD	:TYPE "NOT ALLOWED"
4936	015544	000677				BR	ASKMDE	:GO ASK AGAIN FOR PARAMETERS
4937								:CHECK WORD COUNT AGAINST PHYSICAL MEMORY AVAILABLE
4938	015546	013700	005766		20\$:	MOV	WC,RO	:WORD COUNT LO BITS
4939	015552	005300				DEC	RO	:DECREMENT BY 1
4940	015554	005001				CLR	R1	:WORD COUNT HI BITS
4941	015556	000241				CLC		:CONVERT WORD COUNT TO BYTES
4942	015560	006100				ROL	RO	
4943	015562	006101				ROL	R1	
4944	015564	063700	005762			ADD	MA,RO	:ADD WORD COUNT (WC) TO MEM. ADR. (MA)
4945	015570	005501				ADC	R1	

4946	015572	063701	005764		ADD	MA+2,R1	
4947	015576	020137	006130		CMP	R1,MAHILM+2	;COMPARE HI BITS
4948	015602	101004			BHI	26\$	;BR IF WC TOO BIG
4949	015604	001006			BNE	28\$	;BR IF WC IS OK
4950	015606	020037	006126		CMP	RO,MAHILM	;COMPARE LO BITS
4951	015612	101403			BLOS	28\$	;BR IF WC IS OK
4952	015614	104401	011003	26\$:	TYPE	WC2BIG	;TYPE "*** WC OR MA TOO LARGE"
4953	015620	000651			BR	ASKMDE	;GO BACK TO ASK AGAIN FOR PARAMS
4954							
4955	015622	004737	027000		;SAVE XXDP LOADER IF NECESSARY		
4956	015626	015634		28\$:	JSR	PC,CHKXDP	;GO SAVE XXDP LOADER IF NECESSARY
4957	015630	000137	016720		36\$		; "NO SAVE" RETURN ADDRESS
4958	015634	105737	000041	30\$:	JMP	RUNTST	;JUMP TO THE RUN TESTS ROUTINE
4959	015640	001773		36\$:	TSTB	2#41	;SEE IF LOADED BY XXDP
4960	015642	104401	007443		BEQ	30\$	;BR IF NOT
4961	015646	004737	030624		TYPE	OVLODR	;TYPE "OVERLAY LOADER ? (Y OR N) *"
4962	015652	015634			JSR	PC,RDCHRS	;READ RESPONSE
4963	015654	015634			36\$		; (↑C) RETURN ADDRESS
4964	015656	015634			36\$		; (↑Z) RETURN ADDRESS
4965	015660	022700	000001		36\$		; (↑U) OR ERROR RETURN ADDRESS
4966	015664	001405			CMP	#1,RO	;SEE IF 1 CHAR TYPED
4967	015666	104-01	005272		BEQ	38\$	;BR IF 1 CHAR TYPED
4968	015672	104401	001314	37\$:	TYPE	,BUFFO	;ECHO BAD INPUT
4969	015676	000756			TYPE	\$QUES	
4970	015700	122737	000131	005272	BR	36\$	;GO ASK AGAIN
4971	015706	001001		38\$:	CMPB	#'Y,BUFFO	;SEE IF (Y) TYPED
4972	015710	000747			BNE	40\$	;BR IF NOT (Y)
4973	015712	122737	000116	005272	BR	30\$	;GO RUN TESTS
4974	015720	001362		40\$:	CMPB	#'N,BUFFO	;SEE IF (N) TYPED
4975	015722	000137	015344		BNE	37\$	;BR IF NOT (N)
4976					JMP	ASKMDE	;GO ASK FOR PARAMETERS AGAIN

.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 DRVST,
;AND (↑Z) CAUSES RETURN TO ASKMDE.

4987	015726				TYPLST:		
4988	015726	005001			CLR	R1	;INITIALIZE INDEX
4989	015730	004737	026040		JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
4990	015734	004737	031172		JSR	PC,TYPPRM	;TYPE CURRENT PARAMETER AND VALUE
4991	015740	104401	001315		TYPE	\$CRLF	;TYPE <CR>,<LF>
4992	015744	005737	005532		TST	INTCHR	;SEE IF ANY INPUT AT KBD
4993	015750	001420			BEQ	8\$	;BR IF NO INPUT
4994	015752	122737	000003	005532	CMPB	#003,INTCHR	;SEE IF (↑C) TYPED
4995	015760	001002			4\$		;BR IF NOT (↑C)
4996	015762	000137	013732		JMP	DRVST	;JUMP TO ASK FOR DRIVE(S) AGAIN
4997	015766	122737	000032	005532	4\$:	CMPB	#032,INTCHR
4998	015774	001002			BNE	6\$	;SEE IF (↑Z) TYPED
4999	015776	000137	015344		JMP	ASKMDE	;BR IF NOT (↑Z)
5000	016002	004737	026060		JSR	PC,ECOBAD	;JUMP TO ASK FOR NEW MODE
5001	016006	004737	026040		JSR	PC,PREPKB	;ECHO BAD INPUT
							;PREPARE FOR POSSIBLE KBD INPUT

```

5002 016012 022761 040515 006152 8$: CMP      #MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
5003 016020 001002          BNE      10$      ;BR IF NOT (MA)
5004 016022 062701 000002          ADD      #2,R1 ;INCREMENT PARAMETER INDEX
5005 016026 062701 000002          ADD      #2,R1 ;INCREMENT PARAMETER INDEX
5006 016032 010102          MOV      R1,R2 ;GET COPY OF INDEX
5007 016034 062702 005732          ADD      #PRMLST,R2 ;COMPUTE POSITION IN LIST
5008 016040 022702 005776          CMP      #PRDFLT,R2 ;SEE IF DONE WITH LIST
5009 016044 001333          BNE      1$      ;BR IF NOT DONE YET
5010 016046 032737 100000 005760          BIT      #BIT15,PT ;SEE IF PATTERN 15 SPECIFIED
5011 016054 001005          BNE      12$      ;BR IF PATTERN SPECIFIED
5012 016056 042777 000100 163060          BIC      #BIT6,2$TKS ;DISABLE KBD INTERRUPT
5013 016064 000137 015344          JMP      ASKMDE ;JUMP TO ASK FOR NEW MODE
5014          ;TYPE OUT USER-DEFINED PATTERN 15
5015 016070 104401 011116 12$: TYPE      PFIFTH ;TYPE "USER-DEFINED PATTERN 15 : "
5016 016074 005001          CLR      R1 ;INITIALIZE WORD INDEX
5017 016076 005737 005532          TST      INTCHR ;SEE IF ANY INPUT
5018 016102 001420          BEQ      20$      ;BR IF NO INPUT
5019 016104 122737 000003 005532          CMPB     #003,INTCHR ;SEE IF (↑C) TYPED
5020 016112 001002          BNE      16$      ;BR IF NOT (↑C)
5021 016114 000137 013732          JMP      DRVTST ;JUMP TO ASK FOR DRIVE(S) AGAIN
5022 016120 122737 000032 005532          CMPB     #032,INTCHR ;SEE IF (↑Z) TYPED
5023 016126 001002          BNE      18$      ;BR IF NOT (↑Z)
5024 016130 000137 015344          JMP      ASKMDE ;JUMP TO ASK FOR NEW MODE
5025 016134 004737 026060          JSR      PC,ECOBAD ;ECHO BAD INPUT
5026 016140 004737 026040          JSR      PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5027 016144 004737 031252          JSR      PC,TYPPAT ;TYPE WORD XX = YYYYYY
5028 016150 104401 001315          TYPE     $CRLF ;TYPE <CR>,<LF>
5029 016154 062701 000002          ADD      #2,R1 ;INCREMENT INDEX
5030 016160 022701 000040          CMP      #32.,R1 ;SEE IF 16 WORDS TYPED
5031 016164 001344          BNE      14$      ;BR IF NOT DONE YET
5032 016166 042777 000100 162750          BIC      #BIT6,2$TKS ;DISABLE KBD INTERRUPT
5033 016174 000137 015344          JMP      ASKMDE ;JUMP TO ASK FOR NEW MODE
5034
5035
5036
5037
5038          ;SBTTL TO - OPEN (O) LIST ROUTINE
5039          ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
5040          ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
5041          ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
5042          ;*TTY INPUT. TYPING (↑C) CAUSES IMMEDIATE RETURN TO
5043          ;*DRVTST, AND (↑Z) CAUSES RETURN TO ASKMDE.
5044
5045          OPNLST:
5046 016200 104401 011034          TYPE      DFQUES ;ASK IF ALL DEFAULT VALUES DESIRED
5047 016204 004737 030624          JSR      PC,ROCHRS ;READ RESPONSE TO DEFAULT QUESTION
5048 016210 013732          DRVTST          ;(↑C) RETURN ADDRESS
5049 016212 015344          ASKMDE          ;(↑Z) RETURN ADDRESS
5050 016214 016200          OPNLST          ;(↑U) OR ERROR RETURN ADDRESS
5051 016216 005700          JSR      RO ;SEE IF NULL INPUT
5052 016220 001433          BEQ      NOTDFT ;BR IF DEFAULTS NOT REQUESTED
5053 016222 022737 000104 005272          CMP      #D,BUFFO ;SEE IF (D) TYPED
5054 016230 001405          BEQ      LODFPT ;BR IF DEFAULTS DESIRED
5055 016232 104401 005272          TYPE      ,BUFFO ;ECHO BAD INPUT
5056 016236 104401 001314          TYPE      $QUES
5057 016242 000756          BR      OPNLST ;GO ASK AGAIN

```

```

5058      ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
5059      LODFPT: MOV      #PROFLT,R0      ;LOAD DEFAULT LIST ADDRESS
5060      MOV      #PRMLST,R2      ;LOAD PARAMETER LIST ADDRESS
5061      4$:      MOV      (R0)+,(R2)+    ;LOAD A DEFAULT VALUE
5062      CMP      #PROFLT,R2      ;SEE IF DONE YET
5063      BNE      4$              ;BR IF MORE VALUES TO LOAD YET
5064      TSTB     MDFLAG          ;SEE IF DEFAULT MODE
5065      BNE      6$              ;BR IF NOT DEFAULT MODE
5066      MOV      #1,SEOPCT      ;SET PASS COUNT = 1
5067      JMP      STPASS         ;GO RUN DFLT TESTS
5068      6$:      JMP      ASKMDE      ;JUMP TO ASK FOR NEW MODE
5069      NOTDFT: CLR      R1        ;INITIALIZE INDEX
5070      ;TYPE CURRENT PARAMETER AND VALUE
5071      8$:      JSR      PC,TYPRM    ;TYPE PARAMETER AND VALUE
5072      TYPE     'SPACE1        ;TYPE A SPACE
5073      TYPE     'PROMPT        ;TYPE ASTERISK AND SPACE
5074      ;READ AND CHECK INPUT, IF ANY
5075      JSR      PC,RDCHRS        ;READ OCTAL DIGITS TYPED
5076      DRVTST   ;(C) RETURN ADDRESS FOR RDCHRS
5077      ASKMDE   ;(Z) RETURN ADDRESS FOR RDCHRS
5078      8$:      ;(U) OR ERROR RETURN ADDR. FOR RDCHRS
5079      TST      R0              ;SEE IF ANY INPUT
5080      BNE      10$            ;BR IF ANY INPUT
5081      CMP      #'MA,PRMNM(R1) ;SEE IF (MA) JUST DEFAULTED
5082      BNE      12$            ;BR IF NOT (MA)
5083      ADD      #2,R1           ;INCREMENT INDEX
5084      BR       12$            ;BR TO MOVE ON TO NEXT PARAMETER
5085      10$:     CMP      #10,R0    ;SEE IF 8 CHARACTERS TYPED
5086      BLT      14$            ;BR IF MORE THAN 8 TYPED
5087      MOV      #BUFFD, -(SP)   ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
5088      JSR      PC,OCTBIN      ;CHECK DIGITS AND CONVERT TO BINARY
5089      14$:     ;ERROR RETURN ADDRESS FOR OCTBIN
5090      MOV      (SP)+,LOWOCT    ;GET LOW BINARY BITS
5091      MOV      $HIOCT,HIGOCT  ;GET HIGH BINARY BITS
5092      ;CHECK PARAMETER VALUE AND PUT INTO LIST
5093      JSR      PC,CHKPRM      ;CHECK VALIDITY OF PARAM VALUE
5094      14$:     ;ERROR RETURN ADDR. FOR CHKPRM
5095      ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5096      12$:     JSR      PC,MODP15
5097      ;MOVE ON TO NEXT PARAMETER
5098      ADD      #2,R1           ;INCREMENT THE PARAMETER INDEX
5099      MOV      R1,R2           ;GET COPY OF INDEX
5100      ADD      #PRMLST,R2      ;COMPUTE POSITION IN LIST
5101      CMP      #PROFLT,R2      ;SEE IF DONE WITH LIST
5102      BNE      8$              ;BR IF NOT DONE YET
5103      JMP      ASKMDE          ;JUMP TO ASK FOR NEW MODE
5104      14$:     TYPE     'BUFFD    ;ECHO BAD INPUT
5105      TYPE     '$QUES          ;TYPE <?> AND <CR>, <LF>
5106      BR       8$              ;BR TO ASK AGAIN
5107
5108
5109
5110      .SBTTL TO - SET (S) INDIVIDUAL PARAM ROUTINE
5111      ;*THIS ROUTINE IS USED TO ALLOW SETTING OF ANY INDIVIDUAL
5112      ;*PARAMETER VALUE BY TTY INPUT. A PROMPTER (>) IS
5113      ;*TYPED AND THE ROUTINE WAITS FOR INPUT OF THE FORM

```

```

5114      ; * XX=YYYYYY, WHERE XX IS THE MNEMONIC FOR ANY VALID
5115      ; * PARAMETER IN THE LIST, AND YYYYYY IS THE DESIRED PARAMETER
5116      ; * VALUE IN OCTAL DIGITS. THE PROGRAM THEN ASKS FOR ANOTHER
5117      ; * PARAMETER BY TYPING ">" AGAIN. TYPING (↑C) CAUSES
5118      ; * IMMEDIATE RETURN TO DRVST, AND (↑Z) CAUSES RETURN TO
5119      ; * ASKMODE.
5120
5121      SETPRM:
5122      016464      104401      012517      TYPE      PRMPSP      ; TYPE ">" PROMPTER
5123      016464      ;READ AND CHECK INPUT, IF ANY      PC, RDCHRS
5124      016470      004737      030624      JSR      ; READ INPUT LINE
5125      016474      013732      DRVST      ; (↑C) RETURN ADDRESS FOR RDCHRS
5126      016476      015344      ASKMODE      ; (↑Z) RETURN ADDRESS FOR RDCHRS
5127      016500      016464      SETPRM      ; (↑U) OR ERROR RETURN ADDR. FOR RDCHRS
5128      016502      020027      000004      CMP      RD, #4      ; SEE IF AT LEAST 4 CHARACTERS TYPED
5129      016506      002005      BGE      6$      ; BR IF AT LEAST 4 TYPED
5130      016510      104401      005272      4$:      TYPE      ,BUFFO      ; ECHO BAD INPUT
5131      016514      104401      001314      TYPE      $QUES      ; TYPE (↑) AND (CR), (LF)
5132      016520      000761      BR      SETPRM      ; BR TO ASK FOR INPUT AGAIN
5133      016522      020027      000013      6$:      CMP      RD, #11.      ; SEE IF ELEVEN OR LESS CHARS TYPED
5134      016526      003370      BGT      4$      ; BR IF MORE THAN ELEVEN TYPED
5135      ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
5136      016530      005001      CLR      R1      ; INITIALIZE PARAMETER INDEX
5137      016532      023761      005272      8$:      CMP      BUFFO, PRMNEM(R1)      ; TRY TO MATCH 2-CHAR MNEMONIC
5138      016540      001411      10$      BEQ      10$      ; BR IF PARAMETER FOUND IN TABLE
5139      016542      062701      000002      ADD      #2, R1      ; INCREMENT INDEX
5140      016546      010102      MOV      R1, R2      ; GET COPY OF INDEX
5141      016550      062702      005732      ADD      #PRMLST, R2      ; SEE IF ALL ENTRIES HAVE BEEN CHECKED
5142      016554      022702      005776      CMP      #PRDFLT, R2
5143      016560      001364      BNE      8$      ; BR IF MORE TO CHECK YET
5144      016562      000752      BR      4$      ; BR TO ECHO BAD INPUT
5145
5146      016564      012702      000002      10$:      MOV      #2, R2      ; INITIALIZE BUFFER POINTER
5147      016570      122762      000075      005272      CMPB      #'=', BUFFO(R2)      ; SEE IF NEXT CHAR IS "="
5148      016576      001411      12$      BEQ      12$      ; BR IF IT IS "="
5149      016600      122762      000040      005272      CMPB      #040, BUFFO(R2)      ; SEE IF NEXT CHAR IS A SPACE
5150      016606      001340      4$      BNE      4$      ; BR TO ECHO BAD INPUT
5151      016610      005202      INC      R2      ; INCREMENT BUFFER POINTER
5152      016612      122762      000075      005272      CMPB      #'=', BUFFO(R2)      ; SEE IF NEXT CHAR IS "="
5153      016620      001333      4$      BNE      4$      ; BR TO ECHO INVALID INPUT
5154      016622      005202      12$:      INC      R2      ; INCREMENT BUFFER POINTER
5155      016624      020200      CMP      R2, RO      ; SEE IF MORE CHARS LEFT IN BUFFER
5156      016626      002330      4$      BGE      4$      ; BR TO ECHO INVALID INPUT
5157      016630      122762      000040      005272      CMPB      #040, BUFFO(R2)      ; SEE IF NEXT CHARACTER IS SPACE
5158      016636      001003      14$      BNE      14$      ; BR IF NOT A SPACE
5159      016640      005202      INC      R2      ; INCREMENT BUFFER POINTER
5160      016642      020200      CMP      R2, RO      ; SEE IF MORE CHARS LEFT IN BUFFER
5161      016644      002321      4$      BGE      4$      ; BR IF NONE LEFT
5162      016646      160200      14$:      SUB      R2, RO      ; SUBTRACT POINTER FROM CHAR COUNT
5163      016650      020027      000010      CMP      RO, #10      ; SEE HOW MANY CHARS LEFT
5164      016654      003315      BGT      4$      ; BR IF TOO MANY LEFT
5165
5166      016656      062702      005272      ;CHECK FOR LEGAL PARAMETER VALUE
5167      016662      010246      ADD      #BUFFO, R2      ; GET ADDRESS OF DIGITS
5168      016664      004737      052464      MOV      R2, -(SP)      ; PUT IT ON STACK FOR OCTBIN
5169      016670      016510      JSR      PC, OCTBIN      ; CHECK DIGITS AND CONVERT TO BINARY
5169      4$      ; ERROR RETURN ADDR. FOR OCTBIN

```

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 101
TO - SET (S) INDIVIDUAL PARAM ROUTINE

SEQ 0379
SEQ 0100

5170	016672	012637	005476		MOV	(SP)+,LOWOCT	;GET LOW BINARY BITS
5171	016676	013737	052616	005500	MOV	\$HI OCT,HIG OCT	;GET HIGH BINARY BITS
5172	016704	004737	031630		JSR	PC,CHKPRM	;CHECK VALIDITY OF PARAMETER VALUE
5173	016710	016510			4\$		;ERROR RETURN ADDR. FOR CHKPRM
5174							;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5175	016712	004737	031304		JSR	PC,MODP15	
5176	016716	000662			BR	SETFRM	;RETURN TO ASK FOR ANOTHER PARAMETER
5177							
5178							
5179							
5180							
5181							
5182							
5183							
5184							
5185							
5186							
5187							
5188							
5189							
5190							
5191							
5192							
5193	016720						
5194							
5195	016720	104401	011242				
5196	016724	004737	030624				
5197	016730	013732					
5198	016732	015344					
5199	016734	016720					
5200	016736	005700					
5201	016740	001403					
5202	016742	022700	000005				
5203	016746	002005					
5204	016750	104401	005272				
5205	016754	104401	001314				
5206	016760	000757					
5207	016762	012746	005272				
5208	016766	004737	052464				
5209	016772	016750					
5210	016774	012637	025474				
5211	017000	001763					
5212							
5213	017002	005037	001326				
5214	017006	112737	000001	003120			
5215	017014	012737	177777	005510			
5216	017022	005037	001330				
5217							
5218	017026	012737	000000	177776			
5219	017034	012706	001100				
5220	017040	105037	003130				
5221	017044	005037	005532				
5222	017050	005237	005510				
5223	017054	022737	000010	005510			
5224	017062	001002					
5225	017064	000137	025442				

.SBTTL TO - RUN (R) TESTS ROUTINE

;*THIS ROUTINE IS ENTERED TO SET UP THE RUNNING OF TESTS,
;*WHOSE PARAMETERS HAVE BEEN DETERMINED. IT FIRST ASKS
;*FOR THE DESIRED NUMBER OF PROGRAM PASSES, AND STORES
;*THIS IN \$EOPCT FOR THE END OF PASS ROUTINE. THEN, THE
;*DRIVE LIST IS SCANNED FOR THE FIRST DESIRED DRIVE, AND
;*THE HEADER ON SECTOR 0 IS READ TO GET THE FORMAT FOR THIS
;*DRIVE. THIS IS STORED IN THE FORMAT BYTE, AND A JUMP IS
;*MADE TO THE FIRST TEST.
;*THE END OF PASS ROUTINE RETURNS TO "NEWDRV" TO SELECT
;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
;* (ALL DRIVES) IS COMPLETED.

RUNTST:

;INPUT THE DESIRED NUMBER OF PROGRAM PASSES

4\$:	TYPE	ENTPAS	;ASK FOR NUMBER OF PASSES
	JSR	PC,ROCHRS	;READ RESPONSE
	DRVTST		; (↑C) RETURN ADDRESS
	ASKMDE		; (↑Z) RETURN ADDRESS
	4\$		; (↑U) OR ERROR RETURN ADDRESS
	TST	RO	;SEE IF NULL INPUT
	BEQ	6\$	;GO ASK AGAIN
	CMP	#5,RO	;SEE HOW MANY CHARS TYPED
	BGE	8\$	;BR IF 5 OR LESS
6\$:	TYPE	,BUFFD	;ECHO BAD INPUT
	TYPE	,\$QUES	
	BR	4\$	;GO ASK AGAIN
8\$:	MOV	#BUFFD,-(SP)	;GET BUF ADDR ON STACK FOR OCTBIN
	JSR	PC,OCTBIN	;CHECK DIGITS AND CONVERT TO BINARY
	6\$		;ERROR RETURN ADDRESS FOR OCTBIN
	MOV	(SP)+,\$EOPCT	;SET DESIRED NO. OF PASSES (1-77777)
	BEQ	6\$	;BR IF 0, TO ASK AGAIN
			;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
STPASS:	CLR	\$PASS	;INIT. THE PASS NUMBER
	MOVB	#1,TSTING	;SET "RUNNING TESTS" FLAG
NEWPAS:	MOV	#177777,DRIVE	;INITIALIZE DRIVE NO. TO -1
	CLR	\$DEVCT	;INIT APT DEVICE COUNT TO 0
			;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
NEWDRV:	MOV	#PRO,2#PS	;RE-ESTABLISH PRIORITY 0
	MOV	#STACK,SP	;RESTORE THE STACK
	CLAB	DRIVERS	;CLEAR ERROR COUNT FOR CURRENT DRIVE
	CLR	INTCHR	;CLEAR TTY INPUT BUFFER WORD
	INC	DRIVE	;INCREMENT DRIVE NUMBER
	CMP	#10,DRIVE	;SEE IF DONE WITH THIS PASS
	BNE	2\$	;BR IF NOT DONE CHECKING LIST
	JMP	DUNPAS	;JUMP IF DONE WITH THIS PASS

K08

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 102
TO - RUN (R) TESTS ROUTINESEQ 0380
SEQ 0101

5226	017070	013700	005510	2\$:	MOV	DRIVE,RO	;GET NEW DRIVE NUMBER
5227	017074	105760	005606		TSTB	DRVLST(RO)	;SEE IF THIS DRIVE IS MARKED IN LIST
5228	017100	001752			BEQ	NEWDRV	;BR IF NOT MARKED, TO CHECK ANOTHER
5229	017102	105037	001102		CLRB	\$TSTNM	;INIT TEST NUMBER TO 0
5230	017106	005037	001304		CLR	\$TIMES	;INIT ITERATION COUNT
5231	017112	105760	005616		TSTB	DTYLST(RO)	;SEE IF RK07
5232	017116	001077			BNE	1\$	;BR IF YES
5233	017120	105037	003124		CLRB	TYPFMT	;ELSE LOAD ALL RK06 PARAMETERS
5234	017124	012737	000625	017754	MOV	#625,LCMS	
5235	017132	012737	000631	017756	MOV	#631,LCM1	
5236	017140	012737	000631	006044	MOV	#631,PRMLIM+2	
5237	017146	012737	000632	005734	MOV	#632,LC	
5238	017154	012737	000632	017760	MOV	#632,LSTCYL	
5239	017162	012737	000632	006000	MOV	#632,PRDFLT+2	
5240	017170	012737	000632	006050	MOV	#632,PRMLIM+6	
5241	017176	012737	000632	006054	MOV	#632,PRMLIM+12	
5242	017204	012737	000633	017762	MOV	#633,LCP1	
5243	017212	012737	000400	017764	MOV	#400,HOLD1	
5244	017220	012737	001000	017766	MOV	#1000,HOLD2	
5245	017226	012737	000025	017770	MOV	#25,HOLD3	
5246	017234	012737	010025	017772	MOV	#10025,HOLD4	
5247	017242	012737	010125	017774	MOV	#10125,HOLD5	
5248	017250	012737	000002	017776	MOV	#2,HOLD6	
5249	017256	012737	110244	020000	MOV	#110244,HOLD7	
5250	017264	012737	017500	007216	MOV	#8000,MAXL02	;8000 US FOR RK06
5251	017272	012737	000000	007220	MOV	#0,MAXH12	
5252	017300	012737	022370	007222	MOV	#9464,MAXL04	;75000 US FOR RK06
5253	017306	012737	000001	007224	MOV	#1,MAXH14	
5254	017314	000477			BR	3\$	
5255							
5256	017316	152737	000004	003124	1\$:	BISB	#B.CDT,TYPFMT ;LOAD ALL RK07 PARAMETERS
5257	017324	012737	001451	017754	MOV	#1451,LCMS	
5258	017332	012737	001455	017756	MOV	#1455,LCM1	
5259	017340	012737	001455	006044	MOV	#1455,PRMLIM+2	
5260	017346	012737	001456	005734	MOV	#1456,LC	
5261	017354	012737	001456	017760	MOV	#1456,LSTCYL	
5262	017362	012737	001456	006000	MOV	#1456,PRDFLT+2	
5263	017370	012737	001456	006050	MOV	#1456,PRMLIM+6	
5264	017376	012737	001456	006054	MOV	#1456,PRMLIM+12	
5265	017404	012737	001457	017762	MOV	#1457,LCP1	
5266	017412	012737	001000	017764	MOV	#1000,HOLD1	
5267	017420	012737	002000	017766	MOV	#2000,HOLD2	
5268	017426	012737	002025	017770	MOV	#2025,HOLD3	
5269	017434	012737	012025	017772	MOV	#12025,HOLD4	
5270	017442	012737	012125	017774	MOV	#12125,HOLD5	
5271	017450	012737	000012	017776	MOV	#12,HOLD6	
5272	017456	012737	016104	020000	MOV	#16104,HOLD7	
5273							
5274	017464	012737	014544	007216	MOV	#6500,MAXL02	;6500 US FOR RK07
5275	017472	012737	000000	007220	MOV	#0,MAXH12	
5276	017500	012737	012530	007222	MOV	#5464,MAXL04	;71000 US FOR RK07
5277	017506	012737	000001	007224	MOV	#1,MAXH14	
5278	017514	010037	001332		3\$:	MOV	RO,\$UNIT ;SET DRIVE NO FOR APT
5279							
5280	017520	004737	027522		;READ THE HEADER ON SECTOR 0, TRACK 0, CYL 0, AND GET THE DRIVE FORMAT		
5281	017524	112765	000125	000001	JSR	PC,INITSS	;INIT. DRIVER PARAMS AND S.S.
					MOVB	#RDHEAD,P.CMND(R5)	;SET READ HEADER COMMAND

5282	017532	004737	040316		JSR	PC, DRVCAL	: DO READ HEADER
5283	017536	013712	017770		MOV	HOLD3, (R2)	: ISSUE RD HDR WITH FMT BIT = 0
5284	017542	032712	000200	4\$:	BIT	#BIT7, (P2)	: CHECK FOR C. READY IN CSI
5285	017546	001775			BEQ	4\$	: BR IF NOT RDY YET
5286	017550	013712	017772		MOV	HOLD4, (R2)	: ISSUE RD HDR WITH FMT BIT = 1
5287	017554	032712	000200	6\$:	BIT	#BIT7, (R2)	: CHECK FOR C. READY IN CSI
5288	017560	001775			BEQ	6\$	: BR IF NOT RDY YET
5289	017562	105037	003125		CLRB	FORMAT	: INITIALIZE FORMAT BYTE TO 0
5290	017566	013737	005752	005516	MOV	S2, FS	: INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
5291	017574	013737	005754	005520	MOV	S3, LS	
5292	017602	005762	000024		TST	RKDB(R2)	: POP SILO ONE TIME
5293	017606	032762	001000	000024	BIT	#BIT9, RKDB(R2)	: TEST FORMAT BIT IN HEADER WORD 2
5294	017614	001411			BEQ	8\$	: BR IF 22 SECTOR FORMAT
5295	017616	152737	000020	003125	BISB	#B.CFMT, FORMAT	: SET 20 SECTOR FORMAT FOR THIS DRIVE
5296	017624	013737	005746	005516	MOV	S0, FS	: INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
5297	017632	013737	005750	005520	MOV	S1, LS	
5298	017640	013737	005772	005512	MOV	ST, STALLS	: SET NUMBER OF UNIT STALLS DESIRED
5299	017646	004737	027522		JSR	PC, INITSS	: INIT THE S.S.
5300	017652	004737	036674		JSR	PC, REDBSF	: READ BAD SECTOR FILE FOR THIS DRIVE
5301	017656	113737	005510	011370	MOVB	DRIVE DRVNO	: GET DRIVE NO.
5302	017664	152737	000060	011370	BISB	#'0, DRVNO	: CONVERT TO ASCII
5303	017672	104401	011351		TYPE	TSTORN	: TYPE "TESTING DRIVE X"
5304	017676	005737	001326		TST	\$PASS	: SEE IF THIS IS FIRST PASS
5305	017702	001012			BNE	10\$	: BR IF NOT FIRST PASS
5306	017704	004737	032134		JSR	PC, DRVSR	: TYPE "DRIVE SER. NO. XXX"
5307	017710	004737	032254		JSR	PC, CRTSER	: TYPE "CART. SER. NO. XXXXXXXXXXXX"
5308	017714	032737	000020	005770	BIT	#BIT4, CS	: SEE IF BAD SECTORS SHOULD BE TYPED
5309	017722	001402			BEQ	10\$	: BR IF NOT
5310	017724	004737	037164		JSR	PC, TYPBSF	: TYPE BAD SECTOR FILES
5311	017730	005037	005532	10\$:	CLR	INTCHR	: INIT. TTY INPUT CHAR BUFFER
5312	017734	105737	003136		TSTB	DULACS	: SEE IF DUAL-ACCESS FLAG SET
5313	017740	001420			BEQ	TST1	: BR IF NOT
5314	017742	112737	000021	001102	MOVB	#21, \$STNM	: SET TEST NO.
5315	017750	000137	024516		JMP	RWD1ST	: JUMP TO R/W DATA TEST
5316							
5317	017754	000000			LCM5:	0	: LAST CYL-5
5318	017756	000000			LCM1:	0	: LAST CYL-1
5319	017760	000000			LSTCYL:	0	: LAST CYL 632 FOR RK06, 1456 FOR RK07
5320	017762	000000			LCPI:	0	: LAST CYL+1
5321	017764	000000			HOLD1:	0	
5322	017766	000000			HOLD2:	0	
5323	017770	000000			HOLD3:	0	
5324	017772	000000			HOLD4:	0	
5325	017774	000000			HOLD5:	0	
5326	017776	000000			HOLD6:	0	
5327	020000	000000			HOLD7:	0	
5328							
5329							
5330							
5331							

MO8

5332
5333
5334
5335
5336
5337
5338
5339 020002 000004
5340 020004 012737 000001 001324
5341 020012 004737 030462
5342 020016 004737 030530
5343 020022 000137 020062
5344
5345 020026 004737 027522
5346 020032 004737 026040
5347
5348 020036 112765 000113 000001
5349 020044 004737 040316
5350
5351 020050 013765 005734 000002
5352 020056 004737 032374
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363 020062 000004
5364 020064 012737 000002 001324
5365 020072 004737 030462
5366 020076 004737 030530
5367 020102 000137 020142
5368
5369 020106 004737 027522
5370 020112 004737 026040
5371
5372 020116 013765 005732 000002
5373 020124 004737 032374
5374
5375 020130 013765 005734 000002
5376 020136 004737 032374
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387

```
*****
*TEST 1          RECALIBRATE/SEEK TEST
*
*      THIS TEST PERFORMS A RECALIBRATE, FOLLOWED BY A
*      SEEK TO CYLINDER LC.
*
*****
TST1:  SCOPE
        MOV     #1,TESTN      ;SET TEST NUMBER IN APT MAIL BOX
        JSR     PC,SETUP      ;SET UP FOR LOOP ON ERROR
        JSR     PC,CHKITR     ;SEE IF ITER. NO. = 0 FOR THIS TEST
        JMP     TST2          ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
        JSR     PC,INITSS      ;INITIALIZE DRIVER PARAMS AND SUB-SYS
        JSR     PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT
;RECALIBRATE THE DRIVE
        MOV     #RECAL.P.CMND(R5) ;SET RECAL COMMAND
        JSR     PC,DRVCL       ;RECALIBRATE DRIVE
;DO SEEK TO CYLINDER LC
        MOV     LC,P.CYLN(R5)   ;SET CYLINDER = LC
        JSR     PC,SEEKER       ;DO A READ HEADER OR EXPLICIT SEEK
*****

*TEST 2          SEEK/SEEK TEST
*
*      THIS TEST PERFORMS A SEEK TO CYLINDER FC, FOLLOWED
*      BY A SEEK TO CYLINDER LC.
*
*****
TST2:  SCOPE
        MOV     #2,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     TST3          ;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
;DO SEEK TO CYLINDER FC
        MOV     FC,P.CYLN(R5)   ;SET CYL = FC
        JSR     PC,SEEKER       ;SEEK TO FC
;DO SEEK TO CYLINDER LC
        MOV     LC,P.CYLN(R5)   ;SET CYL = LC
        JSR     PC,SEEKER       ;SEEK TO LC
*****

*TEST 3          CYLINDER ADDRESSING TEST
*
*      THIS TEST VERIFIES THE CYLINDER ADDRESS BITS, BY
*      SEEKING TO CYLINDERS 0,1,2,4,10,20,40,100,200,400(OCT).
*      THEN, THE SEEKS ARE DONE IN REVERSE, TO 400,200,100,
*      40,20,10,4,2,1,0.
*
*****
```

5388
5389 020142 000004
5390 020144 012737 000003 001324
5391 020152 004737 030462
5392 020156 004737 030530
5393 020162 000137 020250
5394
5395 020166 004737 027522
5396 020172 004737 026040
5397
5398 020176 005065 000002
5399 020202 004737 032374
5400
5401 020206 005265 000002
5402 020212 004737 032374
5403 020216 006365 000002
5404 020222 033765 017766 000002
5405 020230 001770
5406
5407 020232 006265 000002
5408 020236 004737 032374
5409 020242 005765 000002
5410 020246 001371
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437 020250 000004
5438 020252 012737 000004 001324
5439 020260 004737 030462
5440 020264 004737 030530
5441 020270 000137 020414
5442
5443 020274 004737 027522

```

*****
TST3:  SCOPE
      MOV     #3,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
      JSR     PC,SETUP        ;SET UP FOR LOOP ON ERROR
      JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP     TST4            ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
      JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
      JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
;SEEK TO CYLINDER 0
      CLR     P,CYLN(R5)      ;SET CYL = 0
      JSR     PC,SEEKER        ;SEEK TO CYL 0
;SEEK TO CYLINDERS 1,2,4,10,20,40,100,200,400 (FORWARD DIRECTION)
      INC     P,CYLN(R5)      ;INITIALIZE CYL TO 1
2$:    JSR     PC,SEEKER        ;SEEK TO CURRENT CYL
      ASL     P,CYLN(R5)      ;GET NEXT CYL NO.
      BIT     HOLD2,P,CYLN(R5);SEE IF DONE WITH FORWARD SEEKS
      BEQ     2$              ;BR IF NOT DONE YET
;SEEK TO CYLINDERS 400,200,100,40,20,10,4,2,1,0 (REVERSE DIRECTION)
4$:    ASR     P,CYLN(R5)      ;GET NEXT CYL NO.
      JSR     PC,SEEKER        ;SEEK TO CURRENT CYLINDER
      TST     P,CYLN(R5)      ;SEE IF DONE IN REVERSE DIRECTION
      BNE     4$              ;BR IF NOT DONE YET

```

```

*****
*TEST 4      CYLINDER ADDRESS CROSSTALK TEST
*
*  THIS TEST PERFORMS SEEKS TO CYLINDERS WHOSE ADDRESSES
*  ARE SUSCEPTIBLE TO BIT CROSSTALK WITHIN THE CYLINDER
*  ADDRESSING LOGIC. THE CYLINDER ADDRESS BITS SEQUENCE
*  AS FOLLOWS :
*
*      000 000 000
*      011 111 111
*      000 000 000
*      011 111 110
*      000 000 000
*      011 111 101
*      000 000 000
*
*      .
*
*      000 000 000
*      010 111 111
*      000 000 000
*      101 111 111
*
*****

```

```

*****
TST4:  SCOPE
      MOV     #4,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
      JSR     PC,SETUP        ;SET UP FOR LOOP ON ERROR
      JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP     TST5            ;JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
      JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS

```

```

5444 020300 004737 026040      JSR      PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT*
5445      ;SEEK TO CYL 0
5446 020304 005065 000002      CLR      P.CYLN(R5)      ;SET CYL = 0
5447 020310 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYLINDER 0
5448      ;SEEK TO CYL 377 (OCT)
5449 020314 012765 000377 000002  MOV      #377,P.CYLN(R5) ;SET CYL = 377
5450 020322 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYL 377
5451      ;SEEK TO REMAINING CROSSTALK CYLINDERS
5452 020326 012737 000001 005514  MOV      #1,CYLNDR      ;INITIALIZE BIT MASK
5453 020334 005065 000002      CLR      P.CYLN(R5)      ;SET CYL = 0
5454 020340 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYL 0
5455 020344 032737 000200 005514  BIT      #BIT07,CYLNDR    ;SEE IF ABOUT TO DO LAST SEEK
5456 020352 001013      BNE      4$      ;BR IF THIS WILL BE LAST SEEK
5457 020354 012765 000377 000002  MOV      #377,P.CYLN(R5) ;SET CYL = 377
5458 020362 043765 005514 000002  BIC      CYLNDR,P.CYLN(R5) ;ZERO A BIT TO PROMOTE CROSSTALK
5459 020370 004737 032374      JSR      PC,SEEKER      ;SEEK TO THIS CYLINDER
5460 020374 006337 005514      ASL      CYLNDR      ;SHIFT BIT MASK
5461 020400 000755      BR      2$      ;GO SEEK TO CYL 0 AGAIN
5462 020402 012765 000577 000002 4$:  MOV      #577,P.CYLN(R5) ;SET CYL = 577
5463 020410 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYL 577
5464
5465
5466
5467      ;*****
5468      ;TEST 5      INCREMENT/DECREMENT SEEK TEST
5469      ;
5470      ;      IN THIS TEST SEEKS ARE DONE IN INCREMENTS OF IC CYLINDERS
5471      ;      STARTING AT CYL FC, AND ENDING AT OR BEYOND LC. THEN, THE
5472      ;      SEEKS ARE DONE IN REVERSE, BACK TO FC. IF FC IS CHOSEN > LC,
5473      ;      THE SEEKS WILL PROCEED IN THE OPPOSITE DIRECTION.
5474      ;
5475      ;*****
5476 020414 000004      TS5:  SCOPE
5477 020416 012737 000005 001324  MOV      #5,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
5478 020424 004737 030462      JSR      PC,SETUP      ;SET UP FOR LOOP ON ERROR
5479 020430 004737 030530      JSR      PC,CHKITR      ;SEE IF ITER. NO. = 0 FOR THIS TEST
5480 020434 000137 020620      JMP      TS6      ;JUMP TO NEXT TEST
5481      ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5482 020440 004737 027522      JSR      PC,INITSS      ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5483 020444 004737 026040      JSR      PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT
5484      ;SEEK IN INCREMENTS, FROM FC TO LC
5485 020450 013737 005732 005522  MOV      FC,NCYL1      ;INITIALIZE NCYL1 TO FC
5486 020456 013765 005522 000002 2$:  MOV      NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5487 020464 004737 032374      JSR      PC,SEEKER      ;SEEK TO NCYL1
5488 020470 023737 005732 005734  CMP      FC,LC      ;SEE IF FC > LC
5489 020476 003010      BGT      4$      ;BR IF FC > LC
5490 020500 063737 005736 005522  ADD      IC,NCYL1      ;INCREMENT NCYL1
5491 020506 023737 005734 005522  CMP      LC,NCYL1      ;SEE IF LC EXCEEDED
5492 020514 002360      BGE      2$      ;BR IF NOT YET
5493 020516 000407      BR      6$      ;LC REACHED - DO REVERSE SEEKS
5494 020520 163737 005736 005522 4$:  SUB      IC,NCYL1      ;DECREMENT NCYL1
5495 020526 023737 005734 005522  CMP      LC,NCYL1      ;SEE IF LC EXCEEDED
5496 020534 003750      BLE      2$      ;BR IF NOT YET
5497      ;SEEK IN INCREMENTS, BACK TO FC
5498 020536 023737 005732 005734 6$:  CMP      FC,LC      ;SEE IF FC > LC
5499 020544 003010      BGT      8$      ;BR IF FC > LC

```

```

5500 020546 163737 005736 005522 SUB IC,NCYL1 ;DECREMENT NCYL1
5501 020554 023737 005732 005522 CMP FC,NCYL1 ;SEE IF FC EXCEEDED
5502 020562 003410 BLE 10$ ;BR IF NOT YET
5503 020564 000415 BR 12$ ;FC REACHED - ALL DONE
5504 020566 063737 005736 005522 8$: ADD IC,NCYL1 ;INCREMENT NCYL1
5505 020574 023737 005732 005522 CMP FC,NCYL1 ;SEE IF FC EXCEEDED
5506 020602 002406 BLT 12$ ;FC REACHED - ALL DONE
5507 020604 013765 005522 000002 10$: MOV NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5508 020612 004737 032374 JSR PC,SEEKER ;SEEK TO NCYL1
5509 020616 000747 BR 6$ ;CONTINUE
5510 020620 12$:
5511
5512
5513 *****
5514 *TEST 6 OSCILLATING SEEK TEST
5515 *
5516 * IN THIS TEST, SEEKS OF INCREASING LENGTH ARE DONE FROM FC
5517 * TO NCYL1, AND BACK TO FC. NCYL1 IS INCREMENTED BY IC AFTER
5518 * EACH SEEK, AND THE SEEKS TO NCYL1 ALWAYS PROCEED FROM FC
5519 * TOWARD LC (FC > LC IS PERMISSIBLE). WHEN LC IS REACHED OR
5520 * EXCEEDED, THE SAME SEEKS ARE DONE IN REVERSE ORDER, UNTIL
5521 * NCYL1 EXCEEDS FC.
5522 *
5523 *****
5524 020620 000004 TS6: SCOPE
5525 020622 012737 _J0006 001324 MOV #6,STESTN ;SET TEST NUMBER IN APT MAIL BOX
5526 020630 004737 030462 JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
5527 020634 004737 030530 JSR PC,CHKITR ;SEE IF ITER. NO. = 0 FOR THIS TEST
5528 020640 000137 021050 JMP TS7 ;JUMP TO NEXT TEST
5529 ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5530 020644 004737 027522 JSR PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5531 020650 004737 026040 JSR PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5532 ;DO OSCILLATING SEEKS TOWARD LC
5533 020654 013737 005732 005522 MOV FC,NCYL1 ;INITIALIZE NCYL1 TO FC
5534 020662 013765 005522 000002 2$: MOV NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5535 020670 004737 032374 JSR PC,SEEKER ;SEEK TO NCYL1
5536 020674 013765 005732 000002 MOV FC,P.CYLN(R5) ;SET CYL = FC
5537 020702 004737 032374 JSR PC,SEEKER ;SEEK TO FC
5538 020706 023737 005732 005734 CMP FC,LC ;SEE IF FC > LC
5539 020714 003010 BGT 4$ ;BR IF FC > LC
5540 020716 063737 005736 005522 ADD IC,NCYL1 ;INCREMENT NCYL1
5541 020724 023737 005734 005522 CMP LC,NCYL1 ;SEE IF LC EXCEEDED
5542 020732 002353 BGE 2$ ;BR IF NOT YET
5543 020734 000407 BR 6$ ;LC REACHED - DO REVERSE SEEKS
5544 020736 163737 005736 005522 4$: SUB IC,NCYL1 ;DECREMENT NCYL1
5545 020744 023737 005734 005522 CMP LC,NCYL1 ;SEE IF LC EXCEEDED
5546 020752 003743 BLE 2$ ;BR IF NOT YET
5547 ;DO OSCILLATING SEEKS TOWARD FC
5548 020754 023737 005732 005734 6$: CMP FC,LC ;SEE IF FC > LC
5549 020762 003010 BGT 8$ ;BR IF FC > LC
5550 020764 163737 005736 005522 SUB IC,NCYL1 ;DECREMENT NCYL1
5551 020772 023737 005732 005522 CMP FC,NCYL1 ;SEE IF FC EXCEEDED
5552 021000 003410 BLE 10$ ;BR IF NOT YET
5553 021002 000422 BR 12$ ;FC REACHED - ALL DONE
5554 021004 063737 005736 005522 8$: ADD IC,NCYL1 ;INCREMENT NCYL1
5555 021012 023737 005732 005522 CMP FC,NCYL1 ;SEE IF FC EXCEEDED

```

```

5556 021020 002413          BLT      12$          ;FC REACHED - ALL DONE
5557 021022 013765 005522 000002 10$: MOV      NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5558 021030 004737 032374          JSR      PC,SEEKER ;SEEK TO NCYL1
5559 021034 013765 005732 000002      MOV      FC,P.CYLN(R5) ;SET CYL = FC
5560 021042 004737 032374          JSR      PC,SEEKER ;SEEK TO FC
5561 021046 000742          BR        6$          ;CONTINUE
5562 021050          12$:

```

```

5563
5564
5565 *****
5566 *TEST 7 CONVERGING/DIVERGING SEEK TEST
5567 *

```

```

5568 * THIS TEST PERFORMS SEEKS WHICH EXERCISE CYLINDER DIFFERENCE
5569 * VALUES. INITIALLY, NCYL1 = FC AND NCYL2 = LC. A SEEK IS
5570 * DONE TO NCYL1 AND THEN TO NCYL2. THEN, NCYL1 IS INCREMENTED
5571 * BY IC IN THE DIRECTION OF LC, AND NCYL2 IS INCREMENTED BY
5572 * IC IN THE DIRECTION OF FC, AND SEEKS ARE DONE TO NCYL1
5573 * AND NCYL2 AGAIN. THIS SEEKING IS CONTINUED WITH INCREMENTED
5574 * VALUES OF NCYL1 AND NCYL2, UNTIL NCYL1 EXCEEDS LC AND NCYL2
5575 * EXCEEDS FC. (NOTE : FC > LC IS PERMISSIBLE.)
5576 * THIS TEST CAUSES SEEKING TO CONVERGING AND THEN DIVERGING
5577 * CYLINDER VALUES.
5578 *

```

```

5579 *****
5580 *TEST7: SCOPE
5581 021050 000004          MOV      #7,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
5582 021052 012737 000007 001324      JSR      PC,SETUP ;SET UP FOR LOOP ON ERROR
5583 021060 004737 030462          JSR      PC,CHKITR ;SEE IF ITER. NO. = 0 FOR THIS TEST
5584 021064 004737 030530          JMP      TST10 ;JUMP TO NEXT TEST
5585 021070 000137 021226          ;RETURN HERE FROM CHKIT? IF TEST SHOULD BE RUN
5586 021074 004737 027522          JSR      PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5587 021100 004737 026040          JSR      PC,PREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5588 021104 013737 005732 005522      MOV      FC,NCYL1 ;SET NCYL1 = FC
5589 021112 013737 005734 005524      MOV      LC,NCYL2 ;SET NCYL2 = LC
5590 021120 013765 005522 000002 2$: MOV      NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5591 021126 004737 032374          JSR      PC,SEEKER ;SEEK TO NCYL1
5592 021132 013765 005524 000002      MOV      NCYL2,P.CYLN(R5) ;SET CYL = NCYL2
5593 021140 004737 032374          JSR      PC,SEEKER ;SEEK TO NCYL2
5594 021144 023737 005732 005734      CMP      FC,LC ;SEE IF FC > LC
5595 021152 003013          BGT      4$          ;BR IF FC > LC
5596 021154 063737 005736 005522      ADD      IC,NCYL1 ;INCREMENT NCYL1
5597 021162 163737 005736 005524      SUB      IC,NCYL2 ;DECREMENT NCYL2
5598 021170 023737 005734 005522      CMP      LC,NCYL1 ;SEE IF LIMITS EXCEEDED
5599 021176 002350          BGE      2$          ;BR IF NOT YET
5600 021200 000412          BR        6$          ;LIMITS REACHED - ALL DONE
5601 021202 163737 005736 005522 4$: SUB      IC,NCYL1 ;DECREMENT NCYL1
5602 021210 063737 005736 005524      ADD      IC,NCYL2 ;INCREMENT NCYL2
5603 021216 023737 005734 005522      CMP      LC,NCYL1 ;SEE IF LIMITS EXCEEDED
5604 021224 003735          BLE      2$          ;BR IF NOT YET
5605 021226          6$:

```

```

5606
5607
5608 *****
5609 *TEST 10 PSEUDO-RANDOM SEEK TEST
5610 *
5611

```

```

5612      ;*      THIS TEST PERFORMS A SEEK TO A PSEUDO-RANDOMLY CHOSEN
5613      ;*      CYLINDER, WHICH IS WITHIN THE RANGE (0-632).
5614      ;*
5615      ;*****
5616      021226 000004      ;ST10: SCOPE
5617      021230 012737 000010 001324      MOV      #10,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX
5618      021236 004737 030462      JSR      PC,SETUP      ;:SET UP FOR LOOP ON ERROR
5619      021242 004737 030530      JSR      PC,CHKITR      ;:SEE IF ITER. NO. = 0 FOR THIS TEST
5620      021246 000137 021300      JMP      TST11      ;:JUMP TO NEXT TEST
5621      ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5622      021252 004737 027522      JSR      PC,INITSS      ;:INITIALIZE DRIVER PARAMS AND SUB-SYS
5623      021256 004737 026040      JSR      PC,PREPKB      ;:PREPARE FOR POSSIBLE KBD INPUT
5624      021262 004737 032554      JSR      PC,RNDADR      ;:SELECT A PSEUDO-RANDOM CYLINDER
5625      021266 013765 005514 000002      MOV      CYLNDR,P.CYLN(R5) ;:SET CYL VALUE
5626      021274 004737 032374      JSR      PC,SEEKER      ;:SEEK TO THIS CYLINDER
5627
5628
5629
5630      ;*****
5631      ;TEST 11      MAXIMUM VELOCITY REVERSAL SEEK TEST
5632      ;*
5633      ;*      THIS TEST PERFORMS A SEEK TO CYLINDER 0, FOLLOWED BY A
5634      ;*      SEEK TO CYL 201 (OCT), AND THEN A RETURN SEEK TO 0. THIS
5635      ;*      OPERATION REQUIRES THE HEADS TO DECELERATE UPON REACHING
5636      ;*      THE POINT OF MAXIMUM VELOCITY, AND INDUCES HEATING IN
5637      ;*      THE SERVO MECHANISM.
5638      ;*
5639      ;*****
5640      021300 000004      ;ST11: SCOPE
5641      021302 012737 000011 001324      MOV      #11,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX
5642      021310 004737 030462      JSR      PC,SETUP      ;:SET UP FOR LOOP ON ERROR
5643      021314 004737 030530      JSR      PC,CHKITR      ;:SEE IF ITER. NO. = 0 FOR THIS TEST
5644      021320 000137 021366      JMP      TST12      ;:JUMP TO NEXT TEST
5645      ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5646      021324 004737 027522      JSR      PC,INITSS      ;:INITIALIZE DRIVER PARAMS AND SUB-SYS
5647      021330 004737 026040      JSR      PC,PREPKB      ;:PREPARE FOR POSSIBLE KBD INPUT
5648      021334 005065 000002      CLR      P.CYLN(R5)      ;:SET CYL = 0
5649      021340 004737 032374      JSR      PC,SEEKER      ;:SEEK TO CYL 0
5650      021344 012765 000201 000002      MOV      #201,P.CYLN(R5) ;:SET CYL = 201 (OCT)
5651      021352 004737 032374      JSR      PC,SEEKER      ;:SEEK TO CYL 201
5652      021356 005065 000002      CLR      P.CYLN(R5)      ;:SET CYL = 0
5653      021362 004737 032374      JSR      PC,SEEKER      ;:SEEK TO CYL 0
5654
5655
5656
5657      ;*****
5658      ;TEST 12      MECHANICAL VIBRATION SEEK TEST
5659      ;*
5660      ;*      THIS TEST PERFORMS GEOMETRICALLY INCREASING SEEKS WITH
5661      ;*      GEOMETRICALLY DECREASING STALL TIME BETWEEN OPERATIONS
5662      ;*      WITH INTENT TO INDUCE VARYING VIBRATIONAL MODES UPON
5663      ;*      THE DRIVE. THE TEST BEGINS WITH LC = 1, AND ST = SM. THEN,
5664      ;*      THE FOLLOWING SEQUENCE IS PERFORMED :
5665      ;*      SEEKS ARE DONE BETWEEN 0 AND LC, TEN TIMES. THEN, ST IS
5666      ;*      DIVIDED BY 2 AND LC IS DOUBLED, AND SEEKS ARE DONE BETWEEN
5667      ;*      0 AND LC AGAIN, TEN TIMES. THIS PROCESS IS CONTINUED FOR NEW

```

F09

```

5668      *      VALUES OF ST AND LC, UNTIL LC EXCEEDS CYL 400 (OCT). THEN,
5669      *      THE WHOLE PROCESS IS REVERSED, WITH ST BEING DOUBLED AND
5670      *      LC DIVIDED BY 2, UNTIL LC BECOMES < 1.
5671      *
5672      *
5673      *****
5673      021366 000004      ST12: SCOPE
5674      021370 0.2737 000012 001324      MOV      #12,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
5675      02.376 004737 030462      JSR      PC,SETUP      ;SET UP FOR LOOP ON ERROR
5676      021402 004737 030530      JSR      PC,CHKITR      ;SEE IF ITER. NO. = 0 FOR THIS TEST
5677      021406 000137 021626      JMP      TS13      ;JUMP TO NEXT TEST
5678      ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5679      021412 004737 027522      JSR      PC,INITSS      ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5680      021416 004737 026040      JSR      PC,PREPKB      ;PREPARE FOR POSSIBLE KBD INPUT
5681      021422 012737 000001 005514      MOV      #1,CYLNDR      ;INIT. CURRENT CYL TO 1
5682      021430 013737 005774 001276      MOV      SM,$TMP6      ;INIT. STALLS TO SM
5683      ;DO INCREASING SEEKS
5684      021436 012737 000012 005542      2$: MOV      #10,$SCRACH      ;INIT. LOOP COUNTER TO 10(DEC)
5685      021444 013737 001276 005512      MOV      $TMP6,STALLS      ;SET CURRENT STALL IN STALL WORD
5686      021452 005065 000002      4$: CLR      P,CYLND(R5)      ;SET CYL = 0
5687      021456 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYL 0
5688      021462 013765 005514 000002      MOV      CYLNDR,P,CYLND(R5) ;SET CURRENT CYLINDER
5689      021470 004737 032374      JSR      PC,SEEKER      ;SEEK TO CURRENT CYLINDER
5690      021474 005337 005542      DEC      $SCRACH      ;DECREMENT LOOP COUNTER
5691      021500 001364      BNE      4$      ;BR IF NOT 5 TIMES YET
5692      021502 000241      CLC      ;CLEAR GARBAGE BIT
5693      021504 006037 001276      ROR      $TMP6      ;DIVIDE STALLS BY 2
5694      021510 006037 001300      ROR      $TMP7      ;SAVE THE DROPPED STALL BIT
5695      021514 006337 005514      ASL      CYLNDR      ;DOUBLE THE CURRENT CYLINDER NUMBER
5696      021520 032737 001000 005514      BIT      #BIT09,CYLNDR      ;SEE IF DONE WITH INCREASING SEEKS YET
5697      021526 001743      BEQ      2$      ;BR IF NOT DONE YET
5698      ;DO DECREASING SEEKS
5699      021530 006237 005514      6$: ASR      CYLNDR      ;DIVIDE CURRENT CYLINDER BY 2
5700      021534 006137 001300      ROL      $TMP7      ;GET A SAVED STALL BIT INTO CARRY
5701      021540 006137 001276      ROL      $TMP6      ;DOUBLE THE STALL
5702      021544 012737 000012 005542      MOV      #10,$SCRACH      ;INIT. LOOP COUNTER TO 10(DEC)
5703      021552 013737 001276 005512      MOV      $TMP6,STALLS      ;SET CURRENT STALL IN STALL WORD
5704      021560 005065 000002      8$: CLR      P,CYLND(R5)      ;SET CYL = 0
5705      021564 004737 032374      JSR      PC,SEEKER      ;SEEK TO CYL 0
5706      021570 013765 005514 000002      MOV      CYLNDR,P,CYLND(R5) ;SET CURRENT CYL NUMBER
5707      021576 004737 032374      JSR      PC,SEEKER      ;SEEK TO CURRENT CYLINDER
5708      021602 005337 005542      DEC      $SCRACH      ;DECREMENT LOOP COUNTER
5709      021606 001364      BNE      8$      ;BR IF NOT 5 TIMES YET
5710      021610 032737 000002 005514      BIT      #BIT01,CYLNDR      ;SEE IF DONE YET
5711      021616 001744      BEQ      6$      ;BR IF NOT DONE YET
5712      021620 013737 005772 005512      MOV      ST,STALLS      ;RESTORE DESIRED NO. OF UNIT STALLS
5713
5714
5715
5716      *****
5717      *TEST 13      MAX ROTATIONAL LATENCY MEASUREMENT
5718      *THIS TEST MEASURES THE INTERVAL OF TIME BETWEEN 2 INDEX MARKS
5719      *IN A SINGLE DISK ROTATION. THE SPECIFIED MAXIMUM ROTATIONAL
5720      *LATENCY = 25 MILLI-SEC + OR - 2.5%. THIS MEASUREMENT IS MADE
5721      * 128 TIMES.
5722      *****
5723      021626 000004      ST13: SCOPE

```

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 111
T13 MAX ROTATIONAL LATENCY MEASUREMENT

SEQ 0389
SEQ 0110

```

5724 021630 012737 000013 001324 MOV #13,STESTN ;SET TEST NUMBER IN APT MAIL BOX
5725 021636 004737 030462 JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
5726 021642 004737 030530 JSR PC,CHKITR ;SEE IF ITER. NO. = 0 FOR THIS TEST
5727 021646 000137 022440 JMP TST14 ;JUMP TO NEXT TEST
5728 ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5729 021652 004737 027522 JSR PC,INITSS ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5730 021656 004737 026040 JSR PC,FREPKB ;PREPARE FOR POSSIBLE KBD INPUT
5731 021662 105737 003133 TSTB DOIM ;SEE IF TIMING TESTS ARE ALLOWED
5732 021666 001002 BNE 26$ ;BR IF TEST ALLOWED
5733 021670 000177 000004 JMP 21$ ;SKIP TO NEXT TEST
5734 021674 004737 032712 26$: JSR PC,CALBRT ;CALIBRATE SOFTWARE TIMER
5735 021700 1$: TST14 ;ERROR RETURN ADDR FOR CALBRT
5736 021700 022440 JSR PC,INIVRB ;INIT VARIABLES TO BE USED
5737 021702 004737 033204 CLR TSTING ;INHIBIT TC, TZ ESCAPE
5738 021706 105037 003120 CLR R3 ;INIT MEASUREMENT COUNTER
5739 021712 005003 ;READ HEADER TO INITIALIZE FORMAT BIT IN CONTROLLER
5740 BICB #B.CFMT,P.CS1H(R5) ;SET FORMAT BIT = 0
5741 021714 142765 000020 000007 MOV #RDHEAD,P.CMND(R5) ;SET READ HDR COMMAND
5742 021722 112765 000125 000001 JSR PC,DRVCAL ;DO READ HDR TO SET FORMAT TO 0
5743 021730 004737 040316 MOV #1,TSTING ;ALLOW TC, TZ ESCAPE
5744 021734 112737 000001 003120 ;CHANGE FORMAT AND READ HDR TO FIND INDEX
5745 2$: CLR R0 ;INIT RK06 INTR INDICATOR
5746 021742 005000 MOV #RHEDHD,ARKVEC ;SET SPECIAL RK06 HANDLER ADDRESS
5747 021744 012777 022422 161066 CLR R1 ;INIT TIME-OUT INDICATOR
5748 021752 005001 BICB #B.CFMT,P.CS1H(R5) ;COMPLEMENT FORMAT BIT TO 1
5749 021754 152765 000020 000007 BIS #CFMT,P.CS1(R5) ;COMPLEMENT FORMAT BIT TO 1
5750 021762 052765 010000 000016 JSR PC,STRCMD ;STORE PREV AND CURRENT COMMANDS
5751 021770 004737 040606 MOV HOLD5,(R2) ;ISSUE RD HDR WITH FMT BIT = 1
5752 021774 013712 017774 6$: TST R0 ;SEE IF RK06 INTR REC'D YET
5753 022000 005700 BNE 8$ ;BR IF RK06 INTR REC'D
5754 022002 001012 INC R1 ;INCREMENT TIME-OUT INDICATOR
5755 022004 005201 BNE 6$ ;BR IF NO TIME-OUT
5756 022006 001374 7$: JSR PC,REPSUP ;GET COMMAND FOR REPORT
5757 022010 004737 041704 JSR PC,TOPROC ;GATHER STATUS
5758 022014 004737 044352 ERROR 10$ ;TIMED-OUT ON READ HEADER
5759 022020 104105 BR 36$ ;GO TO EXIT
5760 022022 000572 28$: RESREG 7$ ;RESTORE R0-R5
5761 022024 104410 BR 7$ ;TAKE ERROR EXIT
5762 022026 000770 ;CHANGE FORMAT AND READ HDR TO MEASURE TIME TO NEXT INDEX
5763 8$: CLR R0 ;INIT RK06 INTERRUPT INDICATOR
5764 022030 005000 CLR R4 ;INIT HI CYCLE COUNT
5765 022032 005004 MOV R5,R1 ;SAVE PARAM BLK ADDRESS
5766 022034 010501 MOV R5,SCRACH
5767 022036 010537 005542 CLR R5
5768 022042 005005 BICB #B.CFMT,P.CS1H(R1) ;COMPLEMENT FORMAT BIT TO 0
5769 022044 142761 000020 000007 BIC #CFMT,P.CS1(R1) ;COMPLEMENT FORMAT BIT TO 0
5770 022052 042761 010000 000016 MOV P.CS1(R1),RKCS1(R2) ;ISSUE READ HDR TO FIND NEXT INDEX
5771 022060 016162 000016 000000 JSR PC,TIMER ;MEASURE THE TIME TO NEXT INDEX
5772 022066 004737 033150 7$ ;ERROR RETURN ADDRESS
5773 022072 022010 ;CONVERT MEAS'D CYCLES TO TIME AND CHECK AGAINST LIMITS
5774 SAVREG ;SAVE R0-R5
5775 022074 104407 JSR PC,CNVTIM ;CONVERT MEAS'D CYCLES TO TIME
5776 022076 004737 034136 28$ ;ERROR RETURN ADDRESS
5777 022102 022024 ADD R3,SUMLO1 ;ADD CONVERTED TIME TO SUM
5778 022104 060337 003176 ADC SUMHI1
5779 022110 005537 003200

```

CZR6M00 RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 112
T13 MAX ROTATIONAL LATENCY MEASUREMENT

SEQ 0390
SEQ 0111

5780	022114	060237	003200	ADD	R2,SUMHI1	
5781	022120	004737	036332	JSR	PC,CTLOUT	:CHECK FOR (1C) OR (1Z) KBD INPUT
5782				;COMPARE R2-R3 AGAINST CURRENT MIN AND MAX MEASUREMENTS		
5783	022124	020237	003214	CMP	R2,MINIH1	:COMP HI BITS TO HI MIN
5784	022130	101010		BHI	32\$	:BR IF > MIN
5785	022132	103403		BLO	30\$	:BR IF < MIN
5786	022134	020337	003212	CMP	R3,MINIL1	:COMP LO BITS TO LO MIN
5787	022140	103004		BHIS	32\$	:BR IF > OR = MIN
5788	022142	010237	003214	30\$: MOV	R2,MINIH1	:SET NEW MIN
5789	022146	010337	003212	MOV	R3,MINIL1	
5790	022152	020237	003220	32\$: CMP	R2,MAXIH1	:COMP HI BITS TO HI MAX
5791	022156	103410		BLO	14\$	:BR IF < MAX
5792	022160	101003		BHI	34\$	:BR IF > MAX
5793	022162	020337	003216	CMP	R3,MAXIL1	:COMP LO BITS TO LO MAX
5794	022166	101404		BLOS	14\$	:BR IF < OR = MAX
5795	022170	010237	003220	34\$: MOV	R2,MAXIH1	:SET NEW MAX
5796	022174	010337	003216	MOV	R3,MAXIL1	
5797				;COMPARE MEAS'D TIME TO SPECIFIED LIMITS		
5798	022200	020227	000000	14\$: CMP	R2,#MINHI1	:COMPARE HI BITS TO HI MINIMUM
5799	022204	101006		BHI	18\$	:BR IF > MIN
5800	022206	103403		BLO	16\$	:BR IF < MIN
5801	022210	020327	057467	CMP	R3,#MINLO1	:COMPARE LO BITS TO LO MIN
5802	022214	103002		BHIS	18\$	:BR IF > OR = MIN
5803	022216	005237	003170	16\$: INC	BLWMIN	:INCREMENT COUNT OF TIMES BELOW MIN
5804	022222	020227	000000	18\$: CMP	R2,#MAXHI1	:COMPARE HI BITS TO HI MAX
5805	022226	103406		BLO	22\$	:BR IF < MAX
5806	022230	101003		BHI	20\$	:BR IF > MAX
5807	022232	020327	062031	CMP	R3,#MAXLO1	:COMPARE LO BITS TO LO MAX
5808	022236	101402		BLOS	22\$	:BR IF < OR = MAX
5809	022240	005237	003172	20\$: INC	ABVMX1	:INCREMENT COUNT OF TIMES ABOVE MAX
5810	022244	104410		22\$: RESREG		:RESTORE R0-R5
5811	022246	005203		INC	R3	:INCREMENT MEASUREMENT COUNTER
5812	022250	013705	005542	MOV	SCRACH,R5	:RESTORE PARAM BLK ADDRESS
5813	022254	022703	000200	CMP	#128,R3	:SEE IF 128 MEASUREMENTS YET
5814	022260	001230		BNE	2\$	:BR IF NOT DONE WITH MEASUREMENTS YET
5815				;DIVIDE SUM BY 128. TO GET AVERAGE TIME OF ROTATION		
5816	022262	012704	000007	MOV	#7,R4	:SET LOOP COUNTER = 7
5817	022266	000241		24\$: CLC		:DIVIDE DOUBLE-WORD SUM BY 2
5818	022270	006037	003200	ROR	SUMHI1	
5819	022274	006037	003176	ROR	SUMLO1	
5820	022300	005304		DEC	R4	:SEE IF DONE WITH DIVISION
5821	022302	001371		BNE	24\$	:BR IF NOT DONE YET
5822				;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)		
5823	022304	032737	000010	BIT	#BIT03,CS	:SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5824	022312	001036		BNE	36\$	:BR IF REPORTS SHOULD BE INHIBITED
5825	022314	104401	011577	TYPE	,ROTIMS	:TYPE "ROTATIONAL TIMES :"
5826	022320	012701	003212	MOV	#MINIL1,R1	:POINTER TO LO MIN
5827	022324	004737	034210	JSR	PC,TYPMIN	:TYPE MIN MEAS'D TIME
5828	022330	013746	003170	MOV	BLWMIN,-(SP)	:GET COUNT OF TIMES BELOW MIN
5829	022334	104405		TYPDS		:CONVERT AND TYPE IT
5830	022336	104401	012007	TYPE	,BELOW	:TYPE " OF 128 BELOW MIN OF "
5831	022342	104401	012334	TYPE	,LIM1	:TYPE SPEC'D MIN LIMIT
5832	022346	012701	003216	MOV	#MAXIL1,R1	:POINTER TO LO MAX
5833	022352	004737	034234	JSR	PC,TYPMAX	:TYPE MAX MEAS'D TIME
5834	022356	013746	003172	MOV	ABVMX1,-(SP)	:GET COUNT OF TIMES ABOVE MAX
5835	022362	104405		TYPDS		:CONVERT AND TYPE IT

CZR6MDO RK611/06 SS VERIF 1
CZR6MDO.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 113
T13 MAX ROTATIONAL LATENCY MEASUREMENT

SEQ 0391
SEQ 0112

5836 022364 104401 012044
5837 022370 104401 012347
5838 022374 012701 003176
5839 022400 004737 034260
5840 022404 104401 001315
5841 022410 012777 045270
5842 022416 000177 177256
5843
5844
5845
5846
5847 022422 005200
5848 022424 005762 000000
5849 022430 100002
5850 022432 000137 045270
5851 022436 000002
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864 022440 000004
5865 022442 012737 000014 001324
5866 022450 004737 030462
5867 022454 004737 030530
5868 022460 000137 022760
5869
5870 022464 004737 027522
5871 022470 004737 026040
5872 022474 105737 003133
5873 022500 001002
5874 022502 000177 000004
5875 022506 004737 032712
5876 022512
5877 022512 022760
5878 022514 004737 033204
5879 022520 112765 000117 000001
5880 022526 005065 000002
5881 022532 004737 040316
5882
5883 022536 005265 000002
5884 022542 004737 033312
5885 022546 022760
5886 022550 060137 003176
5887 022554 005537 003200
5888 022560 060037 003200
5889 022564 004737 036332
5890
5891 022570 012703 003212

```

      TYPE      ,ABOVE      ;TYPE " CF 128 ABOVE MAX OF "
      TYPE      ,LIM2       ;TYPE SPEC'D MAX LIMIT
      MOV       ,SUMLO1,R1   ;POINTER TO AVG TIME
      JSR       PC,TYPAVG   ;TYPE AVERAGE TIME
      TYPE      ,SCRLF      ;TYPE <CR>,<LF>
160422 36$: MOV      ,I.INTR,ARKVEC ;RESTORE RK06 HANDLER ADDRESS
      JMP       ,21$        ;PROCEED TO NEXT TEST

;*****
;SPECIAL INTERRUPT HANDLER FOR READ HEADERS IN ROT. LAT. MEAS. TEST
;*****
RHEDHD: INC      R0          ;SET RK06 INTR HANDLER
      TST      RKCS1(R2)    ;SEE IF CONTROLLER ERROR SET
      BPL      2$          ;BR IF NO ERROR
      JMP      I.INTR      ;LET DRIVER PROCESS THE ERROR
2$:      RTI              ;RETURN FROM INTR

;*****
;TEST 14 ONE CYLINDER SEEK TIME MEASUREMENT
;THIS TEST MEASURES THE TIME REQUIRED TO SEEK BETWEEN 2 ADJACENT
;CYLINDERS, BOTH IN THE FORWARD AND REVERSE DIRECTIONS. SEEKS
;ARE DONE TO 0,1,2,...,631,632 AND THEN TO 631,630,...,2,1,0
;AND THE RESULTS ARE TYPED FOR EACH DIRECTION.
;THE SPECIFIED ONE CYL SEEK TIME IS < 8 MILLI-SEC.
;SEEKS ARE TO & FROM CYL 1456 FOR THE RK07
;*****
TST14: SCOPE
      MOV      #14,$TESTN   ;SET TEST NUMBER IN APT MAIL BOX
      JSR      PC,$SETUP    ;SET UP FOR LOOP ON ERROR
      JSR      PC,CHKITR    ;SEE IF ITER. NO. = 0 FOR THIS TEST
      JMP      TST15        ;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
      TSTB     ,DO$IM       ;SEE IF TIMING TESTS ARE ALLOWED
      BNE      2$          ;BR IF ALLOWED
1$:      JMP      ,34$       ;SKIP TO NEXT TEST
2$:      JSR      PC,CALBRT  ;CALIBRATE THE SOFTWARE TIMER
4$:      TST15             ;ERROR RETURN ADDR FOR CALBRT
      JSR      PC,INIVRB    ;INIT VARIABLES USED
      MOVB     ,#SEEK,P.CMND(R5) ;SET SEEK COMMAND
      CLR      P.CYLN(R5)   ;INIT CYL TO 0
      JSR      PC,DRVCAL    ;DO INITIAL SEEK TO CYL 0
;MEASURE FORWARD SEEK TIME
6$:      INC      P.CYLN(R5) ;INCREMENT CYLINDER BY 1
      JSR      PC,TIMSEK    ;MEASURE A SEEK TIME
      TST15             ;ERROR RETURN ADDR FOR CALBRT
      ADD      R1,SUMLO1    ;ADD MEAS'D TIME TO FORWARD SUM
      ADC      SUMHI1
      ADD      R0,SUMHI1
      JSR      PC,CTLOUT    ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
;COMPAE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
      MOV      ,#MINI1,R3  ;SET POINTER TO CURRENT MIN AND MAX

```

J09

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 114
T14 ONE CYLINDER SEEK TIME MEASUREMENTSEQ 0392
SEQ 0113

```

5892 022574 004737 033624      JSR      PC,CMPTIM      ;COMPARE TO MEAS'D MIN AND MAX
5893                                ;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
5894 022600 020037 007220      CMP      R0,MAXHI2      ;COMPARE HI BITS TO HI MAX
5895 022604 103406              BLO      12$              ;BR IF < MAX
5896 022606 101003              BHI      10$              ;BR IF > MAX
5897 022610 020137 007216      CMP      R1,MAXLO2      ;COMPARE LO BITS TO LO MAX
5898 022614 101402              BLOS     12$              ;BR IF < OR = MAX
5899 022616 005237 003172      10$: INC      ABVMX1      ;INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
5900 022622 023765 017760      12$: CMP      LSTCYL,P.CYLN(R5) ;SEE IF LAST FORWARD SEEK DONE
5901 022630 001342              BNE      6$              ;BR IF NOT DONE
5902                                ;MEASURE A REVERSE SEEK TIME
5903 022632 005365 000002      14$: DEC      P.CYLN(R5)      ;DECREMENT CYL BY 1
5904 022636 004737 033312      JSR      PC,TIMSEK      ;MEASURE A SEEK TIME
5905 022642 022760              TST15              ;ERROR RETURN ADDR FOR CALBRT
5906 022644 060137 003202      ADD      R1,SUML02      ;ADD MEASURED TIME TO REVERSE SUM
5907 022650 005537 003204      ADC      SUMHI2
5908 022654 060037 003204      ADD      R0,SUMHI2
5909 022660 004737 036332      JSR      PC,CTLOUT
5910                                ;COMPARE REVERSE TIME TO CURRENT REVERSE MIN AND MAX
5911 022664 012703 003222      MOV      #MINIL2,R3      ;SET POINTER TO CURRENT MIN AND MAX
5912 022670 004737 033624      JSR      PC,CMPTIM      ;COMPARE TO MEAS'D MIN AND MAX
5913                                ;COMPARE REVERSE TIME TO SPEC'D MAX LIMIT
5914 022674 020037 007220      CMP      R0,MAXHI2      ;COMPARE HI BITS TO HI MAX
5915 022700 103406              BLO      18$              ;BR IF < MAX
5916 022702 101003              BHI      16$              ;BR IF > MAX
5917 022704 020137 007216      CMP      R1,MAXLO2      ;COMPARE LO BITS TO LO MAX
5918 022710 101402              BLOS     18$              ;BR IF < OR = MAX
5919 022712 005237 003174      16$: INC      ABVMX2      ;INCR COUNT OF TIMES ABOVE SPEC'D MAX
5920 022716 005765 000002      18$: TST      P.CYLN(R5)      ;SEE IF LAST REVERSE SEEK DONE
5921 022722 001343              BNE      14$              ;BR IF NOT DONE YET
5922                                ;COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES
5923 022724 013703 017760      MOV      LSTCYL,R3      ;GET NO. OF MEASUREMENTS
5924 022730 004737 033676      JSR      PC,GETAVG      ;COMPUTE AVERAGES
5925                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
5926 022734 032737 000010      BIT      #BIT03,CS      ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5927 022742 001257              BNE      1$              ;BR IF REPORTS SHOULD BE INHIBITED
5928 022744 104401 011644      TYPE      ONECYL      ;TYPE "ONE CYL SEEK TIMES"
5929 022750 012703 012101      MOV      #ABOVE1,R3      ;GET POINTER TO MAX SPEC'D LIMIT MSG
5930 022754 004737 033770      JSR      PC,TYPTMS      ;TYPE TIMING TEST RESULTS
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946 022760 000004
5947 022762 012737 000015 001324      TST15: SCOPE
                                         MOV      #15,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX

```

*TEST 15 AVERAGE SEEK TIME MEASUREMENT
*THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD
*AND REVERSE DIRECTIONS. THE AVERAGE TIME IS CALCULATED FROM THE
*FOLLOWING FORMULA :
* $T_{AVG} = [T1(410)(2) + T2(409)(2) + \dots + T410(1)(2)] / (410)(410)$
* WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS.
*FORWARD AND REVERSE TIMES ARE MEASURED AND TYPED,
*SEPARATELY.
*THE SPECIFIED AVERAGE SEEK TIME IS < 38 MILLI-SEC.
* CYL 814 USED FOR THE RK07

K09

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 115
T15 AVERAGE SEEK TIME MEASUREMENTSEQ 0393
SEQ 0114

5948	022770	004737	030462		JSR	PC, SETUP	; SET UP FOR LOOP ON ERROR
5949	022774	004737	030530		JSR	PC, CHKTR	; SEE IF ITER. NO. = 0 FOR THIS TEST
5950	023000	000137	023316		JMP	TST16	; JUMP TO NEXT TEST
5951					; RETURN	HERE FROM CHKTR IF TEST	SHOULD BE RUN
5952	023004	004737	027522		JSR	PC, INITSS	; INITIALIZE DRIVER PARAMS AND SUB-SYS
5953	023010	004737	026040		JSR	PC, PREPKB	; PREPARE FOR POSSIBLE KBD INPUT
5954	023014	105737	003133		TSTB	DOFIM	; SEE IF TIMING TESTS ARE ALLOWED
5955	023020	001002			BNE	2\$	; BR IF ALLOWED
5956	023022	000177	000004		JMP	24\$	; SKIP TO NEXT TEST
5957	023026	004737	032712		JSR	PC, CALBRT	; CALIBRATE THE SOFTWARE TIMER
5958	023032				4\$:		
5959	023032	023316			TST16		; ERROR RETURN ADDR FOR CALBRT
5960	023034	004737	033204		JSR	PC, INIVRB	; INIT VARIABLES
5961	023040	112765	000117	000001	MOVB	*SEEK, P. CMND(R5)	; SET SEEK COMMAND
5962	023046	005065	000002		CLR	P. CYLN(R5)	; SET CYL = 0
5963	023052	004737	040316		JSR	PC, DRVCAL	; SEEK INITIALLY TO 0
5964	023056	005037	005522		CLR	NCYL1	; SET DESTINATION CYLINDER
5965	023062	013737	017762	005542	MOV	LCPI SCRACH	; INIT COEFFICIENT TO 411(DEC)/814(DEC) FOR RK07
5966	023070	005237	005522		INC	NCYL1	; INCR DESTINATION CYL
5967	023074	005337	005542		DEC	SCRACH	; DECREMENT COEFFICIENT
5968	023100	013765	005522	000002	MOV	NCYL1, P. CYLN(R5)	
5969	023106	004737	033312		JSR	PC, TIMSEK	; MEASURE A FORWARD SEEK TIME
5970	023112	023316			TST16		; ERROR RETURN ADDR FOR CALBRT
5971	023114	004737	033514		JSR	PC, FDTERM	; COMPUTE AND ADD A FORWARD TERM
5972	023120	005065	000002		CLR	P. CYLN(R5)	; SET CYL TO 0
5973	023124	004737	033312		JSR	PC, TIMSEK	; MEASURE A REVERSE SEEK TIME
5974	023130	023316			TST16		; ERROR RETURN ADDR FOR CALBRT
5975	023132	004737	033560		JSR	PC, RVTERM	; COMPUTE AND ADD A REVERSE TERM
5976	023136	023737	005522	017760	CMP	NCYL1, LSTCYL	; SEE IF DONE MEASURING YET
5977	023144	002751			BLT	8\$	; BR IF NOT DONE YET
5978	023146	104401	011617		TYPE	AVGSEK	; TYPE "AVERAGE SEEK TIMES"
5979					; DIVIDE FORWARD	SUM BY (410)(410)/(814)(814) FOR RK07	
5980	023152	013700	003176		MOV	SUML01, R0	; GET DIVIDEND
5981	023156	013701	003200		MOV	SUMH11, R1	
5982	023162	013702	003202		MOV	SUML02, R2	
5983	023166	013703	003204		MOV	SUMH12, R3	
5984	023172	013704	017776		MOV	HOLD6, R4	; GET DIVISOR
5985	023176	013705	020000		MOV	HOLD7, R5	
5986	023202	004737	053244		JSR	PC, M. DPID	; PERFORM DIVISION
5987	023206	010237	003200		MOV	R2, SUMH11	; GET FORWARD AVG
5988	023212	010337	003176		MOV	R3, SUML01	
5989					; DIVIDE REVERSE	SUM BY (410)(410)	
5990	023216	013700	003216		MOV	MAXI1, R0	; GET DIVIDEND
5991	023222	013701	003220		MOV	MAXIH1, R1	
5992	023226	013702	003226		MOV	MAXI2, R2	
5993	023232	013703	003230		MOV	MAXIH2, R3	
5994	023236	004737	053244		JSR	PC, M. DPID	; PERFORM DIVISION
5995	023242	010237	003204		MOV	R2, SUMH12	; GET REVERSE AVG
5996	023246	010337	003202		MOV	R3, SUML02	
5997					; TYPE FORWARD AVERAGE		
5998	023252	104401	011722		TYPE	FORWARD	; TYPE "****FORWARD DIRECTION****"
5999	023256	012701	003176		MOV	SUML01, R1	; POINTER TO FORWARD AVG
6000	023262	004737	034260		JSR	PC, TYPAVG	; TYPE FORWARD AVERAGE
6001	023266	104401	012264		TYPE	SPCDMX	; TYPE "SPEC'D MAX IS 40000 US"
6002					; TYPE REVERSE AVERAGE		
6003	023272	104401	011740		TYPE	, REVRSE	; TYPE "****REVERSE DIRECTION****"

L09

CZR6MD0 RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 116
T15 AVERAGE SEEK TIME MEASUREMENT

SEQ 0394
SEQ 0115

6004 023276 012701 003202
6005 023302 004737 034260
6006 023306 104401 012264
6007 023312 004737 027522
6008
6009
6010
6011
6012
6013
6014
6015
6016
6017
6018 023316 000004
6019 023320 012737 000016 001324
6020 023326 004737 030462
6021 023332 004737 030530
6022 023336 000137 023642
6023
6024 023342 004737 027522
6025 023346 004737 026040
6026 023352 105737 003133
6027 023356 001002
6028 023360 000177 000004
6029 023364 004737 032712
6030 023370
6031 023370 023642
6032 023372 004737 033204
6033 023376 112765 000117 000001
6034 023404 005065 000002
6035 023410 004737 040316
6036 023414 005037 005542
6037
6038 023420 013765 017760 000002
6039 023426 004737 033312
6040 023432 023642
6041 023434 060137 003176
6042 023440 005537 003200
6043 023444 060037 003200
6044 023450 004737 036332
6045
6046 023454 012703 003212
6047 023460 004737 033624
6048
6049 023464 020037 007224
6050 023470 103406
6051 023472 101003
6052 023474 020137 007222
6053 023500 101402
6054 023502 005237 003172
6055
6056 023506 005065 000002
6057 023512 004737 033312
6058 023516 023642
6059 023520 060137 003202

```

MOV      #SUMLO2,R1      ; POINTER TO REVERSE AVG
JSR      PC,TYPAVG       ; TYPE REVERSE AVERAGE
TYPE     SPCDMX          ; TYPE "SPEC'D MAX IS 40000 US"
JSR      PC,INITSS       ; INIT THE SUB-SYS

;*****
;TEST 16      MAXIMUM SEEK TIME MEASUREMENT
;THIS TEST MEASURES THE TIME REQUIRED TO SEEK FROM CYL 0 TO
;CYL 632 OCT. (410 DEC.). THIS TIME REPRESENTS THE MAXIMUM SEEK
;TIME, AND IT IS MEASURED 128 TIMES IN EACH DIRECTION. THE
;THE SPECIFIED MAXIMUM SEEK TIME IS < 75 MILLI-SEC.
;CYL 814 IS THE LAST CYL FOR THE RK07
;*****
TST16:    SCOPE
MOV      #16,$TESTN      ; SET TEST NUMBER IN APT MAIL BOX
JSR      PC,SETUP        ; SET UP FOR LOOP ON ERROR
JSR      PC,CHKITR       ; SEE IF ITER. NO. = 0 FOR THIS TEST
JMP      TST17           ; JUMP TO NEXT TEST
;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
JSR      PC,INITSS       ; INITIALIZE DRIVER PARAMS AND SUB-SYS
JSR      PC,PREPKB       ; PREPARE FOR POSSIBLE KBD INPUT
TSTB     D0TIM           ; SEE IF TIMING TESTS ARE ALLOWED
BNE      2$             ; BR IF ALLOWED
1$:      JMP      24$     ; SKIP TO NEXT TEST
2$:      JSR      PC,CALBRT ; CALIBRATE THE SOFTWARE TIMER
4$:      TST17           ; ERROR RETURN ADDR FOR CALBRT
JSR      PC,INIVRB       ; INIT VARIABLES USED
MOVB     #SEEK,P.CMND(R5) ; SET SEEK COMMAND
CLR      P.CYLN(R5)      ; INIT CYL TO 0
JSR      PC,DRVCAL       ; DO INITIAL SEEK TO CYL 0
CLR      SCRACH          ; INIT MEASUREMENT COUNT
;MEASURE FORWARD SEEK TIME
6$:      MOV      LSTCYL,P.CYLN(R5) ; SET CYL = 632 OCT, 1456 FOR RK07
JSR      PC,TIMSEK       ; MEASURE A SEEK TIME
TST17    R1,SUML01       ; ERROR RETURN ADDR FOR CALBRT
ADD      SUMHI1          ; ADD MEAS'D TIME TO FORWARD SUM
ADC      R0,SUMHI1
JSR      PC,CTLOUT       ; CHECK FOR (↑C) OR (↑Z) KBD INPUT
;COMPARE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
MOV      #MINI1,R3      ; SET POINTER TO CURRENT MIN AND MAX
JSR      PC,CMPTIM       ; COMPARE TO MEAS'D MIN AND MAX
;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
CMP      R0,MAXHI4       ; COMPARE HI BITS TO HI MAX
BLO      12$             ; BR IF < MAX
BHI      10$             ; BR IF > MAX
CMP      R1,MAXLO4       ; COMPARE LO BITS TO LO MAX
BLOS     12$             ; BR IF < OR = MAX
10$:      INC      ABVMX1  ; INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
;MEASURE A REVERSE SEEK TIME
12$:      CLR      P.CYLN(R5) ; SET CYL = 0
JSR      PC,TIMSEK       ; MEASURE A SEEK TIME
TST17    R1,SUML02       ; ERROR RETURN ADDR FOR CALBRT
ADD      R1,SUML02       ; ADD MEASURED TIME TO REVERSE SUM

```

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 117
T16 MAXIMUM SEEK TIME MEASUREMENT

SEQ 0395
SEQ 0116

6060	023524	005537	003204		ADC	SUMHI2	
6061	023530	060037	003204		ADD	R0,SUMHI2	
6062	023534	004737	036332		JSR	PC,CTLOUT	;CHECK FOR (↑C) OR (↑Z) KBD INPUT
6063							REVERSE MIN AND MAX
6064	023540	012703	003222		MOV	#MINI2,R3	;SET POINTER TO CURRENT MIN AND MAX
6065	023544	004737	033624		JSR	PC,CMPITM	;COMPARE TO MEAS'D MIN AND MAX
6066							MAX LIMIT
6067	023550	020037	007224		CMP	R0,MAXHI4	;COMPARE HI BITS TO HI MAX
6068	023554	103406			BLO	18\$	BR IF < MAX
6069	023556	101003			BHI	16\$	BR IF > MAX
6070	023560	020137	007222		CMP	R1,MAXLO4	;COMPARE LO BITS TO LO MAX
6071	023564	101402			BLOS	18\$	BR IF < OR = MAX
6072	023566	005237	003174		16\$: INC	ABVMX2	;INCR COUNT OF TIMES ABOVE SPEC'D MAX
6073	023572	005237	005542		18\$: INC	SCRACH	;INCREMENT MEASUREMENT COUNT
6074	023576	022737	000200	005542	CMP	#128.,SCRACH	;SEE IF DONE WITH 128 MEASUREMENTS
6075	023604	001305			BNE	6\$	BR IF NOT DONE YET
6076							TIME AVERAGES
6077	023606	012703	000200		MOV	#128.,R3	;GET NO. OF MEASUREMENTS
6078	023612	004737	033676		JSR	PC,GETAVG	;COMPUTE AVERAGES
6079							(IN DECIMAL)
6080	023616	032737	000010	005770	BIT	#BIT03,CS	;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
6081	023624	001255			BNE	1\$	BR IF REPORTS SHOULD BE INHIBITED
6082	023626	104401	011675		TYPE	MAXSEK	;TYPE "MAXIMUM SEEK TIMES"
6083	023632	012703	012201		MOV	#ABOVE3,R3	;GET POINTER TO MAX SPEC'D LIMIT MSG
6084	023636	004737	033770		JSR	PC,TYPTMS	;TYPE TIMING TEST RESULTS
6085							
6086							
6087							
6088							
6089							
6090							
6091							
6092							
6093							
6094							
6095	023642	000004					
6096	023644	012737	000017	001324	†ST17: SCOPE		
6097	023652	004737	030462		MOV	#17,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
6098	023656	004737	030530		JSR	PC,SETUP	;SET UP FOR LOOP ON ERROR
6099	023662	000137	024262		JSR	PC,CHKITR	;SEE IF ITER. NO. = 0 FOR THIS TEST
6100					JMP	TS120	;JUMP TO NEXT TEST
6101	023666	004737	027522				SHOULD BE RUN
6102	023672	004737	026040		JSR	PC,INITSS	;INITIALIZE DRIVER PARAMS AND SUB-SYS
6103	023676	112765	000123	000001	JSR	PC,PREPKB	;PREPARE FOR POSSIBLE KBD INPUT
6104	023704	013765	005732	000002	MOVB	#WRDATA,P.CMND(R5)	;SET WRITE DATA COMMAND
6105	023712	113765	005740	000005	MOV	FC,P.CYL(R5)	;SET CYL = FC
6106	023720	013737	005732	001174	MOVB	FT,P.TRACK(R5)	;SET TRACK = FT
6107	023726	013737	005740	001176	MOV	FC,\$REG5	;SET PACK ADR FOR ERROR PRINTOUT
6108	023734	005037	001200		MOV	FT,\$REG6	
6109	023740	105065	000004		CLR	\$REG7	
6110	023744	012765	063554	000010	CLRB	P.SECT(R5)	;SET SECTOR = 0
6111	023752	012765	177400	000012	MOV	#RWBUFF,P.BALO(R5)	;SET DATA BUFFER ADDRESS
6112					MOV	#-256.,P.WC(R5)	;SET WORD CNT FOR 256(DEC) WORDS
6113	023760	012703	000024		;SET NUMBER OF SECTORS IN R3		
6114	023764	105737	003125		MOV	#20.,R3	;SET R3 INITIALLY = 20(DEC)
6115	023770	001002			TSTB	FORMAT	;DETERMINE THE FORMAT
					BNE	4\$	;BR IF 20 SECTOR FORMAT

```

6116 023772 012703 000026      MOV      #22,R3      ;SET R3 = 22(DEC)
6117      ;WRITE THE CURRENT SECTOR WITH THE SECTOR NUMBER + 100(OCT)
6118 023776 116500 000004      4$:      MOV      P.SECT(R5),R0      ;PUT SECTOR NO. INTO R0
6119 024002 062700 000100      ADD      #100,R0      ;ADD 100(OCT) TO SECTOR NO.
6120 024006 004737 034304      JSR      PC,LODSEC      ;LOAD THE ENTIRE BUFFER WITH SECTOR NO. + 100(OCT)
6121 024012 004737 040316      JSR      PC,DRVCL      ;WRITE THE SECTOR
6122 024016 105265 000004      INCB     P.SECT(R5)      ;INCREMENT THE SECTOR NO.
6123 024022 120365 000004      CMPB     R3,P.SECT(R5)      ;SEE IF LAST SECTOR WRITTEN YET
6124 024026 001363      BNE      4$      ;BR IF NOT DONE YET
6125 024030 012737 024030 001110      MOV      #,SLPERR      ;SET NEW LOOP ON ERROR ADRS
6126 024036 004737 030462      JSR      PC,SETUP      ;SET UP FOR LOOP ON ERROR
6127 024042 105065 000004      CLRB     P.SECT(R5)      ;INITIALIZE SECTOR TO 0
6128      ;DO WRITE CHECK OF A SECTOR
6129 024046 116500 000004      8$:      MOV      P.SECT(R5),R0      ;GET SECTOR INTO R0
6130 024052 062700 000100      ADD      #100,R0      ;ADD 100(OCT) TO SECTOR NO.
6131 024056 004737 034304      JSR      PC,LODSEC      ;LOAD BUFFER WITH SECTOR + 100(OCT)
6132 024062 112765 000131 000001      MOV      #WRCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6133 024070 004737 040316      JSR      PC,DRVCL      ;DO A WRITE CHECK
6134      ;READ THE SECTOR AND COMPARE THE DATA TO SECTOR NUMBER + 100(OCT)
6135 024074 112765 000121 000001      MOV      #RDATA,P.CMND(R5) ;SET READ COMMAND
6136 024102 004737 040316      JSR      PC,DRVCL      ;READ THIS SECTOR INTO MEMORY
6137 024106 012701 063554      MOV      #RBUF,R1      ;GET ADDR. OF DATA BUF INTO R1
6138 024112 005004      CLR      R4      ;CLEAR WORD NUMBER
6139 024114 005037 005542      CLR      SCRACH      ;INIT. COMPARE ERROR COUNT
6140 024120 116537 000004 001200      MOV      P.SECT(R5),R4      ;
6141 024126 032737 000002 005504      BIT      #BSERR,RECODE      ;SEE IF BAD SECTOR ERROR OCCURRED
6142 024134 001045      BNE      22$      ;BR IF YES
6143 024136 020021      10$:      CMP      R0,(R1)+      ;COMPARE BUF WORD TO SECTOR + 100(OCT)
6144 024140 001437      BEQ      20$      ;BR IF NO COMPARE ERROR
6145 024142 005737 005542      TST      SCRACH      ;CHECK ERROR COUNT
6146 024146 001006      BNE      12$      ;BR IF NOT FIRST ERROR IN SECTOR
6147 024150 105037 062105      CLRB     DH701+38.      ;ADJUST DATA HEADER FOR MSG
6148 024154 012737 000005 063446      MOV      #5,DF25+2      ;ADJ. ERROR DATA WORD COUNT
6149 024162 104034      ERROR      34      ;TYPE HEADING FOR ERROR MSG
6150 024164 005237 005542      INC      SCRACH      ;INCREMENT ERROR COUNT
6151 024170 032777 000001 154742      BIT      #BIT0,DSWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
6152 024176 001004      BNE      14$      ;BR TO REPORT ALL ERRORS
6153 024200 022737 000012 005542      CMP      #10.,SCRACH      ;SEE IF 10(DEC) ERRORS YET
6154 024206 002420      BLT      22$      ;BR IF > 10.
6155 024210 010437 001202      14$:      MOV      R4,$REG10      ;WORD NUMBER
6156 024214 010037 001204      MOV      R0,$REG11      ;GOOD DATA
6157 024220 016137 001206      MOV      -2(R1),$REG12      ;BAD DATA
6158 024226 104063      ERROR      63      ;TYPE GOOD AND BAD DATA
6159 024230 032777 000100 154702      BIT      #BIT6,DSWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
6160 024236 001004      BNE      20$      ;BR IF JUST 1 ERROR SHOULD BE REPORTED
6161 024240 005204      INC      R4      ;INCREMENT WORD NUMBER
6162 024242 022704 000400      CMP      #256.,R4      ;SEE IF DONE YET
6163 024246 001333      BNE      10$      ;BR IF NOT DONE COMPARING YET
6164 024250 105265 000004      22$:      INCB     P.SECT(R5)      ;INCREMENT SECTOR NO.
6165 024254 120365 000004      CMPB     R3,P.SECT(R5)      ;SEE IF ALL SECTORS CHECKED YET
6166 024260 001272      BNE      8$      ;BR IF NOT DONE YET
6167
6168
6169
6170      ;*****
6171      ;*TEST 20 TRACK ADDRESSING TEST

```

B10

6172
6173
6174
6175
6176
6177
6178 024262 000004
6179 024264 012737 000020 001324
6180 024272 004737 030462
6181 024276 004737 030530
6182 024302 000137 024516
6183
6184 024306 004737 027522
6185 024312 004737 026040
6186 024316 112765 000123 000001
6187 024324 013765 005732 000002
6188 024332 105065 000005
6189 024336 113765 005516 000004
6190 024344 012765 06 54 000010
6191 024352 012765 177400 000012
6192
6193 024360 116500 000005
6194 024364 062700 000100
6195 024370 004737 034304
6196 024374 004737 040316
6197 024400 105265 000005
6198 024404 122765 000003 000005
6199 024412 001362
6200
6201 024414 012737 024414 001110
6202 024422 004737 030462
6203 024426 004737 034326
6204
6205 024432 012737 024432 001110
6206 024440 004737 030462
6207 024444 105065 000005
6208 024450 116500 000005
6209 024454 062700 000100
6210 024460 004737 034304
6211 024464 112765 000123 000001
6212 024472 004737 040316
6213 024476 004737 034326
6214 024502 105265 000005
6215 024506 122765 000003 000005
6216 024514 001355
6217
6218
6219
6220 024516 000040
6221
6222
6223
6224
6225
6226
6227

;*IN THIS TEST, SECTOR FS OF CYL FC IS WRITTEN WITH 256 (DEC) WORDS
;*OF THE TRACK NO. + 100(OCT), FOR EACH OF TRACKS 0,1,2. THEN, A
;*WRITE CHECK OF EACH SECTOR IS DONE TO VERIFY THE WRITES. THEN, EACH
;*OF THE 3 SECTORS IS RE-WRITE, AND AFTER EACH WRITE ALL OF THE
;*THREE SECTORS ARE WRITE-CHECKED TO DETECT TRACK ADDRESSING PROBLEMS.
;*****

TST20: SCOPE
MOV #20,\$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 TST21 ;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
MOVB #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
MOV FC,P.CYL(R5) ;SET CYL = FC
CLRB P.TRCK(R5) ;INITIALIZE TRACK NO. TO 0
MOVB FS,P.SECT(R5) ;SET SECTOR =FS
MOV #RWBUFF,P.BAL0(R5) ;SET DATA BUFFER ADDRESS
MOV #-256,P.WC(R5) ;SET WORD COUNT FOR 256(DEC) WORDS
;WRITE THE TRACK NO. + 100(OCT) AT SECTOR FS OF TRACKS 0,1,2
4\$: MOVB P.TRCK(R5),R0 ;SET TRACK NO. IN R0
ADD #100,R0 ;ADD 100(OCT) TO TRACK NO.
JSR PC,LOADSEC ;LOAD BUFFER WITH TRACK NO. + 100(OCT)
JSR PC,DRVCAL ;WRITE THE SECTOR
INCB P.TRCK(R5) ;INCREMENT THE TRACK NO.
CMPB #3,P.TRCK(R5) ;SEE IF ALL 3 TRACKS WRITTEN YET
BNE 4\$;BR IF NOT DONE YET
;DO A WRITE-CHECK OF THE 3 SECTORS WRITTEN, TO VERIFY THE DATA
MOV #,\$LPERR ;SET NEW LOOP ON ERROR ADRS
JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
JSR PC,TRKCHK
;WRITE A SECTOR AND WRITE-CHECK ALL 3 SECTORS
MOV #,\$LPERR ;SET NEW LOOP ON ERROR ADRS
JSR PC,SETUP ;SET UP FOR LOOP ON ERROR
CLRB P.TRCK(R5) ;INIT TRACK TO 0
6\$: MOVB P.TRCK(R5),R0 ;GET TRACK NO. IN R0
ADD #100,R0 ;ADD 100(OCT) TO TRACK NO.
JSR PC,LOADSEC ;LOAD BUF WITH TRACK + 100(OCT)
MOVB #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
JSR PC,DRVCAL ;RE-WRITE A SECTOR
JSR PC,TRKCHK ;WRITE-CHECK ALL 3 SECTORS
INCB P.TRCK(R5) ;INCR THE TRACK NO.
CMPB #3,P.TRCK(R5) ;SEE IF ALL 3 TRACKS RE-WITTEN YET
BNE 6\$;BR IF NOT DONE YET

RWDTEST:
RWTINX=<\$TN-1>*2 ;R/W TEST INDEX
;*****
;*TEST 21 READ/WRITE DATA TEST
;*THE READ/WRITE DATA TEST HAS 2 DIFFERENT VERSIONS, DEPENDING
;*ON THE VALUE OF THE DATA PATTERN PARAMETER (PT).
;*:
;* FOR PT = 0 :

```
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253 024516 000004
6254 024520 012737 000021 001324
6255 024526 004737 030462
6256 024532 004737 030530
6257 024536 000137 025430
6258
6259 024542 004737 027522
6260 024546 004737 026040
6261 024552 105037 003134
6262 024556 032777 000040 154354
6263 024564 001002
6264 024566 004737 034412
6265 024572 013737 005760 005544
6266 024600 005737 005760
6267 024604 001020
6268
6269 024606 012765 063554 000010
6270 024614 005065 000002
6271 024620 105065 000005
6272 024624 012765 175000 000012
6273 024632 105065 000004
6274 024636 004737 035762
6275 024642 025404
6276 024644 000457
6277
6278 024646 013765 005762 000010
6279 024654 013700 005764
6280 024660 042700 177774
6281 024664 150065 000007
6282 024670 013737 005762 005602
6283 024676 013737 005764 005604
```

```

; *THIS TEST IS THE QUICK VERIFY DEFAULT DATA TEST, WHICH IS RUN
; *WHEN PT=0, DUE TO PARAMETER INPUT CHOICE, OR ADDRESS 200 START.
; *IN THIS TEST, THE ENTIRE PACK IS TESTED WITH A SINGLE DATA PATTERN.
; *THIS PATTERN IS COMPRISED OF THE DATA PATTERNS 00-15, WHICH REPEATS
; *EVERY 256(DEC) WORDS. EACH TRACK IS WRITTEN IN 4 SEGMENTS, FOLLOWED
; *BY A WRITE-CHECK, READ, AND SOFTWARE COMPARE. THE SEGMENTS ARE 6
; *SECTORS EACH, WHICH MEANS THAT SPIRALING OCCURS ON THE LAST
; *SEGMENT WRITTEN ON EACH TRACK.
;
; * FOR PT NOT = 0 :
; *THIS TEST PERFORMS READ/WRITE FUNCTIONS ON THE ENTIRE RANGE OF
; *CYLINDERS (FC-LC), TRACKS (FT-LT), AND SECTORS (FS-LS) SPECIFIED.
; *AT EACH SPECIFIED SECTOR WC WORDS OF THE CURRENT REPEATING DATA
; *PATTERN ARE WRITTEN, AND THEN WRITE-CHECKED, FOLLOWED BY A READ
; *AND SOFTWARE COMPARE. THIS IS DONE FOR ALL THE SPEC'D SECTORS ON ALL
; *THE TRACKS, USING SECTOR INCR IS, AND TRACK INCR IT.
; * THEN IT IS REPEATED USING EACH OF THE OTHER DATA
; *PATTERNS CHOSEN IN PARAMETER PT. THEN, EACH OF THE ABOVE OPERATIONS
; *ARE REPEATED AT EACH OF THE REMAINING CYLINDERS, IN THE SPECIFIED
; *RANGE, USING THE CYLINDER INCREMENT IC.
; * NOTE THAT FS MUST BE CHOSEN < OR = LS, AND FT MUST BE < OR = LT.
; *HOWEVER, FC MAY BE < , OR > LC, AND IF FC>LC, THE CYLINDER
; *ADDRESS WILL BE DECREMENTED BY IC (INSTEAD OF INCREMENTED) TO
; *OBTAIN EACH NEW CYLINDER ADDRESS.
; *****
TST21: SCOPE
      MOV      #21, $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      $EOP           ; JUMP TO END-OF-PASS ROUTINE
; 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
      CLRB     XOVLAB         ; INIT XXDP OVERLAID INDICATOR
      BIT      #BITS, $SWR     ; SEE IF WRITES INHIBITED
      BNE      2$             ; IF YES, DON'T CHANGE RAND DATA PAT
      JSR      PC, $LODP14     ; GENERATE PSEUDO-RAND PAT 14
      MOV      PT, $PATRN      ; GET A COPY OF PT
      TST      PT             ; SEE IF QUICK VERIFY DATA TEST DESIRED
      BNE      7$             ; BR IF NOT QUICK VERIFY
; SET PARAMETERS FOR QUICK VERIFY DEFAULT DATA TEST
      MOV      $RWBUF, P, $BALO(R5) ; SET R/W BUFFER ADDRESS
      CLR      P, $CYLN(R5)    ; INIT CYL TO 0
      4$:      CLRB     P, $TRCK(R5) ; INIT TRACK TO 0
      5$:      MOV      #-1536, P, $WC(R5) ; SET WORD COUNT FOR 6 SECTORS
      CLRB     P, $SECT(R5)    ; INIT SECTOR TO 0
      6$:      JSR      PC, $CHKLIM ; CHK WRD COUNT AND PACK ADR TO AVOID OVERFLOW
      30$      ; RETURN ADDR FOR TRANSFER NOT ALLOWED
      BR       18$            ; PROCEED
; SET PARAMETERS FOR NON-DEFAULT DATA TEST
      7$:      MOV      MA, P, $BALO(R5) ; SET BA BITS 0-15
      MOV      MA+2, RO        ; GET MA BITS 16-21
      BIC      #177774, RO     ; MASK FOR LO 2 BITS
      BISB     RO, P, $BAHT(R5) ; SET BA BITS 16, 17
      MOV      MA, PMA        ; SET BUFFER ADDRESS
      MOV      MA+2, PMA+2
```

6284	024704	105237	003134		INCB	XOVLAD		;SET XXDP OVERLAID INDICATOR
6285	024710	005737	055716		TST	\$KT11		;SEE IF MEMORY MANAGEMENT PRESENT
6286	024714	100002			BPL	9\$		;BR IF NO MEM. MGT.
6287					;PREPARE PAGE ADDRESS REGISTERS FOR RELOCATION			
6288	024716	004737	026604		JSR	PC,PREPAR		;PREPARE MEM MGT FOR RELOCATION
6289					;INITIALIZE TEST PARAMETERS			
6290	024722	013765	005732	000002	5\$:	MOV	FC,P.CYLN(R5)	;INIT CYL TO FC
6291	024730	012701	000001		44\$:	MOV	#1,R1	;INIT PATTERN BIT POINTER
6292	024734	030137	005760		10\$:	BIT	R1,PT	;SEE IF THIS PATTERN SELECTED
6293	024740	001003				BNE	14\$	;BR IF THIS PATTERN IS SELECTED
6294	024742	006301			11\$:	ASL	R1	;SHIFT PATTERN BIT POINTER
6295	024744	001571				BEQ	25\$	;BR IF ALL PATTERNS CHECKED ON THIS CYLINDER
6296	024746	000772				BR	10\$	;GO CHECK ANOTHER PATTERN
6297	024750	113765	005740	000005	14\$:	MOV	FT,P.TRCK(R5)	;INIT TRACK TO FT
6298	024756	113765	005516	000004	16\$:	MOV	FS,P.SECT(R5)	;INITIALIZE SECTOR TO FS
6299	024764	013765	005766	000012	17\$:	MOV	WC,P.WC(R5)	;SET WORD COUNT FOR WC WORDS
6300	024772	005465	000012			NEG	P.WC(R5)	
6301	024776	004737	035762			JSR	PC,CHKLIM	;CHECK WRD CNT AND PACK ADR TO AVOID OVERFLOW
6302	025002	025330			25\$			;RETURN ADDRESS FOR TRANSFER NOT ALLOWED
6303					;PERFORM TESTS AT CURRENT ADDRESS			
6304	025004	012737	025004	001110	18\$:	MOV	#,SLPERR	;SET NEW LOOP ON ERROR ADRS
6305	025012	004737	030462			JSR	PC,SETUP	;SET UP FOR LOOP ON ERROR
6306	025016	004737	037626			JSR	PC,SVPRMS	;STORE PARAMS FOR THIS XFER
6307	025022	004737	034626			JSR	PC,LODBUF	;LOAD DATA INTO R/W BUFFER
6308	025026	105037	003140			CLRB	REISSU	;DUAL-ACC COMMAND RE-ISSUE FLAG
6309	025032	032777	000040	154100		BIT	#BIT5,ASWR	;SEE IF DATA WRITES INHIBITED
6310	025040	001005				BNE	20\$	;BR IF INHIBITED
6311	025042	112765	000123	000001		MOV	WRDATA,P.CMND(R5)	;SET WRITE COMMAND
6312	025050	004737	037740			JSR	PC,TRANSR	;WRITE THE DATA
6313	025054	032777	000020	154056	20\$:	BIT	#BIT4,ASWR	;SEE IF WRITE CHECKS INHIBITED
6314	025062	001014				BNE	22\$	;BR IF INHIBITED
6315	025064	112765	000131	000001		MOV	WRCHK,P.CMND(R5)	;SET WRITE CHECK COMMAND
6316	025072	004737	037740			JSR	PC,TRANSR	;DO WRITE-CHECK OF DATA WRITTEN
6317	025076	105737	003141			TSTB	WCEFLG	;SEE IF A WCE ERROR OCCURRED
6318	025102	001404				BEQ	22\$	;BR IF NOT
6319	025104	032777	000002	154026		BIT	#BIT1,ASWR	;SEE IF READ AND COMPARE SHOULD BE DONE
6320	025112	001417				BEQ	24\$	;BR IF NOT
6321	025114	032777	000010	154016	22\$:	BIT	#BIT3,ASWR	;SEE IF READ AND COMPARE INHIBITED
6322	025122	001013				BNE	24\$	;BR IF INHIBITED
6323	025124	112765	000121	000001		MOV	RDDATA,P.CMND(R5)	;SET READ COMMAND
6324	025132	004737	037740			JSR	PC,TRANSR	;READ THE DATA
6325	025136	032777	000004	153774		BIT	#BIT2,ASWR	;SEE IF SOFTWARE COMPARES INHIBITED
6326	025144	001002				BNE	24\$	;BR IF INHIBITED
6327	025146	004737	035054			JSR	PC,CMPBUF	;PERFORM SOFTWARE COMPARE OF DATA
6328	025152	105737	003136		24\$:	TSTB	DULACS	;SEE IF DUAL-ACCESS DATA TEST
6329	025156	001405				BEQ	27\$	;BR IF NOT DUAL-ACCESS
6330	025160	112765	000140	000001		MOV	RELEAS,P.CMND(R5)	;SET DRIVE RELEASE COMMAND
6331	025166	004737	040316			JSR	PC,DRVCAL	;RELEASE THE DRIVE
6332	025172	104411			27\$:	SCOPER		;CHECK FOR INTERNAL LOOP ON ERROR
6333	025174	005737	005760			TST	PT	;SEE IF QUICK VERIFY DATA TEST DESIRED
6334	025200	001031				BNE	40\$	;BR IF NOT QUICK VERIFY
6335					;INCREMENT PACK ADDRESS FOR QUICK VERIFY DEFAULT DATA TEST			
6336	025202	062765	000006	000004		ADD	#6,P.SECT(R5)	;INCREMENT SECTOR BY 6
6337	025210	126527	000004	000030		CMPB	P.SECT(R5),#24.	;SEE IF SECTOR LIMIT REACHED YET
6338	025216	002002				BGE	34\$	;BR IF LIMIT EXCEEDED
6339	025220	000137	024636			JMP	6\$	;GO BACK AND TEST

E10

CZR6M00 RK611/06 SS VERIF 1
CZR6M0.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 122
T21 READ/WRITE DATA TESTSEQ 0400
SEQ 0121

6340	025224	105265	000005		34\$:	INCB	P.TRCK(R5)	; INCREMENT TRACK
6341	025230	122765	000003	000005		CMPB	#3 P.TRCK(R5)	; SEE IF TRACK LIMIT EXCEEDED
6342	025236	001402				BEQ	36\$	; BR IF YES
6343	025240	000137	024624			JMP	5\$	
6344	025244	005265	000002		36\$:	INC	P.CYLN(R5)	; INCREMENT CYL
6345	025250	023765	017762	000002		CMP	LCP1,P.CYLN(R5)	; SEE IF CYL LIMIT EXCEEDED
6346	025256	001452				BEQ	30\$	; BR IF YES
6347	025260	000137	024620			JMP	4\$	
6348								
6349	025264	063765	005756	000004	40\$:	ADD	IS,P.SECT(R5)	; INCREMENT SECTOR BY IS
6350	025272	126537	000004	005520		CMPB	P.SECT(R5),LS	; SEE IF LS EXCEEDED
6351	025300	003631				BLE	17\$	; BR IF NOT YET
6352	025302	116503	000005			MOVB	P.TRCK(R5),R3	; GET TRACK NO.
6353	025306	063703	005744			ADD	IT,R3	; INCREMENT TRACK
6354	025312	110365	000005			MOVB	R3,P.TRCK(R5)	; PUT IT BACK
6355	025316	020337	005742			CMP	R3,LT	; SEE IF LT EXCEEDED
6356	025322	003615				BLE	16\$	; BR IF NOT YET
6357	025324	000137	024742			JMP	11\$	; JMP TO CHECK FOR NEXT PATTERN
6358	025330	023737	005732	005734	25\$:	CMP	FC,LC	; SEE IF FC>LC
6359	025336	003011				BGT	26\$	; BR IF FC>LC
6360	025340	063765	005736	000002		ADD	IC,P.CYLN(R5)	; INCREMENT THE CYLINDER
6361	025346	026537	000002	005734		CMP	P.CYLN(R5),LC	; SEE IF LC EXCEEDED
6362	025354	003013				BGT	30\$	; BR IF LC EXCEEDED
6363	025356	000137	024730			JMP	44\$	
6364	025362	163765	005736	000002	26\$:	SUB	IC,P.CYLN(R5)	; DECREMENT THE CYLINDER
6365	025370	026537	000002	005734		CMP	P.CYLN(R5),LC	; SEE IF LC EXCEEDED
6366	025376	002402				BLT	30\$	; BR IF LC EXCEEDED
6367	025400	000137	024730			JMP	44\$	
6368	025404	105737	003134		30\$:	TSTB	XOVLAD	; SEE IF XXDP MIGHT BE OVERLAID NOW
6369	025410	001402				BEQ	31\$	; BR IF NOT
6370	025412	004737	027262			JSR	PC,GETXDP	; RESTORE XXDP LOADER, IF SAVED
6371	025416	105737	003144		31\$:	TSTB	UBMPRS	; SEE IF UNIBUS MAP PRESENT
6372	025422	001402				BEQ	32\$	; BR IF NOT
6373	025424	005037	172516		32\$:	CLR	#SR3	; DISABLE 22-BIT MODE AND UNIBUS MAP
6374	025430							
6375								
6376								
6377								
6378								
6379								
6380								
6381								
6382								
6383								
6384								
6385								
6386	025430							
6387	025430	000004						
6388	025432	005237	001330					
6389	025436	000137	017026					
6390	025442							
6391	025442	042777	000100	153474				
6392	025450	005037	001102					
6393	025454	005037	001304					
6394	025460	005237	001326					
6395	025464	042737	100000	001326				

.SBTTL END OF PASS ROUTINE

```

;*****
; INCREMENT THE PASS NUMBER ($PASS)
; TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
; IF THERES A MONITOR GO TO IT
; IF THERE ISN'T JUMP TO APTPAS

```

\$EOP:

```

SCOPE
INC $DEVCT ; INCREMENT DEVICE COUNT FOR APT
JMP NEWDRV ; GO SEE IF MORE DRIVES TO TEST ON THIS PASS
DUNPAS: ; THIS IS TRULY THE END OF A PASS
BIC #BIT6,$STKS ; DISABLE TTY KBD INTERRUPT
CLR $TSTNM ; ZERO THE TEST NUMBER
CLR $TIMES ; ZERO THE NUMBER OF ITERATIONS
INC $PASS ; INCREMENT THE PASS NUMBER
BIC #100000,$PASS ; DON'T ALLOW A NEG. NUMBER

```

```

6396 025472 005327
6397 025474 000001
6398 025476 003022
6399 025500 012737
6400 025502 000001
6401 025504 025474
6402 025506 104401 025553
6403 025512 013746 001326
6404 025516 104405
6405 025520 104401 025550
6406 025524 013700 000042
6407 025530 001405
6408 025532 000005
6409 025534 004710
6410 025536 000240
6411 025540 000240
6412 025542 000240
6413 025544
6414 025544 000137
6415 025546 025570
6416 025550 377 377 000
6417 025553 015 042412 042116
6418 025560 050040 051501 020123
6419 025566 000043
6420 025570 122737 000001 001340
6421 025576 001007
6422 025600 023727 001326 000002
6423 025606 103403
6424 025610 005237 001102
6425 025614 000775
6426 025616 000137 017014
6427
6428
6429
6430
6431
6432
6433 025622 032765 010000 000020
6434 025630 001002
6435 025632 000137 042164
6436 025636 010046
6437 025640 010146
6438 025642 012700 000001
6439 025646 005001
6440 025650 120137 005510
6441 025654 001403
6442 025656 005201
6443 025660 006300
6444 025662 000772
6445 025664 040037 005540
6446 025670 012601
6447 025672 012600
6448 025674 000137 044342
6449
6450
6451

;EOPT: DEC (PC)+ ;:LOOP?
; BGT $DOAGN ;:YES
; MOV (PC)+,2(PC)+ ;:RESTORE COUNTER
; SENDCT: .WORD 1
; SEOPT TYPE $SENDMG ;: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 $42,RO ;:GET MONITOR ADDRESS
; BEQ $DOAGN ;:BRANCH IF NO MONITOR
; RESET ;:CLEAR THE WORLD
; SENDAD: JSR PC,(RO) ;:GO TO MONITOR
; NOP ;:SAVE ROOM
; NOP ;:FOR
; NOP ;:ACT11
; $DOAGN: JMP 2(PC)+ ;:RETURN
; $RTNAD: .WORD APTPAS
; $ENULL: .BYTE -1,-1,0 ;:NULL CHARACTER STRING
; SENDMG: .ASCIZ <15><12>/END PASS #/

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

; THIS IS THE ABNORMAL RETURN FROM THE DRIVER.
; WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
; (NED = NON-EXISTENT DRIVE)
NEDHDL: BIT $NED,P.CS2(R5) ;:SEE IF NED ON DRIVE SELECT
; BNE 1$ ;:BR IF NED SET
; JMP ERRHDL ;:GO HANDLE OTHER ERROR
1$: MOV RO,-(SP) ;:SAVE RO,R1
; MOV R1,-(SP)
; MOV #1,RO ;:SET BIT POINTER
; CLR R1 ;:CLEAR COUNTER
2$: CMPB R1,DRIVE ;:SEE IF R1 = CURRENT DRIVE NUMBER
; BEQ 3$ ;:BR IF =
; INC R1 ;:INCREMENT COUNTER
; ASL RO ;:SHIFT BIT POINTER
; BR 2$ ;:TRY AGAIN
3$: BIC RO,NEWON ;:CLEAR ONLINE BIT FOR THIS DRIVE
; MOV (SP)+,R1 ;:RESTORE RO,R1
; MOV (SP)+,RO
; JMP RETNML ;:TAKE NORMAL RETURN

;:*****

```

G10

6452
6453
6454
6455 025700 010146
6456 025702 017701 153240
6457 025706 042701 177600
6458 025712 120127 000172
6459 025716 003005
6460 025720 120127 000141
6461 025724 002402
6462 025726 042701 000040
6463 025732 010137 005532
6464 025736 122701 000003
6465 025742 001007
6466 025744 104401 012426
6467 025750 042777 000100 153166
6468 025756 012601
6469 025760 000002
6470 025762 122701 000032
6471 025766 001003
6472 025770 104401 012433
6473 025774 000765
6474 025776 122701 000022
6475 026002 001003
6476 026004 104401 012440
6477 026010 000757
6478 026012 122701 000007
6479 026016 001003
6480 026020 104401 012452
6481 026024 000751
6482 026026 104401 005532
6483 026032 104401 001315
6484 026036 000744
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498 026040 005077 153102
6499 026044 005037 005532
6500 026050 052777 000100 153066
6501 026056 000207
6502
6503
6504
6505
6506
6507

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

```
;*****
.SBTTL PREPKB - PREPARE FOR KEYBOARD INPUT
;THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
;DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
;INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
;ENABLES KBD INTERRUPT.
;CALL:
;JSR PC, PREPKB
;*****
PREPKB: CLR @STKB ;CLEAR KBD BUFFER AND DONE BIT
        CLR INTCHR ;CLEAR TTY INPUT WORD
        BIS #BIT6, @STKS ;ENABLE KBD INTERRUPT
        RTS PC ;RETURN
```

```
;*****
.SBTTL ECHOBAD - ECHO BAD TTY INPUT
;THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
```

H10

6508
6509
6510
6511
6512
6513
6514 026060 104401 005532
6515 026064 104401 001314
6516 026070 000207
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526
6527 026072 104407
6528 026074 012737 000200 055716
6529 026102 004737 055660
6530 026106 005737 055716
6531 026112 100406
6532 026114 013737 056162 006126
6533 026122 005037 006130
6534 026126 000436
6535
6536 026130 013700 056164
6537 026134 005001
6538 026136 012702 000006
6539 026142 000241
6540 026144 006100
6541 026146 006101
6542 026150 005302
6543 026152 001373
6544
6545 026154 063700 056162
6546 026160 005501
6547 026162 010037 006126
6548 026166 010137 006130
6549 026172 000241
6550 026174 006100
6551 026176 006101
6552 026200 006100
6553 026202 006101
6554 026204 006100
6555 026206 006101
6556 026210 020127 000037
6557 026214 103403
6558 026216 112737 000001 003144
6559
6560 026224 104401 007413
6561 026230 012746 006126
6562 026234 004737 053422
6563 026240 004737 053736

```

; *TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
; *REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
; *AND <LF> ARE DONE.
; * CALL - JSR PC,ECOBAD
; *****
ECOBAD: TYPE ,INTCHR ;ECHO BAD CHARACTER
        TYPE $QUES ;TYPE <?> AND <CR>, <LF>
        RTS PC ;RETURN

; *****
; *SIZMEM - SIZE MEMORY, SET LIMITS
; *THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE, AND
; *IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
; *IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D).
; *AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70).
; *****
SIZMEM: SAVREG ;SAVE R0-R5
        MOV #200,$KT11 ;SET MEM MGT KEY FOR $SIZE
        JSR PC,$SIZE ;SIZE MEMORY
        TST $KT11 ;SEE IF MEM MGT PRESENT
        BMI $S ;BR IF MEM MGT PRESENT
        MOV $LSTAD,MAHILM ;SET MEM LIMIT LO BITS
        CLR MAHILM+2 ;CLEAR MEM LIMIT HI BITS
        BR 16$ ;GO TYPE LAST ADDRESS
;SHIFT SAF LEFT 6, AND PUT IN R1-R0
8$: MOV $LSTBK,R0 ;LO BITS
    CLR R1 ;HI BITS
    MOV #6,R2 ;SET LOOP COUNT = 6
12$: CLC ;ROTATE LOOP
    ROL R0
    ROL R1
    DEC R2
    BNE 12$
;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
    ADD $LSTAD,R0 ;ADD LO BITS
    ADC R1 ;HI BITS
    MOV R0,MAHILM ;SET LO BITS OF MEM LIMIT
    MOV R1,MAHILM+2 ;SET HI BITS OF MEM LIMIT
    CLC
    ROL R0
    ROL R1
    ROL R0
    ROL R1
    ROL R0
    ROL R1
    CMP R1,#37 ;SEE IF 22 BIT ADDR
    BLO 16$ ;BR IF NO
    MOVB #1,UBMPRS ;SET "UNIBUS MAP PRESENT" FLAG
;TYPE "LAST PHYS MEM ADR = XXXXXXXX"
16$: TYPE LSTMEM ;TYPE "LAST PHYS MEM ADR ="
    MOV MAHILM,-(SP) ;PUT POINTER ON STACK
    JSR PC,$$SDB20 ;CONVERT BINARY TO OCTAL ASCII
    JSR PC,$$SSUPRS ;TYPE "XXXXXXX"

```

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 126
ECOBAD - ECHO BAD TTY INPUT

SEQ 0404
SEQ 0125

6564 026244 104401 001315
6565 026250 104401 001315
6566 026254 104410
6567 026256 000207

TYPE , \$CRLF ; TYPE <CR>, <LF>
TYPE , \$CRLF
RESREG ; RESTORE R0-R5
RTS PC ; RETURN

6568
6569
6570
6571
6572
6573
6574
6575
6576
6577 026260 016646 000002
6578 026264 104403
6579 026266 006
6580 026267 000
6581 026270 104401 007623
6582 026274 004737 030624
6583 026300 026332
6584 026302 026332
6585 026304 026332
6586 026306 005700
6587 026310 001001
6588 026312 000207
6589 026314 020027 000006
6590 026320 003407
6591 026322 104401 005272
6592 026326 104401 001314
6593 026332 162716 000010
6594 026336 000207
6595 026340 012746 005272
6596 026344 004737 052464
6597 026350 026322
6598 026352 012600
6599 026354 005737 052616
6600 026360 001360
6601 026362 010066 000002
6602 026366 000207

```

*****
; GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
; ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
; TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
; TOP OF STACK.
*****
GETPRM: MOV     2(SP), -(SP)      ; GET OLD VALUE
        TYPE    6               ; TYPE IT
        .BYTE   0               ; SIX DIGITS
        .BYTE   0               ; SUPPRESS LEADING ZEROS
        TYPE    NEWMSG          ; TYPE "NEW ="
        JSR     PC, RDCHRS      ; READ NEW VALUE FROM KBD
        7$      (1C) RETURN ADDRESS
        7$      (1Z) RETURN ADDRESS
        7$      (1U) OR ERROR RETURN ADDRESS
        TST     R0              ; SEE IF ANY CHARS TYPED
        BNE     4$             ; BR IF YES
        RTS     PC              ; RETURN - OLD VALUE UNCHANGED
4$:      CMP     R0, #6         ; SEE IF > 6 CHARS TYPED
        BLE     8$             ; BR IF NOT BAD
6$:      TYPE    , BUFFO        ; ECHO BAD INPUT
        TYPE    , $QUES
        7$:      SUB     #10, (SP) ; FIX ERROR RETURN PC
        RTS     PC              ; ERROR RETURN
8$:      MOV     #BUFFO, -(SP)   ; PUT POINTER TO CHARS ON STACK
        JSR     PC, OCTBIN      ; CONVERT DIGITS TO BINARY
        6$      MOV     (SP)+, R0 ; ERROR RETURN ADDRESS
        TST     $HI0C+          ; GET NEW BINARY VALUE
        BNE     6$             ; SEE IF HI BITS ARE 0
        MOV     R0, 2(SP)       ; BR IF NOT
        RTS     PC              ; PUT NEW VALUE ON STACK
        ; RETURN

```

6603
6604
6605
6606
6607
6608
6609
6610
6611
6612
6613 026370 022737 000176 001140
6614 026376 001010
6615 026400 013746 000176
6616 026404 104401 007614
6617 026410 004737 026260
6618 026414 012637 000176
6619 026420 000207

```

*****
; SBTTL GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION
; THIS SUBROUTINE ALLOWS THE CONTENTS OF THE SOFTWARE SWITCH
; REGISTER TO BE MODIFIED BY TTY INPUT. THE SUBROUTINE TYPES
; "SWR = XXXXXX NEW =" AND WAITS FOR A NEW OCTAL VALUE
; OF UP TO SIX DIGITS TO BE TYPED.
*****
GTSWRG: CMP     #SWREG, SWR      ; SEE IF SOFTWARE SWR SELECTED
        BNE     6$             ; BR IF NOT
        MOV     SWREG, -(SP)    ; PUT OLD VALUE ON STACK
        TYPE    SWMSG          ; TYPE "SWR ="
        JSR     PC, GETPRM      ; TYPE OLD, GET NEW SWREG VALUE
        MOV     (SP)+, SWREG    ; STORE NEW VALUE
        6$:      RTS     PC      ; RETURN

```

J10

6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630
6631
6632
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643
6644
6645
6646
6647
6648
6649
6650
6651
6652
6653
6654
6655
6656
6657
6658
6659
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674
6675

026422	104407		
026424	005001		
026426	012702	172340	
026432	012703	000010	
026436	012762	077406	177740
026444	010122		
026446	062701	000200	
026452	005303		
026454	001370		
026456	012742	177600	
026462	105737	003144	
026466	001403		
026470	012737	000060	172516
026476	012737	001000	177572
026504	104410		
026506	000207		
026510	004737	041704	
026514	013737	177572	001174
026522	013737	177574	001176
026530	013737	177576	001200
026536	012737	026562	000004
026544	012737	000340	000006
026552	013737	172516	001202
026560	000406		
026562	022626		
026564	112737	000003	063526
026572	105037	062201	
026576	104121		
026600	000137	044134	

```

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

```

```

*****
INITMM: SAVREG          ;SAVE R0-R5
        CLR             R1          ;INIT FOR PAR LOADING
        MOV             #KIPAR0,R2 ;ADDR OF FIRST PAR
        MOV             #8,R3       ;LOAD 8 PAR'S AND 8 PDR'S
        MOV             #77406,-40(R2) ;PDR = 4K UP, READ/WRITE
        MOV             R1,(R2)+    ;LOAD A PAR
        ADD             #200,R1     ;UPDATE FOR NEXT PAR
        DEC             R3          ;DECREMENT LOOP COUNTER
        BNE             4$          ;LOOP UNTIL ALL 8 ARE LOADED
        MOV             #177600,-(R2) ;SET UP KIPAR7 FOR I/O PAGE
        TSTB            UBMPRS      ;SEE IF 22-BIT ADDRESSES
        BEQ             10$         ;BR IF NOT
        MOV             #60,2#SR3   ;ENABLE 22-BIT MODE,AND UNIBUS MAP
        MOV             #BIT9,2#SR0 ;ENABLE KT11 TRAPS
        RESREG          ;RESTORE R0-R5
        RTS             PC          ;RETURN

```

```

*****
;SERVICE ROUTINE FOR MEM MGT TRAPS
*****
KTERHD: JSR             PC,REPSUP   ;GATHER STATUS FOR PRINTOUT
        MOV             2#SR0,$REG5 ;GET SR0
        MOV             2#SR1,$REG6 ;GET SR1
        MOV             2#SR2,$REG7 ;GET SR2
        MOV             #8,2#ERRVEC ;SET TIME-OUT VECTOR
        MOV             #PR7,2#ERRVEC+2
        MOV             2#SR3,$REG10 ;GET SR3
        BR             10$
        CMP             (SP)+,(SP)+ ;CLEAN UP STACK
        MOVB            #3,DF30+18. ;FIX PRINTOUT FOR NO SR3
        CLRB            DH704+11.
        ERROR           121         ;KT11 FAILURE
        JMP             HLTPRG      ;ABORT !!!

```

```

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

```

K10

6676
6677 026604 104407
6678 026606 004737 026422
6679 026612 013700 005602
6680 026616 042700 017777
6681 026622 013701 005604
6682 026626 010003
6683 026630 010104
6684 026632 006100
6685 026634 006101
6686 026636 006100
6687 026640 006101
6688 026642 000301
6689 026644 006100
6690 026646 106001
6691 026650 010137 003206
6692 026654 105737 003144
6693 026660 001420
6694
6695 026662 012700 170200
6696 026666 012701 000037
6697 026672 010320
6698 026674 010420
6699 026676 062703 020000
6700 026702 005504
6701 026704 077106
6702 026706 042765 160000 000010
6703 026714 142765 000003 000007
6704 026722 104410
6705 026724 000207
6706
6707
6708
6709
6710
6711
6712
6713 026726 005737 006130
6714 026732 001404
6715 026734 012737 160000 005546
6716 026742 000412
6717 026744 023727 006126 157776
6718 026752 103370
6719 026754 013737 006126 005546
6720 026762 062737 000002 005546
6721 026770 162737 006000 005546
6722 026776 000207
6723
6724
6725
6726
6727
6728
6729
6730
6731

```

*****
PREPAR: SAVREG          ;SAVE R0-R5
      JSR PC,INITMM      ;INIT MEM MGT REGISTERS
      MOV PMA,R0          ;LO BITS OF MA
      BIC #17777,R0       ;MASK FOR BITS 13-15
      MOV PMA+2,R1        ;HI BITS OF MA
      MOV R0,R3           ;SAVE THESE BITS
      MOV R1,R4
      ROL R0              ;GET MA BITS 13-21 INTO R1 BITS 7-15
      ROL R1
      ROL R0
      ROL R1
      SWAB R1
      ROL R0
      RORB R1
      MOV R1,SAVPAR       ;CONSTANT FOR LOADING PAR6 LATER
      TSTB UBMPRS         ;SEE IF UNIBUS MAP ENABLED
      BEQ 9$              ;BR IF NOT (NO UNIBUS MAPPING)
;PREPARE FOR 22-BIT ADDRESSING - LOAD UNIBUS MAP REGISTERS
      MOV #MAPLOO,R0      ;STARTING ADDR OF MAP REGISTERS
      MOV #31,R1          ;SET REGISTER COUNTER
4$:   MOV R3,(R0)+         ;LOAD A MAP REGISTER
      MOV R4,(R0)+
      ADD #20000,R3        ;ADD 4K WORDS
      ADC R4
      SOB R1,4$           ;LOOP UNTIL 31(DEC) ARE LOADED
      BIC #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
      BICB #3,P.BAHI(R5)
9$:   RESREG              ;RESTORE R0-R5
      RTS PC              ;RETURN
*****

;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
4$:   MOV #160000,XXDPAD  ;START OF 28K IS END OF XXDP
      BR 8$
6$:   CMP MAHILM,#157776  ;SEE IF MEM LIMIT > OR = 157776
      BHS 4$              ;BR IF YES
      MOV MAHILM,XXDPAD  ;HI MEM LIMIT IS END OF XXDP
      ADD #2,XXDPAD       ;ADD 2 BYTES
8$:   SUB #6000,XXDPAD    ;COMPUTE START OF XXDP
      RTS PC              ;RETURN
*****

;CHKXDP - CHECK FOR IMMINENT OVERLAY OF XXDP LOADER
;THIS SUBROUTINE DETERMINES IF THE CURRENT MA AND WC COMBINATION
;WOULD CAUSE OVERLAY OF THE XXDP LOADER ON A WRITE TRANSFER.
;IF SO, AN ATTEMPT IS MADE TO MOVE THE LOADER INTO UNUSED MEMORY.
;IF THIS IS NOT POSSIBLE, A RETURN IS MADE TO THE ADDRESS

```

Address	Op-Code	Op-Code Hex	Op-Code Dec	Op-Code Name	Op-Code Comment
6732					;*FOLLOWING THE CALL. THE SAVE ADDRESS IS STORED IN XXDPSAV AND XXDPSAV+2.
6733					;*IF IT IS NOT NECESSARY TO MOVE THE LOADER, A NORMAL RETURN IS MADE.
6734					;* CALL - JSR PC,CHKXDP
6735					;* (RETURN ADR FOR NO SAVE)
6736					*****
6737	027000	104407		CHKXDP: SAVREG	SAVE RO-R5
6738				;SEE IF LOADER MUST BE PROTECTED	
6739	027002	105737	003136	TSTB DULACS	;SEE IF DUAL-ACCESS DATA TEST
6740	027006	001005		BNE 2\$	;BR IF YES
6741	027010	012700	005626	MOV #STLST,RO	;GET ADRS OF TEST LIST
6742	027014	005760	000040	TST RWTINX(RO)	;SEE IF R/W DATA TEST TO BE RUN
6743	027020	001406		BEQ 4\$	;EXIT IF NOT TO BE RUN
6744	027022	005737	005760	2\$: TST PT	;SEE IF DEFAULT DATA TEST
6745	027026	001403		BEQ 4\$	;BR IF DEFAULT DATA TEST
6746	027030	005737	005764	TST MA+2	;SEE IF HI MEM ADR = 0
6747	027034	001404		BEQ 6\$	;BR IF 0
6748	027036	062716	000002	4\$: ADD #2,(SP)	;FIX UP NORMAL RETURN PC
6749	027042	104410		5\$: RESREG	;RESTORE RO-R5
6750	027044	000207		RTS PC	;RETURN
6751	027046	013700	005766	6\$: MOV WC,RO	;GET WORD COUNT LO BITS
6752	027052	005300		DEC RO	;DECREMENT IT
6753	027054	005001		CLR R1	;SET HI BITS = 0
6754	027056	000241		CLC	;DOUBLE IT FOR BYTES
6755	027060	006100		ROL RO	
6756	027062	006101		ROL R1	
6757	027064	063700	005762	ADD MA,RO	;COMPUTE LAST ADR OF XFER IN R1-RO
6758	027070	005501		ADC R1	
6759	027072	063701	005764	ADD MA+2,R1	
6760	027076	105037	003135	CLRB XDPSVD	;INIT LOADER SAVE INDICATOR
6761	027102	004737	026726	JSR PC,FNDXDP	;COMPUTE START ADR OF XXDP
6762	027106	013703	005546	MOV XXDPAD,R3	;GET START OF XXDP LOADER
6763	027112	062703	005776	ADD #5776,R3	;COMPUTE LAST ADDRESS OF XXDP
6764	027116	023703	005762	CMP MA,R3	;SEE IF MA > LAST ADR OF XXDP
6765	027122	101345		BHI 4\$	;BR IF YES - DON'T MOVE XXDP
6766	027124	005701		TST R1	;CHECK HI BITS OF LAST XFER ADDRESS
6767	027126	001003		BNE 8\$	;BR IF NOT 0
6768	027130	020037	005546	CMP RO,XXDPAD	;CHECK LO BITS OF LAST XFER ADR
6769	027134	103740		BLO 4\$	;BR IF < XXDP ADR - DON'T MOVE XXDP
6770	027136	105237	003135	8\$: INCB XDPSVD	;SET SAVE INDICATOR
6771				;ATTEMPT TO FIND A SAVE AREA FOR LOADER	
6772	027142	023727	005762	100000 CMP MA,#100000	;SEE IF MA > OR = 16K
6773	027150	103410		BLO 12\$	;BR IF NOT
6774	027152	012737	072000	005550 MOV #72000,XXDPSAV	;SAVE ADR = 72000
6775	027160	005037	005552	CLR XDPSAV+2	
6776	027164	004737	027254	10\$: JSR PC,SAVXDP	;SAVE THE LOADER
6777	027170	000722		BR 4\$	;BR TO RETURN
6778	027172	062700	000002	12\$: ADD #2,RO	;ADD 2 BYTES TO LAST ADR OF XFER
6779	027176	005501		ADC R1	
6780	027200	010002		MOV RO,R2	;GET A COPY
6781	027202	010103		MOV R1,R3	
6782	027204	062702	005776	ADD #5776,R2	;COMPUTE END OF SAVE AREA
6783	027210	005503		ADC R3	
6784	027212	020337	006130	CMP R3,MAHILM+2	;SEE IF THIS EXCEEDS MEMORY LIMIT
6785	027216	101011		BHI 16\$	;BR IF YES - CAN'T SAVE LOADER
6786	027220	001003		BNE 14\$	;BR IF < MEM LIMIT
6787	027222	020237	006126	CMP R2,MAHILM	;SEE IF LO BITS EXCEED MEM LIMIT

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 130
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0408
SEQ 0129

M10

6788 027226 101005
6789 027230 010037 005550
6790 027234 010137 005552
6791 027240 000751
6792 027242 017616 000000
6793 027246 105037 003135
6794 027252 000673
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809 027254 104407
6810 027256 005004
6811 027260 000410
6812 027262 104407
6813 027264 105037 003134
6814 027270 105737 003135
6815 027274 001425
6816 027276 012704 000001
6817 027302 105737 000041
6818 027306 001420
6819 027310 012702 003000
6820 027314 013703 005546
6821 027320 013700 005550
6822 027324 013701 005552
6823 027330 005737 055716
6824 027334 100417
6825 027336 005704
6826 027340 001005
6827 027342 012320
6828 027344 005302
6829 027346 001375
6830 027350 104410
6831 027352 000207
6832 027354 062700 006000
6833 027360 062703 006000
6834 027364 014043
6835 027366 005302
6836 027370 001375
6837 027372 000766
6838
6839 027374 004737 026422
6840 027400 006100
6841 027402 006101
6842 027404 006100
6843 027406 006101

```

      BHI      16$      ;BR IF YES - CAN'T SAVE LOADER
14$:  MOV      R0,XDPSAV ;SET SAVE ADR IN HI MEMORY
      MOV      R1,XDPSAV+2
      BR       10$      ;GO SAVE LOADER
16$:  MOV      2(SP), (SP) ;FIX UP NO-SAVE RETURN PC
      CLRB     XDPSVD    ;CLEAR THE SAVE INDICATOR
      BR       5$      ;BR TO 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
      CLRB     XOVLAD   ;CLEAR XXDP OVERLAD INDICATOR
      TSTB     XDPSVD   ;SEE IF XXDP WAS SAVED
      BEQ      XDP2     ;BR IF NOT SAVED
      MOV      #1,R4    ;SET INDICATOR TO GET XXDP
XDP1:  TSTB     2#41     ;SEE IF LOADED BY XXDP
      BEQ      XDP2     ;BR IF NOT
      MOV      #1536.,R2 ;GET SET TO MOVE 1536(DEC) WORDS
      MOV      XXDPAD,R3 ;GET ORIG ADR OF START OF XXDP
      MOV      XDPSAV,R0 ;GET SAVE ADR LO BITS
      MOV      XDPSAV+2,R1 ;GET SAVE ADR HI BITS
      TST      SKT11    ;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
6$:   MOV      (R3)+, (R0)+ ;SAVE A WORD
      DEC      R2       ;SEE IF 1536(DEC) WORDS YET
      BNE     6$        ;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
8$:   MOV      -(R0), -(R3) ;GET A WORD
      DEC      R2       ;SEE IF 1536(DEC) WORDS YET
      BNE     8$        ;BR IF NOT YET
      BR      XDP2      ;GO EXIT
; COME HERE IF MEM MGT
XDP3: JSR      PC, INITMM ;INIT MEM MGT REGISTERS
      ROL      R0       ;GET ADR BITS 13-21 INTO R1 BITS 7-15
      ROL      R1
      ROL      R0
      ROL      R1
```

N10

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 131
GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION

SEQ 0409
SEQ 0130

```

6844 027410 000301          SWAB      R1
6845 027412 006100          ROL      R0
6846 027414 106001          RORB     R1
6847 027416 010137 172354    MOV      R1,2#KIPAR6 ;SET UP PAR6
6848 027422 013701 005550    MOV      XDPSAV,R1 ;GET LO ADRS BITS
6849 027426 042701 160000    BIC      #160000,R1 ;FIX UP R1 TO REFERENCE PAR6
6850 027432 052701 140000    BIS      #140000,R1
6851 027436 005704          16$: TST      R4 ;SEE IF WANT TO SAVE OR GET XXDP
6852 027440 001007          BNE      18$ ;BR IF WANT TO GET XXDP
6853 027442 012305          MOV      (R3)+,R5 ;SAVE A WORD
6854 027444 005237 177572    INC      2#SR0 ;TURN ON MEMORY MANAGEMENT
6855 027450 010521          MOV      R5,(R1)+
6856 027452 005337 177572    DEC      2#SR0 ;TURN OFF MEMORY MANAGEMENT
6857 027456 000406          BR       20$
6858 027460 005237 177572    18$: INC      2#SR0 ;TURN ON MEMORY MANAGEMENT
6859 027464 012105          MOV      (R1)+,R5 ;GET A WORD
6860 027466 005337 177572    DEC      2#SR0 ;TURN OFF MEMORY MANAGEMENT
6861 027472 010523          MOV      R5,(R3)+
6862 027474 032701 020000    20$: BIT      #BIT13,R1 ;SEE IF OVERFLOW TO PAGE 7
6863 027500 001405          BEQ      22$ ;BR IF NOT
6864 027502 042701 020000    BIC      #BIT13,R1 ;SET PAGE = 6 AGAIN
6865 027506 062737 000200 172354 ADD      #200,2#KIPAR6 ;UPDATE PAR BY 4K
6866 027514 005302          22$: DEC      R2 ;SEE IF 1536 WORDS YET
6867 027516 001347          BNE      16$ ;BR IF NOT YET
6868 027520 000713          BR       XDP2 ;BR TO RETURN
6869
6870
6871
6872 ;*****
6873 ;SBTTL INITSS - INITIALIZE SUBSYSTEM
6874 ;*
6875 ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
6876 ;*AND DOES A SUBSYSTEM CLEAR.
6877 ;* USES - R2,R5
6878 ;* CALL:
6879 ;*
6880 ;* JSR PC,INITSS
6881 ;*****
6882 027522 012737 042164 003046 INITSS: MOV      #ERRHDL,A.ABNL ;SET UP ABNORMAL ERROR RETURN ADDRESS
6883 027530 012737 040716 003044 MOV      #ERRFRE,A.NORM ;SET UP NORMAL RETURN ADDRESS
6884 027536 013702 003036          MOV      RKBAS,R2 ;GET ADDRESS OF RK611 REGISTERS
6885 027542 012705 002630          MOV      #PARM0,R5 ;GET ADDRESS OF PARAMETER BLOCK
6886 027546 105037 003143          CLRB     NORTRY ;CLEAR "NO-RETRY" FLAG
6887 027552 004737 027622          JSR      PC,CLRPRM ;CLEAR DRIVER INPUT PARAMETERS
6888 027556 112765 000177 000001 MOVVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6889 027564 004737 040316          JSR      PC,DRVCL ;DO SUBSYSTEM CLEAR
6890 027570 113765 005510 000000 MOVVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
6891 027576 113737 005510 002714 MOVVB    DRIVE,PARM1 ;SET DRIVE NO. IN ALTERNATE P.B.
6892 027604 113765 003125 000007 MOVVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
6893 027612 153765 003124 000007 BISB     TYPFMT,P.CS1H(R5) ;SET DRV TYPE BIT
6894 027620 000207          RTS      PC ;SUBROUTINE EXIT
6895
6896
6897
6898 ;*****
6899 ;SBTTL CLRPRM - CLEAR DRIVER INPUT PARAMETERS

```

B11

6900
6901
6902
6903
6904
6905 027622 010046
6906 027624 010546
6907 027626 010500
6908 027630 062705 000016
6909 027634 005020
6910 027636 020005
6911 027640 001375
6912 027642 012605
6913 027644 012600
6914 027646 000207
6915
6916
6917
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934 027650 105037 003124
6935 027654 004737 027522
6936 027660 113765 005510 000000
6937
6938 027666 001012
6939 027670 122737 000013 000041
6940 027676 001006
6941 027700 105737 003116
6942 027704 001003
6943 027706 104401 010134
6944 027712 000517
6945 027714 042712 000100
6946 027720 113762 005510 000010
6947 027726 105737 003124
6948 027732 001003
6949 027734 012712 000001
6950 027740 000402
6951 027742 012712 002001
6952 027746 005037 005542
6953 027752 005237 005542
6954 027756 001375
6955 027760 032762 001000 000010

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

CLRPRM: MOV RO, -(SP) ;SAVE RO
MOV RS, -(SP) ;SAVE RS
MOV R5, RO ;GET PARAMETER BLOCK ADDRESS
ADD #P.CS1, R5 ;COMPUTE LIMIT ADDRESS
1\$: CLR (R0)+ ;CLEAR A WORD IN PARAMETER BLOCK
CMP RO, R5 ;SEE IF DONE YET
BNE 1\$;BR IF NOT DONE YET
MOV (SP)+, R5 ;RESTORE RS
MOV (SP)+, RO ;RESTORE RO
RTS PC ;RETURN

;
; *****
; SBTTL SCNDRV - SCAN DRIVE FOR STATUS
; *THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
; *THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
; *AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
; *OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
; *MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
; *A RET, RN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
; *AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
; *MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
; *ALSO, IF RUNNING FROM AORS 200 AND RK06 IS LOAD MEDIUM,
; *DRIVE 0 WILL BE REJECTED FOR USE.
; * CALL - JSR PC,SCNDRV
; *
; * (ERROR RETURN ADDRESS)
; *

;
; *****
SCNDRV: CLR B TYPFMT ;ASSUME RK06 FOR FIRST TRY
JSR PC, INITSS ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
MOV B, DRIVE, P.DRVN(R5) ;GET DRIVE NUMBER
;SEE IF DRIVE 0 IS XXDP LOAD MEDIUM
BNE 3\$;BR IF NOT DRIVE 0
CMP B, #13, 2#41 ;SEE IF RK06 IS XXDP MEDIUM
BNE 3\$;BR IF NOT
TST B, MDFLAG ;SEE IF 200 START
BNE 3\$;BR IF NOT
TYPE "DROXDP" ;TYPE "DRIVE 0 IS LOAD MEDIUM"
BR 2\$;TAKE ERROR EXIT
3\$: BIC #IE, (R2) ;DISABLE RK06 INTERRUPT
MOV B, DRIVE, RKCS2(R2) ;SET DRIVE NO. IN RKCS2
TST B, TYPFMT ;SEE IF RK07
BNE 5\$;BR IF YES
MOV #GO, (R2) ;ELSE SET GO FOR RK06 IN CS1
BR 7\$
5\$: MOV #<CDT!GO>, (R2) ;SET GO FOR RK07 IN CS1
7\$: CLR SCRACH ;CLEAR STALL CTR
14\$: INC SCRACH ;INCREMENT STALL COUNTER
BNE 14\$;STALL FOR SEVERAL MILLI-SEC
BIT #MDS, RKCS2(R2) ;SEE IF MDS ERROR

```
6956 027766 001417      BEQ      18$      ;BR IF NOT
6957 027770 112765 000101 000001      MOV     #SELDV,P.CMND(R5) ;SET CMND FOR REPORT
6958 027776 011265 000016      MOV     (R2),P.CS1(R5) ;GET RKCS1
6959 030002 004737 050212      JSR     PC,I.CST1 ;GET OTHER RK611 REGISTERS
6960 030006 004737 041704      JSR     PC,REPSUP ;SET UP ERROR REPORT
6961 030012 104052      ERROR 52 ;REPORT MDS ERROR
6962 030014 052737 000200 005504      BIS     #ABORT,RECODE ;SET ABORT FLAG
6963 030022 000137 043760      JMP     ALLTRM ;GO ABORT TESTING
6964 030026 032762 000040 000014 18$:      BIT     #DTYPE,RKER(R2) ;SEE IF DRV TYPE ERROR
6965 030034 001403      BEQ     20$      ;BR IF NO,=RK06
6966 030036 152737 000004 003124      BISB    #B.CDT,TYFMT ;ELSE SET RK07 FLAG
6967 030044 004737 027522      JSR     PC,INITSS ;INIT THE SUBSYSTEM
6968 030050 112765 000141 000001      MOV     #R0STAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
6969 030056 012737 025622 003046      MOV     #NEDHDL,A.ABNL ;SET NED ABNORMAL RETURN ADDRESS
6970 030064 012737 000377 005540      MOV     #377,NEWON ;INIT. ON-LINE INDICATOR
6971 030072 004737 040316      JSR     PC,DRVCAL ;READ ALL DRIVE STATUS
6972      ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
6973 030076 012737 042164 003046      MOV     #ERRHDL,A.ABNL ;RESTORE ABNL RETURN ADDRESS
6974 030104 022737 000377 005540      CMP     #377,NEWON ;SEE IF NED INDICATION ON THIS DRIVE
6975 030112 001425      BEQ     4$      ;BR IF NED NOT SET
6976 030114 113737 005510 007656      MOV     DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
6977 030122 152737 000060 007656      BISB    #'0,BADDRV+6 ;CONVERT TO ASCII
6978 030130 104401 007650      TYPE    ,BADDRV ;TYPE "DRIVE X"
6979 030134 104401 007661      TYPE    ,NXDRV ;TYPE "NON-EXISTENT"
6980 030140 132737 000200 001341      BITB    #BIT7,$ENVM ;SEE IF UNDER APT
6981 030146 001401      BEQ     2$      ;BR IF NO
6982 030150 104123      ERROR 123 ;NED UNDER APT SIZING
6983      ;SERVICE ERRORS HERE
6984 030152 042762 000100 000000 2$:      BIC     #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT
6985 030160 017616 000000      MOV     2(SP),(SP) ;SET UP ERROR RETURN ADDRESS
6986 030164 000207      RTS     PC ;ERROR RETURN
6987      ;SEE IF DRIVE IS READY
6988 030166 032765 000200 000040 4$:      BIT     #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
6989 030174 001013      BNE     6$      ;BR IF DRIVE IS READY
6990 030176 113737 005510 007656      MOV     DRIVE,BADDRV+6 ;GET DRIVE NO. INTO BUF
6991 030204 152737 000060 007656      BISB    #'0,BADDRV+6 ;CONVERT TO ASCII
6992 030212 104401 007650      TYPE    ,BADDRV ;TYPE "DRIVE X"
6993 030216 104401 007675      TYPE    ,NTREDY ;TYPE "NOT READY"
6994 030222 000753      BR      2$      ;TAKE ERROR EXIT
6995      ;SEE IF DRIVE IS WRITE ENABLED
6996 030224 032765 004000 000040 6$:      BIT     #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
6997 030232 001413      BEQ     8$      ;BR IF WRITE LOCK NOT SET
6998 030234 113737 005510 007656      MOV     DRIVE,BADDRV+6 ;GET DRIVE NO.
6999 030242 152737 000060 007656      BISB    #'0,BADDRV+6 ;CONVERT TO ASCII
7000 030250 104401 007650      TYPE    ,BADDRV ;TYPE "DRIVE X"
7001 030254 104401 007707      TYPE    ,WRTLOK ;TYPE "WRITE-LOCKED"
7002 030260 000734      BR      2$      ;TAKE ERROR EXIT
7003      ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
7004 030262 112765 000103 000001 8$:      MOV     #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7005 030270 004737 040316      JSR     PC,DRVCAL ;SET VOLUME VALID
7006 030274 112765 000113 000001      MOV     #RECAL,P.CMND(R5) ;SET RECAL COMMAND
7007 030302 004737 040316      JSR     PC,DRVCAL ;RECALIBRATE DRIVE
7008 030306 112765 000121 000001      MOV     #R0DATA,P.CMND(R5) ;SET READ COMMAND
7009 030314 105737 003124      TSTB    TYFMT ;SEE IF RK07
7010 030320 001004      BNE     25$     ;BR IF YES
7011 030322 012765 000632 000002      MOV     #632,P.CYLN(R5)
```

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 134
SCNDRV - SCAN DRIVE FOR STATUS

SEQ 0412
SEQ 0133

```

7012 030330 000403 BR 27$
7013 030332 012765 001456 000002 25$: MOV #1456,P.CYLN(R5)
7014 030340 112765 000002 000005 27$: MOV #LSTTRK,P.TRCK(R5) ;SET TRACK=2
7015 030346 012765 063554 000010 MOV #RWBUF,P.BAL0(R5) ;BUS ADDRESS
7016 030354 012765 177774 000012 MOV #-4,P.WC(R5) ;READ 4 WORDS
7017 030362 142765 000020 000007 BICB #B,CFMT,P.CS1H(R5) ;SET 22 SECTOR FORMAT
7018 030370 153765 003124 000007 BISB TYPFMT,P.CS1H(R5) ;SET CORRECT DRV TYPE
7019 030376 004737 040316 JSR PC,DRVCL ;READ 4 WORDS OF BAD SECTOR FILE
7020 030402 032737 100000 005504 BIT #ANYDER,RECODE ;SEE IF DATA ERROR
7021 030410 001402 BEQ 10$ ;BR IF OK
7022 030412 104401 012362 TYPE #BAD632 ;TYPE READ ERROR MESSAGE
7023 030416 022737 177777 063562 10$: CMP #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
7024 030424 001013 BNE 12$ ;BR IF NOT ALL 1'S
7025 030426 113737 005510 007656 MOVB DRIVE,BADDRV+6 ;GET DRIVE NO.
7026 030434 152737 000060 007656 BISB #'0,BADDRV+6 ;CONVERT TO ASCII
7027 030442 104401 007650 TYPE ,BADDRV ;TYPE "DRIVE X"
7028 030446 104401 007722 TYPE ,ALNPAK ;TYPE "LOADED WITH ALIGN PACK"
7029 030452 000637 BR 2$ ;TAKE ERROR EXIT
7030 ;ERROR FREE RETURN
7031 030454 062716 000002 12$: ADD #2,(SP) ;FIX UP RETURN PC
7032 030460 000207 RTS PC ;RETURN
7033
7034
7035
7036 ;*****
7037 ;*SETUP - SET UP FOR LOOP ON ERROR
7038 ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
7039 ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!
7040 ;*****
7041 030462 011637 001076 SETUP: MOV (SP),#STACK-2 ;MOVE RETURN PC ON STACK
7042 030466 012706 001076 MOV #STACK-2,SP ;RE-INIT THE STACK POINTER
7043 030472 005037 005504 CLR RECODE ;CLEAR ERROR RECOVERY FLAGS
7044 030476 105037 003126 CLRB ERRCNT ;CLEAR RETRY ERROR COUNT
7045 030502 012705 002630 MOV #PARAM,R5 ;SET PARAM BLK ADRS
7046 030506 112765 000177 000001 MOVB #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
7047 030514 004737 040316 JSR PC,DRVCL ;CLEAR THE SUBSYSTEM
7048 030520 012737 000000 177776 MOV #PR0,#PS ;RE-ESTABLISH PRIORITY 0
7049 030526 000207 RTS PC ;RETURN
7050
7051
7052
7053 ;*****
7054 ;*SBTTL CHKITR - CHECK CURRENT TEST ITERATION NUMBER
7055 ;*THIS SUBROUTINE LOADS THE ITERATION NUMBER FOR THE CURRENT
7056 ;*TEST INTO $TIMES, AND CHECKS IT TO DETERMINE IF THIS TEST
7057 ;*SHOULD BE RUN. IF ITERATION NO. = 0, THE SUBROUTINE RETURNS
7058 ;*TO THE STORED PC (AND THE TEST WILL BE SKIPPED). IF IT IS
7059 ;*NOT 0, THE RETURN IS MADE TO PC+2, AND THE TEST WILL
7060 ;*BE RUN. HOWEVER, ON THE FIRST PASS (QUICK VERIFY) OF A RUN
7061 ;*WHICH STARTED AT ADRS 200, ALL TESTS WILL BE RUN ONCE.
7062
7063 030530 010146 000144 001115 CHKITR: MOV R1, -(SP) ;SAVE R1
7064 030532 112737 000144 001115 MOVB #100,$ERMAX ;SET MAX ERROR CNT TO 100 FOR $SCOPE
7065 030540 105737 003136 TSTB DULACS ;SEE IF DUAL-ACCESS FLAG SET
7066 030544 001006 BNE 3$ ;BR IF YES
7067 030546 105737 003116 TSTB MDFLAG ;SEE IF 200 START

```

E11

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 135
CHKITR - CHECK CURRENT TEST ITERATION NUMBER

SEQ 0413
SEQ 0134

7068	030552	001007				BNE	4\$		;BR IF NOT
7069	030554	005737	001326			TST	\$PASS		;SEE IF FIRST PASS
7070	030560	001004				BNE	4\$		;BR IF NOT
7071	030562	012737	000001	001304	3\$:	MOV	#1,\$TIMES		;SET UP FOR 1 ITERATION
7072	030570	000410				BR	1\$		;GO RUN DUAL-ACCESS TEST
7073	030572	113701	001102		4\$:	MOVB	\$TSTNM,R1		;GET CURRENT TEST NUMBER
7074	030576	005301				DEC	R1		;DECREMENT BY 1
7075	030600	006301				ASL	R1		;DOUBLE IT, TO GET TEST LIST INDEX
7076	030602	016137	005626	001304		MOV	TSTLST(R1),\$TIMES		;LOAD ITERATION NUMBER
7077	030610	001403				BEQ	2\$		;BR IF TEST SHOULD BE SKIPPED
7078	030612	062766	000004	000002	1\$:	ADD	#4,2(SP)		;ADJUST RETURN PC TO RUN THIS TEST
7079	030620	012601			2\$:	MOV	(SP)+,R1		;RESTORE R1
7080	030622	000207				RTS	PC		;RETURN

7081
7082
7083
7084
7085
7086
7087
7088
7089
7090
7091
7092
7093
7094
7095
7096
7097
7098
7099
7100
7101
7102
7103
7104
7105
7106

```

;*****
;SBTTL RDCHRS - READ A STRING OF KBD INPUT CHARS
;THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
;CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
;THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
;RUB-OUT AND (↑) FEATURES ARE PROVIDED.
;IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
;RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
;TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
;TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
;IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
;FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
;IS TAKEN.
;THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
;IN R0.
;
;CALL -      JSR      PC, RDCHRS
;            <CONTROL-C RETURN ADDRESS>
;            <CONTROL-Z RETURN ADDRESS>
;            <CONTROL-U OR ERROR RETURN ADDRESS>
;            RETURN
;*****

```

7107	030624					RDCHRS:			
7108	030624	010146				MOV	R1,-(SP)		;SAVE R1
7109	030626	010246				MOV	R2,-(SP)		;SAVE R2
7110	030630	0050				CLR	R0		;INITIALIZE CHARACTER COUNT
7111	030632	0050				CLR	R1		;INITIALIZE RUB-OUT INDICATOR
7112						;READ A CHARACTER			
7113	030634	104406			2\$:	RDCHR			;READ A CHARACTER
7114	030636	112602				MOVB	(SP)+,R2		;GET CHARACTER INTO R2
7115						;CHECK FOR (↑C)			
7116	030640	122702	000003			CMPB	#003,R2		;SEE IF (↑C) TYPED
7117	030644	001006				BNE	4\$		;BR IF NOT (↑C)
7118	030646	104401	012426			TYPE	CNTRLC		;ECHO (↑C)
7119	030652	017666	000004	000004	3\$:	MOV	#4(SP),4(SP)		;PUT RETURN ADDRESS ON STACK
7120	030660	000523				BR	24\$		;BR TO TAKE EXIT
7121						;CHECK FOR (↑Z)			
7122	030662	122702	000032		4\$:	CMPB	#032,R2		;SEE IF (↑Z) TYPED
7123	030666	001006				BNE	6\$		;BR IF NOT (↑Z)

Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380</
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----------

G11

7180
7181
7182
7183
7184
7185
7186
7187
7188
7189
7190
7191
7192
7193
7194
7195
7196
7197
7198
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229
7230
7231
7232
7233
7234
7235

031136	104401	012474
031142	010146	
031144	006216	
031146	005216	
031150	104403	
031152	002	
031153	000	
031154	104401	012467
031160	016146	005626
031164	104403	
031166	006	
031167	000	
031170	000207	

```

*****
: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 R1, 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 HIG OCT ;CLEAR HIGH BINARY BITS
CMP #MA, PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
BNE 1$ ;BR IF NOT (MA)
MOV PRMLST+2(R1), HIG OCT ;PUT HI BITS OF PARAM. INTO HIG OCT
1$: MOV #LOWOCT, -(SP) ;PUT ADDRESS OF BINARY VALUE ON STACK
JSR PC, 2$SD820 ;CONVERT BINARY TO OCTAL ASCII
JSR PC, 2$SSUPRS ;TYPE YYYYYYYY, SUPPRESSING LEADING 0'S
RTS PC ;RETURN

```

```

*****
:SBTTL TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15
:*THIS SUBROUTINE TYPES THE LINE : WORD XX = YYYYYY,
:*WHERE XX IS THE NUMBER OF THE CURRENT WORD OF

```

CZR6M00 RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 138
TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15

SEQ 0416
SEQ 0137

H11

7236
7237
7238
7239
7240
7241
7242
7243 031252
7244 031252 104401 012502
7245 031256 010146
7246 031260 006216
7247 031262 104403
7248 031264 002
7249 031265 001
7250 031266 104401 012510
7251 031272 016146 007156
7252 031276 104403
7253 031300 006
7254 031301 001
7255 031302 000207
7256
7257
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272
7273 031304
7274 031304 010046
7275 031306 010146
7276 031310 010246
7277
7278 031312 022761 052120 006152
7279 031320 001127
7280
7281 031322 032737 100000 005760
7282 031330 001523
7283
7284 031332 104401 011167
7285 031336 004737 030624
7286 031342 031610
7287 031344 031620
7288 031346 031404
7289 031350 005700
7290 031352 001512
7291 031354 022737 000115 005272

```
; *USER-DEFINED PATTERN 15, AND YYYYYY IS ITS 16-BIT  
; *VALUE IN OCTAL DIGITS. <CR> AND <LF> ARE NOT TYPED.  
; *ON ENTRY, R1 MUST CONTAIN THE INDEX OF THE CURRENT  
; *WORD IN THE PATTERN 15 TABLE.  
; * CALL - JSR PC, TYPPAT  
; *****
```

```
TYPPAT:      TYPE      WORDSP      ; TYPE "WORD "  
MOV          R1, -(SP)      ; PUT INDEX ONTO STACK  
ASR          (SP)          ; DIVIDE BY TWO FOR WORD NO.  
TYP0S        ; GO TYPE WORD NUMBER  
      .BYTE      2          ; DIGIT COUNT = 2 FOR TYP0C  
      .BYTE      1          ; TELL TYP0S TO TYPE LEADING ZEROS  
TYPE          EQUALS        ; TYPE " = "  
MOV          PAT15(R1), -(SP) ; GET BINARY VALUE OF THIS WORD  
TYP0S        ; 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, PRMNEM(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  
4$:          TYPE          MODFY15      ; ASK WHETHER PATTERN 15 SHOULD BE MODIFIED  
JSR          PC, RDCHRS      ; READ RESPONSE  
24$:          ; (↑C) RETURN ADDRESS  
25$:          ; (↑Z) RETURN ADDRESS  
8$:          ; (↑U) OR ERROR RETURN ADDRESS  
TST          R0             ; SEE IF NULL INPUT  
BEQ          22$            ; BR IF MODIFICATION NOT REQUESTED  
CMP          #'M, BUFF0     ; SEE IF (M) TYPED
```

CZR6M00 RK611/06 SS VERIF 1
CZR6M0.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 139
MODP15 - MODIFY USER-DEFINED PATTERN 15

SEQ 0417
SEQ 0138

7292	031362	001405			BEQ	6\$		;BR IF MODIFICATION REQUESTED
7293	031364	104401	005272		TYPE	,BUFFO		;ECHO BAD INPUT
7294	031370	104401	001314		TYPE	,SQUES		
7295	031374	000756			BR	4\$		;GO ASK AGAIN
7296					;MODIFY PATTERN 15			
7297	031376	104401	011145		6\$: TYPE	,SELFP15		;TYPE "MODIFY PATTERN 15"
7298	031402	005001			CLR	R1		;INITIALIZE WORD INDEX
7299	031404	004737	031252		8\$: JSR	PC, TYPPAT		;TYPE CURRENT WORD AND VALUE
7300	031410	104401	012474		TYPE	,SPACE1		;TYPE A SPACE
7301	031414	104401	012514		TYPE	,PROMPT		;TYPE ASTERISK AND SPACE
7302					;READ AND CHECK	INPUT, IF ANY		
7303	031420	004737	030624		JSR	PC, RDCHRS		;READ NEW DATA PATTERN WORD
7304	031424	031610			24\$			; (1C) RETURN ADDRESS FOR RDCHRS
7305	031426	031620			26\$			; (1Z) RETURN ADDRESS FOR RDCHRS
7306	031430	031404			8\$			; (1U) OR ERROR RETURN ADDR. FOR RDCHRS
7307	031432	005700			TST	RO		;SEE IF ANY INPUT
7308	031434	001006			BNE	12\$		;BR IF ANY INPUT
7309	031436	062701	000002		10\$: ADD	#2, R1		;INCREMENT WORD INDEX
7310	031442	022701	000040		CMP	#32, R1		;SEE IF ALL DONE
7311	031446	001454			BEQ	22\$		;BR IF DONE, TO RETURN
7312	031450	000755			BR	8\$		;CONTINUE WITH NEXT WORD
7313	031452	022737	000041	005272	12\$: CMP	#1, BUFFO		;SEE IF (!) TYPED
7314	031460	001431			BEQ	16\$		;BR TO PROPAGATE CURRENT VALUE
7315	031462	005002			CLR	R2		;INIT. (!) INDICATOR
7316	031464	122760	000041	005271	CMPB	#1, BUFFO-1(R0)		;SEE IF LAST CHAR IN BUF IS (!)
7317	031472	001004			BNE	14\$		;BR IF NOT (!)
7318	031474	105060	005271		CLRB	BUFFO-1(R0)		;INSERT TERMINATOR BYTE
7319	031500	005300			DEC	R0		;DECREMENT CHAR COUNT
7320	031502	005202			INC	R2		;SET (!) INDICATOR
7321	031504	022700	000006		14\$: CMP	#6, R0		;SEE HOW MANY CHARS NOW
7322	031510	002426			BLT	20\$		;BR IF TOO MANY
7323	031512	012746	005272		MOV	#BUFFO, -(SP)		;GET BUF. ADDR. ON STACK FOR OCTBIN
7324	031516	004737	052464		JSR	PC, OCTBIN		;CHECK DIGITS AND CONVERT TO BINARY
7325	031522	031566			20\$			;ERROR RETURN ADDRESS FOR OCTBIN
7326	031524	012600			MOV	(SP)+, R0		;GET BINARY VALUE
7327	031526	005737	052616		TST	\$HI OCT		;SEE IF VALUE EXCEEDS 16 BITS
7328	031532	001015			BNE	20\$		;BR TO ECHO BAD INPUT
7329	031534	010061	007156		MOV	R0, PAT15(R1)		;PUT NEW WORD VALUE INTO TABLE
7330	031540	005702			TST	R2		;SEE IF (!) WAS TYPED
7331	031542	001735			BEQ	10\$		;BR IF (!) WAS NOT TYPED
7332					;PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE			
7333	031544	022701	000036		16\$: CMP	#30, R1		;SEE IF ALL DONE YET
7334	031550	001413			BEQ	22\$		;BR IF DONE
7335	031552	016161	007156	007160	MOV	PAT15(R1), PAT15+2(R1)		;PROPAGATE TO NEXT WORD
7336	031560	062701	000002		ADD	#2, R1		;INCREMENT WORD INDEX
7337	031564	000767			BR	16\$		;LOOP UNTIL DONE
7338					;ECHO BAD INPUT			
7339	031566	104401	005272		20\$: TYPE	,BUFFO		;ECHO BAD INPUT
7340	031572	104401	001314		TYPE	,SQUES		;TYPE <?> AND <CR>, <LF>
7341	031576	000702			BR	8\$		;BR TO ASK AGAIN
7342					;NORMAL RETURN			
7343	031600	012602			22\$: MOV	(SP)+, R2		;RESTORE R2
7344	031602	012601			MOV	(SP)+, R1		;RESTORE R1
7345	031604	012600			MOV	(SP)+, R0		;RESTORE R0
7346	031606	000207			RTS	PC		;RETURN
7347					; (1C) RETURN			

J11

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 140
MODP15 - MODIFY USER-DEFINED PATTERN 15SEQ 0418
SEQ 0139

```

7348 031610 012766 013732 000006 24$: MOV #DRVTST,6(SP) ;PC FOR (1C) RETURN
7349 031616 000770 BR 22$ ;BR TO RETURN
7350 : (1Z) RETURN
7351 031620 012766 015344 000006 26$: MOV #ASKMDE,6(SP) ;PC FOR (1Z) RETURN
7352 031626 000764 BR 22$ ;BR TO RETURN
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368
7369
7370
7371

```

```

;*****
;SBTTL CHKPRM - CHECK VALUE OF INPUT PARAMETER
;THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
;HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
;LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
;INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
;TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
;IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
; (PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
;THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
;VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
;CALL - JSR PC,CHKPRM
; <ERROR RETURN ADDRESS>
;RETURN
;*****

```

```

7372 031630 104407 CHKPRM: SAVREG ;SAVE R0-R5
7373 031632 010102 MOV R1,R2 ;GET COPY OF INDEX
7374 031634 006302 ASL R2 ;DOUBLE IT
7375 031636 022761 040515 006152 CMP #MA,PRMNM(R1) ;SEE IF PARAMETER IS (MA)
7376 031644 001422 BEQ 20$ ;BR IF (MA)
7377 031646 005737 005500 TST HIGOCT ;SEE IF HIGH BITS = 0
7378 031652 001403 BEQ 18$ ;BR IF ZERO
7379 031654 017616 000000 16$: MOV 2(SP), (SP) ;GET ERROR RETURN PC
7380 031660 000475 BR 30$ ;GO TO RETURN
7381 :CHECK VALIDITY OF 16-BIT PARAMETER VALUE
7382 031662 023762 005476 006042 18$: CMP LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
7383 031670 103771 BLO 16$ ;BR IF INPUT VALUE TOO SMALL
7384 031672 023762 005476 006044 CMP LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
7385 031700 101365 BHI 16$ ;BR IF INPUT VALUE IS TOO LARGE
7386 :UPDATE 16-BIT PARAMETER VALUE IN LIST
7387 031702 013761 005476 005732 MOV LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
7388 031710 000457 BR 28$ ;BR TO RETURN
7389 :CHECK VALIDITY OF 32-BIT PARAMETER VALUE
7390 031712 032737 000001 005476 20$: BIT #BIT0,LOWOCT ;SEE IF MA IS ODD
7391 031720 001355 BNE 16$ ;BR IF MA IS ODD
7392 031722 023762 005500 006044 CMP HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
7393 031730 103751 BLO 16$ ;BR IF HIGH WORD IS TOO SMALL
7394 031732 001004 BNE 24$ ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
7395 031734 023762 005476 006042 CMP LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
7396 031742 103744 BLO 16$ ;BR IF LOW WORD IS TOO SMALL
7397 031744 023762 005500 006050 24$: CMP HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
7398 031752 101340 BHI 16$ ;BR IF HIGH WORD TOO BIG
7399 031754 001004 BNE 26$ ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
7400 031756 023762 005476 006046 CMP LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
7401 031764 101333 BHI 16$ ;BR IF LOW WORD IS TOO LARGE
7402 :UPDATE 32-BIT PARAMETER VALUE IN LIST
7403 031766 013761 005476 005732 26$: MOV LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST

```

K11

7404 031774 013761 005500
7405
7406 032002 004737 032060
7407 032006 104401 010710
7408 032012 010037 003176
7409 032016 010137 003200
7410 032022 012746 003176
7411 032026 004737 053422
7412 032032 004737 053736
7413 032036 104401 001315
7414 032042 062766 000002
7415 032050 062716 000002
7416 032054 104410
7417 032056 000207
7418
7419
7420
7421
7422
7423
7424
7425 032060 013700 006126
7426 032064 013701 006130
7427 032070 163700 005762
7428 032074 005601
7429 032076 163701 005764
7430 032102 100413
7431 032104 000241
7432 032106 006001
7433 032110 006000
7434 032112 062700 000001
7435 032116 005501
7436 032120 005701
7437 032122 001403
7438 032124 005000
7439 032126 012701 000001
7440 032132 000207
7441
7442
7443
7444
7445
7446
7447
7448
7449 032134 104407
7450 032136 004737 027522
7451 032142 112765 000141
7452 032150 004737 040316
7453 032154 104401 011374
7454 032160 104401 011410
7455 032164 016501 000054
7456 032170 012704 053524
7457 032174 010446
7458 032176 012703 000003
7459 032202 006101

005734
;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
MOV HIGOC PRLST+2(R1) ;PUT HIGH WORD INTO LIST
JSR PC, MXWRDC ;GET WORD COUNT IN R1-RO
TYPE MAWRDC ;TYPE "MAX WORD COUNT = "
MOV RO, SUMLO1
MOV R1, SUMHI1
MOV #SUMLO1, -(SP) ;PUT POINTER ON STACK
JSR PC, \$OB20 ;CONVERT TO OCTAL
JSR PC, \$SUPRS ;TYPE IT
TYPE \$CRLF ;TYPE <CR>, <LF>
ADD #2, 14(SP) ;INCREMENT R1 INDEX
ADD #2, (SP) ;GET NORMAL RETURN PC
RESREG ;RESTORE RO-R5
RTS PC ;RETURN
000014
28\$:
30\$:
;*****
;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, RO ;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
MOV #RDSTAT P.CMND(R5) ;SET READ STATUS COMMAND
JSR PC, DRVCAL ;READ STATUS OF THIS DRIVE
TYPE 'DRIV' ;TYPE "DRIVE"
TYPE 'SERNM' ;TYPE "SER. NO."
MOV P.A11(R5), R1 ;GET "A" STATUS BYTE 11
MOV #SOCTVL, R4 ;GET ADDR OF CHAR BUFFER
MOV R4, -(SP) ;STORE IT ON STACK FOR \$SUPRS
MOV #3, R3 ;INIT CHAR COUNT
ROL R1 ;INITIALIZE BIT POSITIONS

L11

7460 032204 006101
7461 032206 006101
7462 032210 006101
7463 032212 006101
7464 032214 006101
7465 032216 010100
7466 032220 042700 177760
7467 032224 052700 000060
7468 032230 110024
7469 032232 005303
7470 032234 001364
7471 032236 105014
7472 032240 004737 053736
7473 032244 104401 001315
7474 032250 104410
7475 032252 000207

```
4$:      ROL      R1
          ROL      R1      ;GET NEXT 4 BITS
          ROL      R1
          ROL      R1
          MOV      R1,R0    ;GET A WORKING COPY
          BIC      #177760,R0 ;CLEAR ALL BUT LOW 4 BITS
          BIS      #0,R0    ;CONVERT A DIGIT TO ASCII
          MOVB     R0,(R4)+ ;PUT ASCII DIGIT INTO CHAR BUFFER
          DEC      R3      ;DECREMENT CHAR COUNT
          BNE      4$      ;BR IF NOT 3 CHARS YET
          CLRB     (R4)    ;INSERT NULL TERMINATOR
          JSR      PC,@$$SUPRS ;TYPE DRIVE SER. NUMBER
          TYPE     ,$$CLRF ;TYPE <CR> AND <LF>
          RESREG   PC      ;RESTORE R0-R5
          RTS      PC      ;RETURN
```

7476
7477
7478
7479
7480
7481
7482
7483
7484 032254 004737 027522
7485 032260 142765 000020 000007
7486 032266 105737 003125
7487 032272 001402
7488 032274 105265 000004
7489 032300 112765 000121 000001
7490 032306 013765 017760 000002
7491 032314 112765 000002 000005
7492 032322 012765 063554 000010
7493 032330 012765 177776 000012
7494 032336 004737 040316
7495 032342 104401 011401
7496 032346 104401 011410
7497 032352 012746 063554
7498 032356 004737 053422
7499 032362 004737 053736
7500 032366 104401 001315
7501 032372 000207

```
;*****
;SBTTL CRTSER - TYPE CARTRIDGE SERIAL NUMBER
;THIS SUBROUTINE TYPES "CART. SER. NO. XXXXXXXXXX" (IN OCTAL),
;WITH LEADING ZEROS SUPPRESSED.
;*****
CRTSER:  JSR      PC,INITSS ;CLEAR S.S. AND PARAMETERS
          BICB     #B.CFMT,P.CS1H(R5) ;SET 22 SECTOR FORMAT
          TSTB     FORMAT ;CHECK THE ACTUAL FORMAT
          BEQ      4$      ;BR IF 22 SECTORS
          INCB     P.SECT(R5) ;IF 20 SECTORS, READ SECTOR 1
          4$:      MOVB  #RDATA,P.CMND(R5) ;SET READ COMMAND
          MOV      LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)/1456(OCT) FOR RK07
          MOVB     #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
          MOV      #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
          MOV      #-2,P.WC(R5) ;SET WORD COUNT TO READ 2 WORDS
          JSR      PC,DRVCLAL ;READ SERIAL NO. IN BSF
          TYPE     ,CART    ;TYPE "CART."
          TYPE     ,SERNM   ;TYPE "SER. NO."
          MOV      #RWBUF,-(SP) ;GET POINTER FOR $DB20
          JSR      PC,@$$SDB20 ;CONVERT BINARY TO OCTAL
          JSR      PC,@$$SUPRS ;TYPE CART. SERIAL NO. IN OCTAL
          TYPE     ,$$CLRF ;TYPE <CR> AND <LF>
          RTS      PC      ;RETURN
```

7502
7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515

```
;*****
;SBTTL SEEKER - PERFORM IMPLICIT OR EXPLICIT SEEK
;THIS SUBROUTINE CHECKS SWR BIT 7, AND IF IT IS 0, AN IMPLICIT
;SEEK VIA THE READ HEADER COMMAND IS PERFORMED. IF THE HEADER
;DOES NOT CONTAIN THE CORRECT CYLINDER NO. AN ERROR IS REPORTED.
;IF SWR BIT 7 = 1, AN EXPLICIT SEEK COMMAND IS PERFORMED.
;ON ENTRY TO THE SUBROUTINE, THE DRIVE AND CYLINDER NUMBERS
;MUST BE PRE-LOADED INTO THE PARAMETER BLOCK.
;CALL JSR PC,SEEKER
;*****
```

M11

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 143
SEEKER - PERFORM IMPLICIT OR EXPLICIT SEEK

SEQ 0421
SEQ 0142

7516	032374	010046			SEEKER: MOV	RO, -(SP)	;SAVE RO
7517	032376	112765	000117	000001	MOVB	#SEEK, P.CMND(R5)	;SET SEEK COMMAND
7518	032404	032777	000200	146526	BIT	#BIT07, 2SWR	;SEE IF EXPLICIT SEEKS REQUESTED
7519	032412	001023			BNE	2\$	;BR IF EXPLICIT SEEKS
7520	032414	112765	000125	000001	MOVB	#RDHEAD, P.CMND(R5)	;SET READ HEADER COMMAND
7521	032422	004737	040316		JSR	PC, DRVCAL	;READ A HEADER
7522	032426	016200	000024		MOV	RKDB(R2), RO	;GET WORD 1 OF ACTUAL HEADER
7523	032432	020065	000002		CMP	RO, P.CYLN(R5)	;COMPARE TO EXPECTED CYLINDER
7524	032436	001413			BEQ	4\$	;BR IF EQUAL
7525	032440	004737	041704		JSR	PC, REPSUP	;STORE PREV AND CURRENT CMNDS
7526	032444	016537	000002	001174	MOV	P.CYLN(R5), \$REG5	;GET GOOD CYL NO.
7527	032452	010037	001176		MOV	RO, \$REG6	;GET BAD CYL NO.
7528	032456	104062			ERROR	62	;CYLINDER MISCOMPARE
7529	032460	000402			BR	4\$	
7530	032462	004737	040316		2\$: JSR	PC, DRVCAL	;PERFORM COMMAND
7531	032466	012600			4\$: MOV	(SP)+, RO	;RESTORE RO
7532	032470	000207			RTS	PC	;RETURN

```

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

```

7544	032472	010046			STALL: MOV	RO, -(SP)	;SAVE RO
7545	032474	010146			MOV	R1, -(SP)	;SAVE R1
7546	032476	032777	000400	146434	BIT	#BIT08, 2SWR	;APPLY RANDOM STALL ?
7547	032504	001407			BEQ	1\$	;BR IF NOT RANDOM
7548	032506	004737	052620		JSR	PC, \$RAND	;GENERATE PSEUDO-RANDOM NUMBER
7549	032512	013700	052720		MOV	\$LONUM, RO	;GET IT INTO RO
7550	032516	006200			ASR	RO	;SCALE IT DOWN
7551	032520	006200			ASR	RO	
7552	032522	000403			BR	2\$	;GO STALL WITH RANDOM NO.
7553	032524	013700	005512		1\$: MOV	STALLS, RO	;GET REQUESTED NO. OF STALLS
7554	032530	001406			BEQ	6\$	;RETURN IF NO STALL REQUIRED
7555	032532	012701	000016		2\$: MOV	#14., R1	;SET CONSTANT FOR 40 US
7556	032536	005301			4\$: DEC	R1	;INNER LOOP COUNTER
7557	032540	001376			BNE	4\$	;INNER LOOP BR
7558	032542	005300			DEC	RO	;OUTER LOOP COUNTER
7559	032544	001372			BNE	2\$	;OUTER LOOP BR
7560	032546	012601			6\$: MOV	(SP)+, R1	;RESTORE R1
7561	032550	012600			MOV	(SP)+, RO	;RESTORE RO
7562	032552	000207			RTS	PC	;RETURN

```

;*****
;SBTTL 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

```

7572
7573 032554 004737 052620
7574 032560 013737 052720 005514
7575 032566 042737 177000 005514
7576 032574 023737 017760 005514
7577 032602 002003
7578 032604 043737 017764 005514
7579 032612 000207

```
*****  
RNDADR. JSR PC,SRAND ;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
```

.SBTTL ROUTINES TO HANDLE KW11-L OR P CLOCK

7588 032614 105737 003132
7589 032620 001004
7590 032622 012777 000100 150320
7591 032630 000207
7592 032632 012777 174575 150314
7593 032640 005737 000166
7594 032644 001403
7595 032646 012777 174060 150300
7596 032654 012777 000131 150270
7597
7598 032662 000207
7599

```
*****  
;CLKON - ENABLE KW11-L OR P CLOCK INTERRUPT  
*****  
CLKON: TSTB PCLKF ;SEE WHICH CLOCK WILL BE USED  
        BNE 1$ ;BR IF P-CLOCK TO BE USED  
        MOV #100,ALKS ;L-CLOCK, ENABLE INTERRUPT  
        RTS PC ;RETURN  
1$: MOV #-1667.,APKSB ;SET 60 HZ. COUNT  
     TST HZ ;DETERMINE LINE FREQUENCY  
     BEQ 3$ ;BR IF 60 HZ.  
     MOV #-2000.,APKSB ;SET 50 HZ. COUNT  
3$: MOV #131,APKS ;ENABLE INTERRUPT,CNT UP,REP. INT.  
     ; 100 KHZ.,AND RUN  
     RTS PC ;RETURN
```

7603 032664 005200
7604 032666 000002
7605
7606
7607

```
*****  
;CLOCK - HANDLE KW11-L OR P CLOCK INTERRUPT  
*****  
CLOCK: INC R0 ;SIGNIFY A CLOCK TICK  
        RTI ;RETURN FROM INTR
```

7609 032670 105737 003132
7610 032674 001003
7611 032676 005077 150246
7612 032702 000207
7613 032704 005077 150242
7614 032710 000207
7615
7616
7617
7618
7619

```
*****  
;CLKOF - DISABLE KW11-L OR P CLOCK INTERRUPT  
*****  
CLKOF: TSTB PCLKF ;SEE WHICH CLOCK USED  
        BNE 1$ ;BR IF P-CLOCK USED  
        CLR ALKS ;DISABLE KW11-L INTR  
        RTS PC ;RETURN  
1$: CLR APKS ;DISABLE KW11-P INTR  
     RTS PC ;RETURN
```

7620
7621
7622
7623
7624
7625
7626
7627

```
*****  
;SBTTL CALBRT - CALIBRATE THE SOFTWARE TIMER  
;THIS SUBROUTINE CALIBRATES THE SOFTWARE TIMER USED IN THE TIMING  
;TESTS. IT CALLS SUBROUTINE "TIMER" TO MEASURE THE TIME BETWEEN  
; 2 TICKS OF THE KW11-L OR P CLOCK. THIS INTERVAL EQUALS THE LINE  
;PERIOD. IN ALL, 128 MEASUREMENTS ARE MADE AND AVERAGED, AND THE  
;RESULTANT CALIBRATION CONSTANT IS LEFT IN TCONHI-TCONLO (IN  
; NANO-SEC). IF THE SUBROUTINE HAPPENS TO TIME OUT WAITING FOR  
;A TICK OF THE CLOCK, A RETURN IS MADE TO THE ADDRESS LISTED  
;IMMED. AFTER THE CALL TO CALBRT.
```

B12

7628
7629
7630
7631
7632 032712 104407
7633 032714 005002
7634 032716 005037 003176
7635 032722 005037 003200
7636 032726 005001
7637 032730 005000
7638
7639 032732 004737 032614
7640 032736 005700
7641 032740 001013
7642 032742 005201
7643 032744 001374
7644 032746 104401 011556
7645 032752 104401 011513
7646 032756 105037 003133
7647 032762 017616 000000
7648 032766 000464
7649
7650 032770 005000
7651 032772 005004
7652 032774 005005
7653 032776 004737 033150
7654 033002 032746
7655
7656 033004 060537 003176
7657 033010 005537 003200
7658 033014 060437 003200
7659 033020 103752
7660 033022 005202
7661 033024 022702 000200
7662 033030 001336
7663
7664 033032 012702 000007
7665 033036 000241
7666 033040 006037 003200
7667 033044 006037 003176
7668 033050 005302
7669 033052 001371
7670
7671 033054 005000
7672 033056 005001
7673 033060 012702 000376
7674 033064 012703 050053
7675 033070 005737 000166
7676 033074 001404
7677 033076 012702 000461
7678 033102 012703 026400
7679 033106 013704 003200
7680 033112 013705 003176
7681 033116 004737 053244
7682 033122 010237 003166
7683 033126 001307

```

; * CALL - JSR PC,CALBRT
; * <ERROR RETURN ADDRESS>
; *****

CALBRT: SAVREG                ;SAVE R0-R5
      CLR R2                  ;INIT CALIBRATION LOOP CONSTANT
      CLR SUMLO1              ;INIT SUM TO 0
      CLR SUMHI1
2$:   CLR R1                  ;INIT TIME-OUT INDICATOR
      CLR R0                  ;INIT CLOCK INTR INDIC.
;GET IN SYNCH WITH CLOCK
      JSR PC,CLKON            ;ENABLE KW11-L OR P CLOCK INTR
4$:   TST R0                  ;SEE IF CLOCK INTR REC'D YET
      BNE 8$                  ;BR IF INTR REC'D
      INC R1                  ;INCREMENT CLOCK TIME-OUT INDICATOR
      BNE 4$                  ;BR IF NO TIME-OUT
6$:   TYPE ,CLKFAL            ;TIMED-OUT ON KW11-L OR P INTERRUPT
      TYPE ,TIMSKP            ;SAY ALL TIMING TESTS WILL BE SKIPPED
      CLRB DOTIM              ;DON'T ALLOW TIMING TESTS
      MOV 2(SP), (SP)         ;FIX UP ERROR RETURN PC
      BR 18$                  ;GO TO RETURN

;RUN TIMER TO NEXT CLOCK TICK
8$:   CLR R0                  ;INIT CLOCK INTR INDIC
      CLR R4                  ;INIT HI BITS OF CYCLE COUNT
      CLR R5                  ;LO BITS
      JSR PC,TIMER            ;COUNT CYCLES TIL NEXT CLOCK INTR
      BNE 6$                  ;ERROR RETURN ADDRESS
;ADD MEASURED CYCLES TO SUM
      ADD R5,SUMLO1           ;ADD MEASUREMENT TO SUM
      ADC SUMHI1
      ADD R4,SUMHI1
      BCS 6$                  ;BR IF SUM OVERFLOW
      INC R2                  ;INCR CALIBRATION LOOP COUNT
      CMP #128.,R2           ;SEE IF 128 MEASUREMENTS MADE YET
      BNE 2$                  ;BR IF NOT YET
;TAKE AVERAGE OF 128 MEASUREMENTS
      MOV #7,R2              ;SET DIVIDE LOOP COUNT
12$:  CLC                     ;DIVIDE SUM BY 2
      ROR SUMHI1
      ROR SUMLO1
      DEC R2                  ;DECREMENTT LOOP COUNT
      BNE 12$                ;BR IF DIVISION NOT DONE YET
;DIVIDE LINE PERIOD IN N-SEC BY TIMER CYCLE COUNT TO GET CALIB. CONST.
      CLR R0                  ;SET DIVIDEND = 16666667 NS
      CLR R1
      MOV #376.,R2
      MOV #1020523,R3
      TST HZ                  ;DETERMINE THE LINE FREQUENCY
      BEQ 16$                ;BR IF 60 HZ.
      MOV #461.,R2           ;SET DIVIDEND = 20000000 NS
      MOV #1011520.,R3
16$:  MOV SUMHI1,R4           ;SET DIVISOR
      MOV SUMLO1,R5
      JSR PC,M.OPID           ;PERFORM THE DIVISION
      MOV R2,TCONHI          ;GET FINAL TIMING CONSTANT
      BNE 6$                  ;BR IF HI BITS NOT 0

```

7684 033130 010337 003164
7685 033134 062716 000002
7686 033140 004737 032670
7687 033144 104410
7688 033146 000207

MOV R3, TCONLO ; GET LO BITS
ADD #2, (SP) ; FIX UP ERROR-FREE RETURN PC
18\$: JSR PC, CLKOF ; DISABLE KWI-L OR P CLOCK INTERRUPT
RESREG ; RESTORE R0-R5
RTS PC ; RETURN

7689
7690
7691
7692

; SBTTL TIMER - SOFTWARE TIMER SUBROUTINE
; THIS SUBROUTINE LOOPS THROUGH THE SOFTWARE TIMING LOOP,
; INCREMENTING THE CYCLE COUNT IN R4-R5 EACH LOOP, AND CHECKS
; THE INTERRUPT INDICATION IN R0, FOR COMPLETION. IF A COUNT
; OF 4,194,304 IS REACHED IN R4-R5, A TIME-OUT ERROR RETURN
; IS MADE TO THE ADDRESS LISTED AFTER THE CALL.
; CALL - JSR PC, TIMER
; (ERROR RETURN ADDRESS)

7693
7694
7695
7696
7697
7698
7699
7700

; *** SPECIAL TIMING LOOP ***
TIMER: ADD #1, R5 ; INCREMENT LO CYCLE COUNT
ADC R4 ; ADD CARRY TO HI CYCLE COUNT
BIT #BIT07, R4 ; SEE IF TIMED-OUT WAITING FOR INTR
BNE 4\$; BR IF TIME-OUT
TST R0 ; SEE IF INTERRUPT RECEIVED
BEQ TIMER ; BR IF INTERRUPT NOT REC'D YET

7701
7702
7703 033150 062705 000001
7704 033154 005504
7705 033156 032704 000200
7706 033162 001005
7707 033164 005700
7708 033166 001770

ADD #2, (SP) ; FIX UP ERROR-FREE RETURN PC
RTS PC ; ERROR-FREE RETURN
4\$: MOV @ (SP), (SP) ; FIX UP ERROR RETURN PC
RTS PC ; ERROR RETURN

7709
7710 033170 062716 000002
7711 033174 000207
7712 033176 017616 000000
7713 033202 000207

7714
7715
7716
7717

; INIVRB - INITIALIZE VARIABLES USED IN TIMING TESTS

INIVRB: CLR BLWMIN ; COUNT OF TIMES BELOW SPEC'D MIN
CLR ABVMX1 ; COUNT OF FORWARD TIMES ABOVE SPEC'D MAX
CLR ABVMX2 ; COUNT OF REVERSE TIMES ABOVE SPEC'D MAX
CLR SUMLO1 ; FORWARD TIME SUM AND AVERAGE
CLR SUMHI1
CLR SUMLO2 ; REVERSE TIME SUM AND AVERAGE
CLR SUMHI2

7718
7719
7720 033204 005037 003170
7721 033210 005037 003172
7722 033214 005037 003174
7723 033220 005037 003176
7724 033224 005037 003200
7725 033230 005037 003202
7726 033234 005037 003204

MOV #-1, MINIL1 ; MIN MEAS'D FORWARD TIME
MOV #-1, MINIH1
CLR MAXIL1 ; MAX MEAS'D FORWARD TIME
CLR MAXIH1
MOV #-1, MINIL2 ; MIN MEAS'D REVERSE TIME
MOV #-1, MINIH2
CLR MAXIL2 ; MAX MEAS'D REVERSE TIME
CLR MAXIH2
RTS PC ; RETURN

7727 033240 012737 177777 003212
7728 033246 012737 177777 003214
7729 033254 005037 003216
7730 033260 005037 003220
7731 033264 012737 177777 003222
7732 033272 012737 177777 003224
7733 033300 005037 003226
7734 033304 005037 003230
7735 033310 000207

7736
7737
7738
7739

D12

```

7740 .SBTTL TIMSEK - MEASURE AN EXPLICIT SEEK TIME
7741 ;*THIS SUBROUTINE MEASURES THE AMOUNT OF TIME REQUIRED TO PERFORM
7742 ;*A SEEK TO THE CYLINDER ADDRESS IN P.CYLN IN THE PARAM BLK. THIS
7743 ;*TIME IS RETURNED IN RO-R1, IN US. IF THE SEEK TIMES-OUT, OR IF
7744 ;*THE TIME CALCULATION OVERFLOWS, A MESSAGE IS TYPED, AND AN ERROR
7745 ;*RETURN IS MADE TO THE ADDRESS LISTED AFTER THE CALL.
7746 ;* CALL - JSR PC,TIMSEK
7747 ;*          (ERROR RETURN ADDRESS)
7748 ;*****
7749 timsek: MOV R2,-(SP) ;SAVE R2
7750         MOV R5,-(SP) ;SAVE R5
7751         CLR RO ;INIT DRIVE INTR INDICATOR
7752         CLR R4 ;INIT HI CYCLE COUNT
7753         MOV #SEEKHD,ARKVEC ;SET SPECIAL RK06 SEEK TIMER HANDLER VECTOR
7754         MOV #SEEK,P.CMND(R5) ;SET SEEK COMMAND
7755         MOV R5,2$ ;SET PARAM BLK ADDRESS FOR DRIVER
7756         CLR R5 ;INIT LO CYCLE COUNT
7757         ADD #RKCS1,R2 ;GET ADR OF RKCS1 IN R2 FOR SEEKHD
7758         JSR PC,STRCMD ;STORE PREV AND CURRENT COMMANDS
7759         JSR PC,C.INIT ;START THE SEEK
7760         .WORD 0 ;PARAM BLK ADR GOES HERE
7761         JSR PC,TIMER ;MEASURE TIME TIL DRIVE ATT'N
7762         B$ ;ERROR RETURN ADDRESS FOR TIMER
7763         JSR PC,CNVTIM ;CONVERT MEAS'D CYCLES TO TIME
7764         B$ ;ERROR RETURN ADR FOR CNVTIM
7765         MOV R2,RO ;PUT HI TIME BITS IN RO
7766         MOV R3,R1 ;PUT LO TIME BITS IN R1
7767         ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
7768         MOV #I.INTR,ARKVEC ;RESTORE RK06 INTR HANDLER
7769         MOV (SP)+,R5 ;RESTORE R5
7770         MOV (SP)+,R2 ;RESTORE R2
7771         MOV #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
7772         JSR PC,DRVCAL ;CLEAR THE SUBSYSTEM
7773         RTS PC ;RETURN
7774         MOV RK06,R2 ;GET RK06 REG ADR
7775         BIC #IE,RKCS1(R2) ;DISABLE RK06 INTERRUPT
7776         JSR PC,REPSUP ;GET PREV AND CURRENT CMNDS
7777         JSR PC,TOPROC ;GATHER STATUS
7778         ERROR 106 ;TIMED-OUT ON SEEK
7779         MOV #4(SP),4(SP) ;FIX ERROR RETURN PC
7780         BR 4$ ;BR TO EXIT
7781 ;*****
7782 ;*SPECIAL INTERRUPT HANDLER FOR SEEKS IN SEEK TIMING TESTS
7783 ;*****
7784 seekhd: MOV (R2),R3 ;GET BITS OF RKCS1
7785         BPL 2$ ;BR IF CERR NOT SET
7786         JMP I.INTR ;LET DRIVER PROCESS THE ERROR
7787         BIT #DI,R3 ;SEE IF DRIVE INTR SET YET
7788         BEQ 4$ ;BR IF NOT YET
7789         INC RO ;SET RK06 INTR INDICATOR
7790         RTI ;RETURN FROM INTERRUPT
7791 4$:
7792
7793
7794 ;*****
7795 ;*F0TERM - COMPUTE A TERM OF THE FORWARD SEEK AVERAGE FORMULA,

```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 148
TIMSEK - MEASURE AN EXPLICIT SEEK TIME

E12

SEQ 0426
SEQ 0147

7796
7797
7798 033514 104407
7799 033516 010002
7800 033520 010103
7801 033522 005004
7802 033524 013705 005542
7803 033530 006305
7804 033532 004737 053204
7805 033536 060337 003204
7806 033542 005502
7807 033544 060237 003202
7808 033550 005537 003200
7809 033554 104410
7810 033556 000207
7811
7812
7813
7814
7815
7816
7817
7818 033560 104407
7819 033562 010002
7820 033564 010103
7821 033566 005004
7822 033570 013705 005542
7823 033574 006305
7824 033576 004737 053204
7825 033602 060337 003230
7826 033606 005502
7827 033610 060237 003226
7828 033614 005537 003220
7829 033620 104410
7830 033622 000207
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840 033624 020063 000002
7841 033630 101006
7842 033632 103402
7843 033634 020113
7844 033636 103003
7845 033640 010063 000002
7846 033644 010113
7847 033646 020063 000006
7848 033652 103410
7849 033654 101003
7850 033656 020163 000004
7851 033662 101404

```
; *AND ADD IT TO THE FORWARD SUM IN SUMLO1-SUMHI1-SUMLO2-SUMHI2.
; *****
FDTERM: SAVREG                                ;SAVE R0-R5
        MOV R0,R2                                ;GET MEASUREMENT INTO R2-R3
        MOV R1,R3
        CLR R4
        MOV SCRACH,R5                            ;GET COEFFICIENT
        ASL R5                                    ;MULTIPLY BY 2
        JSR PC,M.DPIM                            ;MULTIPLY COEFFICIENT
        ADD R3,SUMHI2                            ;ADD TO FORWARD SUM
        ADC R2
        ADD R2,SUMLO2
        ADC SUMHI1
        RESREG                                    ;RESTORE R0-R5
        RTS PC

; *****
; *RVTERM - COMPUTE A TERM OF THE REVERSE AVERAGE FORMULA.
; *AND ADD IT TO THE REVERSE SUM IN MAXI1-MAXIH1-MAXIL2-MAXIH2.
; *****
RVTERM: SAVREG                                ;SAVE R0-R5
        MOV R0,R2                                ;GET MEASUREMENT INTO R2-R3
        MOV R1,R3
        CLR R4
        MOV SCRACH,R5                            ;GET COEFFICIENT
        ASL R5                                    ;MULTIPLY BY 2
        JSR PC,M.DPIM                            ;MULTIPLY COEFFICIENT
        ADD R3,MAXIH2                            ;ADD TO REVERSE SUM
        ADC R2
        ADD R2,MAXIL2
        ADC MAXIH1
        RESREG                                    ;RESTORE R0-R5
        RTS PC

; *****
; *CMPTIM - COMPARE MEAS'D TIME TO MEAS'D MIN AND MAX
; *AND REPLACE, IF NECESSARY. TIME IS IN R0-R1, POINTER
; *TO LO MIN IS IN R3.
; *****
CMPTIM: CMP R0,2(R3)                            ;COMPARE HI BITS TO HI MIN
        BHI 6$                                    ;BR IF > MIN
        BLO 4$                                    ;BR IF < MIN
        CMP R1,(R3)                            ;COMPARE LO BITS TO LO MIN
        BHI 6$                                    ;BR IF > OR = MIN
        MOV R0,2(R3)                            ;SET NEW MIN HI BITS
        MOV R1,(R3)                            ;SET NEW MIN LO BITS
        CMP R0,6(R3)                            ;COMPARE HI BITS TO HI MAX
        BLO 10$                                    ;BR IF < MAX
        BHI 8$                                    ;BR IF > MAX
        CMP R1,4(R3)                            ;COMPARE LO BITS TO LO MAX
        BLOS 10$                                ;BR IF < OR = MAX
```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 149
TIMSEK - MEASURE AN EXPLICIT SEEK TIME

SEQ 0427
SEQ 0148

7852 033664 010063 000006
7853 033670 010163 000004
7854 033674 000207
7855
7856
7857
7858
7859
7860
7861

8\$: MOV R0,6(R3) ;SET NEW MAX HI BITS
MOV R1,4(R3) ;SET NEW MAX LO BITS
10\$: RTS PC ;RETURN

;GETAVG - COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES BY
;DIVIDING TIME SUMS BY NO. OF MEASUREMENTS (PASSED IN R3 ON ENTRY).

GETAVG: SAVREG ;SAVE R0-R5

7862 033676 104407
7863 033700 010305
7864 033702 005000
7865 033704 005001
7866 033706 013702 003200
7867 033712 013703 003176
7868 033716 005004
7869 033720 004737 053244
7870 033724 010237 003200
7871 033730 010337 003176
7872 033734 005000
7873 033736 005001
7874 033740 013702 003204
7875 033744 013703 003202
7876 033750 004737 053244
7877 033754 010237 003204
7878 033760 010337 003202
7879 033764 104410
7880 033766 000207
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894 033770
7895
7896 033770 104401 011722
7897 033774 012701 003212
7898 034000 004737 034210
7899 034004 104401 001315
7900 034010 012701 003216
7901 034014 004737 034234
7902 034020 013746 003172
7903 034024 104405
7904 034026 010337 034034
7905 034032 104401
7906 034034 000000
7907 034036 012701 003176

MOV R3,R5 ;LO DIVISOR = MEASUREMENT COUNT
CLR R0 ;SET DIVIDEND = FORWARD TIME SUM
CLR R1
MOV SUMHI1,R2
MOV SUMLO1,R3
CLR R4 ;HI DIVISOR = 0
JSR PC,M.DPID ;PERFORM DIVISION
MOV R2,SUMHI1 ;STORE FORWARD TIME AVG
MOV R3,SUMLO1
CLR R0 ;SET DIVIDEND = REVERSE TIME SUM
CLR R1
MOV SUMHI2,R2
MOV SUMLO2,R3
JSR PC,M.DPID ;PERFORM DIVISION
MOV R2,SUMHI2 ;STORE REVERSE TIME AVG
MOV R3,SUMLO2
RESREG ;RESTORE R0-R5
RTS PC ;RETURN

;SBTTL TYPTMS - TYPE RESULTS OF SEEK TIMING MEASUREMENTS
;THIS SUBROUTINE TYPES THE MIN,MAX, AND AVG SEEK TIMES
;*(IN DECIMAL) MEASURED IN ONE OF THE SEEK TIMING TESTS
;*IN BOTH THE FORWARD AND REVERSE DIRECTIONS. ALSO TYPED ARE
;*THE MAX SPEC'D TIME AND THE NO. OF SEEKS EXCEEDING IT
;* AND THE TOTAL NO. OF MEASUREMENTS. POINTER TO SPECIFIED
;*MAX MESSAGE MUST BE IN R3 ON ENTRY.

TYPTMS:
;TYPE FORWARD MEASUREMENTS
TYPE FORWARD ;TYPE "***FORWARD DIRECTION***"
MOV #MINI1,R1 ;POINTER TO LO MIN
JSR PC,TYPMIN ;TYPE MIN MEAS'D TIME
TYPE \$CRLF ;TYPE <CR> AND <LF>
MOV #MAXI1,R1 ;POINTER TO LO MAX
JSR PC,TYPMAX ;TYPE MAX MEAS'D TIME
MOV ABVMX1,-(SP) ;GET COUNT OF TIMES ABOVE MAX
TYPDS ;CONVERT AND TYPE IT
MOV R3,4\$;SET POINTER TO SPEC'D MAX MESSAGE
TYPE ;TYPE " OF XXX ABOVE MAX OF XXXXX US "
4\$: .WORD 0 ;SPEC'D MAX MSG POINTER GOES HERE
MOV #SUMLO1,R1 ;SET POINTER TO LO AVG

G12

7908 034042 004737 034260
7909 034046 104401 001315
7910
7911 034052 104401 011740
7912 034056 012701 003222
7913 034062 004737 034210
7914 034066 104401 001315
7915 034072 012701 003226
7916 034076 004737 034234
7917 034102 013746 003174
7918 034106 104405
7919 034110 010337 034116
7920 034114 104401
7921 034116 000000
7922 034120 012701 003202
7923 034124 004737 034260
7924 034130 104401 001315
7925 034134 000207
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940 034136 013702 003166
7941 034142 013703 003164
7942 034146 004737 053204
7943 034152 005700
7944 034154 001002
7945 034156 005701
7946 034160 001403
7947 034162 017616 000000
7948 034166 000207
7949 034170 005004
7950 034172 012705 001750
7951 034176 004737 053244
7952 034202 062716 000002
7953 034206 000207
7954
7955
7956
7957
7958
7959
7960
7961 034210 104401 011756
7962 034214 010146
7963 034216 004737 053542

```

      JSR      PC,TYPAVG      ;TYPE AVG TIME
      TYPE     $CRLF          ;TYPE <CR> AND <LF>
;TYPE REVERSE MEASUREMENTS
      TYPE     REVERSE      ;TYPE "****REVERSE DIRECTION****"
      MOV      $MINIL2,R1    ;POINTER TO LO MIN
      JSR      PC,TYPMIN     ;TYPE MIN MEAS'D TIME
      TYPE     $CRLF          ;TYPE <CR> AND <LF>
      MOV      $MAXIL2,R1    ;POINTER TO LO MAX
      JSR      PC,TYPMAX     ;TYPE MAX MEAS'D TIME
      MOV      ABVMX2,-(SP)   ;GET COUNT OF TIMES ABOVE MAX
      TYPDS     ;CONVERT AND TYPE IT
      MOV      R3,6$         ;GET POINTER TO SPEC'D MAX MESSAGE
      TYPE     " OF XXX ABOVE MAX OF XXXXX US "
6$:      .WORD    0           ;SPEC'D MAX MSG POINTER GOES HERE
      MOV      $SUMLO2,R1    ;SET POINTER TO LO AVG
      JSR      PC,TYPAVG     ;TYPE AVG TIME
      TYPE     $CRLF          ;TYPE <CR> AND <LF>
      RTS      PC            ;RETURN

;*****
;* CNVTIM - CONVERT SOFTWARE TIMER CYCLES TO TIME
;* THIS SUBROUTINE MULTIPLIES THE CYCLE COUNT IN R4-R5 BY THE
;* TIME CONSTANT IN TCONHI-TCONLO, TO OBTAIN TIME IN N-SEC. THIS
;* IS THEN DIVIDED BY 1000 (DEC) TO OBTAIN TIME IN MICRO-SEC.
;* THIS TIME IS RETURNED IN R2-R3. IF THE MULTIPLICATION OVERFLOWS
;* 32 BITS, AN ERROR RETURN IS MADE TO THE ADDRESS LISTED AFTER THE
;* CALL TO CNVTIM.
;* CALL - JSR PC,CNVTIM
;* <ERROR RETURN ADDRESS>
;*****
CNVTIM: MOV      TCONHI,R2    ;SET MULTIPLIER = TIME CONST
      MOV      TCONLO,R3
      JSR      PC,M.DPIM     ;MULTIPLY
      TST      R0            ;MAKE SURE HI 2 WORDS ARE 0
      BNE      4$            ;BR IF NOT 0
      TST      R1
      BEQ      6$
4$:      MOV      2(SP),(SP)   ;SET ERROR RETURN PC
      RTS      PC            ;ERROR RETURN
6$:      CLR      R4          ;SET HI BITS = 0
      MOV      #1000,R5      ;SET LO DIVISOR = 1000 (DEC)
      JSR      PC,M.DPID     ;CONVERT TIME TO US
      ADD      #2,(SP)       ;FIX ERROR-FREE RETURN PC
      RTS      PC            ;ERROR-FREE RETURN

;*****
;* TYPMIN - TYPE MIN MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
;* ON ENTRY.
;*****
TYPMIN: TYPE     MINEQ        ;TYPE "MIN = "
      MOV      R1,-(SP)      ;GET POINTER TO LO TIME
      JSR      PC,$SDB20     ;CONVERT TO DEC

```

CZR6M00 R#611 06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 151
TYPTMS - TYPE RESULTS OF SEEK TIMING MEASUREMENTS

SEQ 0429
SEQ 0150

7964 034222 004737 053736
7965 034226 104401 012003
7966 034232 000207
7967
7968
7969
7970
7971
7972
7973
7974 034234 104401 011765
7975 034240 010146
7976 034242 004737 053542
7977 034246 004737 053736
7978 034252 104401 012003
7979 034256 000207
7980
7981
7982
7983
7984
7985
7986
7987 034260 104401 011774
7988 034264 010146
7989 034266 004737 053542
7990 034272 004737 053736
7991 034276 104401 012003
7992 034302 000207
7993
7994
7995
7996
7997
7998
7999
8000
8001 034304 104407
8002 034306 012701 063554
8003 034312 010021
8004 034314 020127 064554
8005 034320 001374
8006 034322 104410
8007 034324 000207
8008
8009
8010
8011
8012
8013
8014
8015
8016
8017 034326 104407
8018 034330 016546 000004
8019 034334 105065 000005

JSR PC,2#SSUPRS ;TYPE IT
TYPE MICROS ;TYPE "US "
RTS PC

```

*****
; *TYPMAX - TYPE MAX MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
; *ON ENTRY.
*****
TYPMAX: TYPE MAXEQ ;TYPE "MAX = "
        MOV R1,-(SP) ;GET POINTER TO LO TIME
        JSR PC,2#SDB2D ;CONVERT TO DEC
        JSR PC,2#SSUPRS ;TYPE IT
        TYPE MICROS ;TYPE "US "
        RTS PC

```

```

*****
; *TYPAVG - TYPE AVERAGE MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
; *ON ENTRY.
*****
TYPAVG: TYPE AVGEQ ;TYPE "AVG = "
        MOV R1,-(SP) ;GET POINTER TO LO TIME
        JSR PC,2#SDB2D ;CONVERT TO DEC
        JSR PC,2#SSUPRS ;TYPE IT
        TYPE MICROS ;TYPE "US "
        RTS PC

```

```

*****
; * LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF R0 INTO ALL
; *256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
*****
LODSEC: SAVREG ;SAVE R0-R5
        MOV #RWBUF,R1 ;GET ADDRESS OF DATA BUF INTO R1
2$: MOV R0,(R1)+ ;PUT WORD INTO BUFFER
        CMP R1,#RWBUF+512. ;SEE IF 256 WORDS WRITTEN YET
        BNE 2$ ;BR IF NOT DONE YET
        RESREG ;RESTORE R0-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 R0-R5
        MOV P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
        CLRB P.TRCK(R5) ;CLEAR THE TRACK NO.

```

CZR6M00 RK611/06 SS VERIF 1
CZR6M0.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 152
TYPTMS - TYPE RESULTS OF SEEK TIMING MEASUREMENTS

SEQ 0430
SEQ 0151

```

8020
8021 034340 116500 000005
8022 034344 062700 000100
8023 034350 004737 034304
8024 034354 112765 000131 000001
8025 034362 004737 040316
8026 034366 105265 000005
8027 034372 122765 000003 000005
8028 034400 001357
8029 034402 012665 000004
8030 034406 104410
8031 034410 000207
8032
8033
8034
8035
8036
8037
8038
8039 034412 104407
8040 034414 012701 000010
8041 034420 012700 007116
8042 034424 004737 052620
8043 034430 013720 052720
8044 034434 013720 052716
8045 034440 005301
8046 034442 001370
8047 034444 104410
8048 034446 000207
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058
8059
8060
8061
8062 034450 104407
8063 034452 013746 005600
8064 034456 005416
8065 034460 005046
8066 034462 116616 000003
8067 034466 005066 000002
8068 034472 116566 000004 000002
8069 034500 066616 000002
8070 034504 005066 000002
8071 034510 012700 000026
8072 034514 105737 003125
8073 034520 001402
8074 034522 012700 000024
8075 034526 020016

```

```

;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
2$:  MOVB  P.TRCK(R5),RO  ;GET TRACK NO. INTO RO
      ADD  #100,RO        ;ADD 100(OCT) TO TRACK NO.
      JSR  PC,LODSEC      ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
      MOVB  #WRCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
      JSR  PC,DRVCAL      ;PERFORM THE WRITE CHECK
      INCB  P.TRCK(R5)    ;INCREMENT THE TRACK NO.
      CMPB  #3,P.TRCK(R5) ;SEE IF DONE WITH ALL TRACKS
      BNE  2$             ;BR IF NOT DONE YET
      MOV  (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
      RESREG ;RESTORE RO-R5
      RTS  PC             ;RETURN

```

```

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

```

```

LODP14: SAVREG ;SAVE RO-R5
      MOV  #8,R1 ;INIT LOOP COUNTER TO 8.
      MOV  #PAT14,RO ;GET ADDRESS OF PATTERN 14 BUFFER
4$:  JSR  PC,$RAND ;GENERATE 2 16-BIT RANDOM NUMBERS
      MOV  $LONUM,(RO)+ ;PUT ONE NUMBER INTO PATTERN
      MOV  $SHNUM,(RO)+ ;PUT OTHER NO. INTO PATTERN
      DEC  R1 ;SEE IF 16 WORDS LOADED YET
      BNE  4$ ;BR IF NOT YET
      RESREG ;RESTORE RO-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 RO-R5
      MOV  LASTWC,-(SP) ;STORE WORD COUNT
      NEG  (SP) ;MAKE IT POSITIVE
18$:  CLR  -(SP) ;MAKE ROOM ON STACK
      MOVB  3(SP),(SP) ;STORE NO. OF SECTORS TRANSFERRED
      CLR  2(SP) ;CLEAR LOCATION ON STACK
      MOVB  P.SECT(R5),2(SP) ;STORE STARTING SECTOR
      ADD  2(SP),(SP) ;DETERMINE FINAL SECTOR ADDRESS
      CLR  2(SP) ;CLEAR NO. OF TRACKS TRANSFERRED
      MOV  #22,RO ;SET FOR 22 SECTORS
      TSTB  FORMAT ;DETERMINE THE FORMAT
      BEQ  19$ ;BR IF 22 SECTORS
      MOV  #20,RO ;SET FOR 20 SECTORS
19$:  CMP  RO,(SP) ;CHECK FOR SECTOR OVERFLOW

```

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 153
FINADR - COMPUTE FINAL PACK ADDRESS

SEQ 0431
SEQ 0152

8076 034530 101004
8077 034532 160016
8078 034534 005266 000002
8079 034540 000772
8080 034542 112637 005577
8081 034546 005046
8082 034550 116516 000005
8083 034554 066616 000002
8084 034560 005066 000002
8085 034564 122716 000003
8086 034570 101005
8087 034572 162716 000003
8088 034576 005266 000002
8089 034602 000770
8090 034604 112637 005576
8091 034610 066516 000002
8092 034614 011637 005574
8093 034620 005726
8094 034622 104410
8095 034624 000207

```

      BHI      20$      ; NO, CHECK IF SECTOR CORRECT
      SUB      RO, (SP) ; DECREMENT SECTOR COUNT BY 20 OR 22
      INC      2(SP)    ; INCREMENT TRACKS TRANSFERRED
      BR       19$      ; CHECK FOR SECTOR OVERFLOW
20$:  MOV      (SP)+, FINSEC ; STORE FINAL SECTOR
      CLR      -(SP)    ; MAKE ROOM FOR TRACKS TRANSFERRED
      MOV      P, TRACK(R5), (SP) ; STORE STARTING TRACK
      ADD      2(SP), (SP) ; DETERMINE FINAL TRACK ADDRESS
      CLR      2(SP)    ; CLEAR FINAL CYLINDER
21$:  CMP      #3, (SP)  ; CHECK FOR TRACK OVERFLOW
      BHI      22$      ; NO, CHECK FINAL TRACK
      SUB      #3, (SP)  ; DECREMENT TRACK COUNT BY 3
      INC      2(SP)    ; INCR CYL COUNT
      BR       21$      ; CHECK FOR TRACK OVERFLOW
22$:  MOV      (SP)+, FINTRK ; STORE FINAL TRACK
      ADD      P, CYLN(R5), (SP) ; CALCULATE FINAL CYLINDER
      MOV      (SP), FINCYL ; STORE FINAL CYLINDER
      TST      (SP)+    ; CLEAN OFF STACK
      RESREG   ; RESTORE RO-R5
      RTS      PC      ; RETURN

```

8096
8097
8098
8099
8100
8101
8102
8103
8104
8105
8106
8107
8108
8109
8110 034626 104407
8111 034630 013702 005572
8112 034634 005737 005544
8113 034640 001007
8114 034642 012700 006216
8115 034646 012746 000400
8116 034652 012701 063554
8117 034656 000463
8118 034660 005000
8119 034662 032701 000001
8120 034666 001003
8121 034670 005200
8122 034672 006201
8123 034674 000772
8124 034676 006300
8125 034700 006300
8126 034702 006300
8127 034704 006300
8128 034706 006300
8129 034710 062700 006216
8130 034714 012746 000020
8131 034720 013701 005602

```

;*****
;SBTTL LODBUF - LOAD THE READ/WRITE DATA BUFFER
;THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
;* PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
;* OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
;* IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
;* OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
;* ENTRY, ARE LOADED INTO THE BUFFER.
;*****
LODBUF: SAVREG      ; SAVE 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, RO ; SET DATA PATTERN STARTING ADDRESS
      MOV      #256, -(SP) ; SET PATTERN WORD COUNT
      MOV      #RWBUF, R1 ; SET BUFFER ADDRESS
      BR       30$      ; PROCEED
3$:  CLR      RO        ; INIT PATTERN NUMBER
4$:  BIT      #BIT0, R1  ; SEE IF THIS BIT IS SET
      BNE      6$        ; BR IF THIS BIT IS SET
      INC      RO        ; INCREMENT PATTERN NO.
      ASR      R1        ; SHIFT TO EXAMINE NEXT BIT
      BR       4$        ; BR TO CHECK NEXT BIT
6$:  ASL      RO        ; MULTIPLY PATTERN NO. BY 32(DEC)
      ASL      RO
      ASL      RO
      ASL      RO
      ASL      RO
      ADD      #PAT00, RO ; GET ADDRESS OF DESIRED PATTERN
      MOV      #16, -(SP) ; SET PATTERN WORD COUNT
      MOV      PMA, R1   ; SET BUFFER ADDRESS

```

K12

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 154
LODBUF - LOAD THE READ/WRITE DATA BUFFERSEQ 0432
SEQ 0153

8132	034724	005737	055716			TST	\$KT11	;SEE IF MEM MGT PRESENT
8133	034730	100036				BPL	30\$	;BR IF NOT PRESENT
8134						;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE		
8135	034732	013737	003206	172354		MOV	SAVPAR,2*#KIPAR6	;SET UP WORKING PAR
8136	034740	052737	000001	177572		BIS	#BIT0,2*#SR0	;TURN ON MEMORY MANAGEMENT
8137	034746	042701	160000			BIC	#160000,R1	;FORCE RELOCATION THRU KIPAR6
8138	034752	052701	140000			BIS	#140000,R1	
8139	034756	010003			22\$:	MOV	R0,R3	;GET A COPY OF PATTERN ADDRESS
8140	034760	011604				MOV	(SP),R4	;INIT PATTERN WORD COUNT
8141	034762	012321			24\$:	MOV	(R3)+,(R1)+	;LOAD A DATA WORD INTO BUFFER
8142	034764	032701	020000			BIT	#BIT13,R1	;SEE IF OVERFLOW TO NEXT PAGE
8143	034770	001405				BEQ	26\$	;BR IF NO OVERFLOW
8144	034772	062737	000200	172354		ADD	#200,2*#KIPAR6	;INCREMENT PAR BY 4K FOR NEW PAGE
8145	035000	042701	020000			BIC	#BIT13,R1	;SET PAGE = 6 AGAIN
8146	035004	005302			26\$:	DEC	R2	;DECREMENT WORD COUNTER
8147	035006	001403				BEQ	28\$	;BR IF ALL DONE
8148	035010	005304				DEC	R4	;DECREMENT PATTERN WORD COUNT
8149	035012	001363				BNE	24\$	;BR IF NOT DONE WITH PATTERN YET
8150	035014	000760				BR	22\$	;BR TO REPEAT THE PATTERN
8151	035016	042737	000001	177572	28\$:	BIC	#BIT0,2*#SR0	;DISABLE MEMORY MANAGEMENT
8152	035024	000410				BR	44\$	;GO TO RETURN
8153						;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE		
8154	035026	010003			30\$:	MOV	R0,R3	;GET A COPY OF PATTERN ADDRESS
8155	035030	011604				MOV	(SP),R4	;INIT PATTERN WORD COUNT
8156	035032	012321			34\$:	MOV	(R3)+,(R1)+	;LOAD A DATA WORD INTO BUFFER
8157	035034	005302				DEC	R2	;DECREMENT WORD COUNTER
8158	035036	001403				BEQ	44\$	;BR IF ALL DONE
8159	035040	005304				DEC	R4	;DECREMENT PATTERN WORD COUNT
8160	035042	001373				BNE	34\$	;BR IF NOT DONE WITH PATTERN YET
8161	035044	000770				BR	30\$	;BR TO REPEAT THE PATTERN
8162	035046	005726			44\$:	TST	(SP)+	;POP THE STACK
8163	035050	104410				RESREG		;RESTORE R0-R5
8164	035052	000207				RTS	PC	;RETURN

8165
8166
8167
8168
8169
8170
8171
8172
8173
8174
8175
8176
8177
8178
8179
8180
8181
8182
8183
8184
8185
8186
8187

035054	104407		
035056	112737	000040	062105
035064	012737	000007	063446
035072	013702	005572	
035076	005037	005542	
035102	004737	041704	
035106	005737	005544	
035112	001007		
035114	012700	006216	
035120	012746	000400	
035124	012701	063554	

```

;*****
;SBTTL CMPBUF - SOFTWARE COMPARE DATA
;THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
;TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
;PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
;00-15 (QUICK VERIFY DEFAULT DATA TEST).
;IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
;REPEATING DATA PATTERN TO COMPARE AGAINST.
;*****
CMPBUF: SAVREG          ;SAVE R0-R5
MOV B    #40,DH701+38. ;RESTORE ERROR MSG PARAMS
MOV      #7,DF25+2
MOV      WDSXFR,R2      ;SET NO. OF WORDS
CLR      SCRACH         ;CLEAR COMPARE ERROR COUNT
JSR      PC,REPSUP      ;STORE PREV CMND FOR POSS. PRINT
TST      PATRN          ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
BNE      3$             ;BR IF NOT
MOV      #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
MOV      #256,-(SP)      ;SET PATTERN WORD COUNT
MOV      #RWBUF,R1       ;SET BUFFER ADDRESS

```

8188	035130	000466			BR	30\$	; PROCEED
8189	035132	005000			3\$: CLR	RO	; INIT PATTERN NO.
8190	035134	032701	000001		4\$: BIT	#BIT0,R1	; SEE IF THIS BIT IS SET
8191	035140	001003			BNE	6\$	; BR IF THIS BIT IS SET
8192	035142	005200			INC	RO	; INCREMENT PATTERN NUMBER
8193	035144	006201			ASR	R1	; SHIFT TO EXAMINE NEXT BIT
8194	035146	000772			BR	4\$	; BR TO CHECK NEXT BIT
8195	035150	006300			6\$: ASL	RO	; MULTIPLY PATTERN NO. BY 32(DEC)
8196	035152	006300			ASL	RO	
8197	035154	006300			ASL	RO	
8198	035156	006300			ASL	RO	
8199	035160	006300			ASL	RO	
8200	035162	062700	006216		ADD	#PAT00,RO	; GET ADDRESS OF DESIRED PATTERN
8201	035166	012746	000020		MOV	#16,-(SP)	; PATTERN WORD COUNT
8202	035172	013701	005602		MOV	PMA,R1	; SET BUFFER ADDRESS
8203	035176	005737	055716		TST	\$KT11	; SEE IF MEM MGT PRESENT
8204	035202	100041			BPL	30\$	; BR IF NOT PRESENT
8205					; COMPARE LOOP FOR MEM MGT STARTS HERE		
8206	035204	013737	003206	172354	MOV	SAVPAR,2#KIPAR6	; SET UP WORKING PAR
8207	035212	052737	000001	177572	BIS	#BIT0,2#SRO	; TURN ON MEM MGT
8208	035220	042701	160000		BIC	#160000,R1	; FORCE RELOCATION THRU KIPAR6
8209	035224	052701	140000		BIS	#140000,R1	
8210	035230	010003			22\$: MOV	RO,R3	; GET A COPY OF PATTERN ADDRESS
8211	035232	011604			MOV	(SP),R4	; INIT PATTERN WORD COUNT
8212	035234	022321			24\$: CMP	(R3)+,(R1)+	; COMPARE DATA WORD TO PATTERN WORD
8213	035236	001404			BEQ	25\$	; BR IF NO ERROR
8214	035240	013737	172354	001216	MOV	2#KIPAR6,\$REG16	; SET PAR FOR PRINTOUT
8215	035246	000432			BR	35\$	; GO HANDLE ERROR
8216	035250	032701	020000		25\$: BIT	#BIT13,R1	; SEE IF OVERFLOW TO NEXT PAGE
8217	035254	001405			BEQ	26\$	; BR IF NO OVERFLOW
8218	035256	062737	000200	172354	ADD	#200,2#KIPAR6	; INCR PAR BY 4K FOR NEW PAGE
8219	035264	042701	020000		BIC	#BIT13,R1	; SET PAGE = 6 AGAIN
8220	035270	005302			26\$: DEC	R2	; DECREMENT WORD COUNTER
8221	035272	001002			BNE	27\$	; BR IF NOT ALL DONE YET
8222	035274	000137	035740		JMP	54\$	; ALL DONE - GET OUT
8223	035300	005304			27\$: DEC	R4	; DECREMENT PATTERN WORD COUNT
8224	035302	001354			BNE	24\$	; BR IF NOT DONE WITH PATTERN YET
8225	035304	000751			BR	22\$	; BR TO REPEAT THE PATTERN
8226					; COMPARE LOOP FOR NO MEM MGT STARTS HERE		
8227	035306	010003			30\$: MOV	RO,R3	; GET COPY OF PATTERN ADDRESS
8228	035310	011604			MOV	(SP),R4	; INIT PATTERN WORD COUNT
8229	035312	022321			34\$: CMP	(R3)+,(R1)+	; COMPARE DATA WORD TO PATTERN WORD
8230	035314	001002			BNE	31\$	; BR IF COMPARE ERROR
8231	035316	000137	035720		JMP	40\$	; JUMP IF DATA COMPARES OK
8232	035322	105037	062105		31\$: CLRB	DH701+38.	; ADJUST DATA HEADER FOR MSG
8233	035326	012737	000005	063446	MOV	#5,DF25+2	; ADJUST ERROR DATA WORD COUNT
8234					; COMMON COMPARE ERROR HANDLER		
8235	035334	010237	001202		35\$: MOV	R2,\$REG10	; GET WORD NO.
8236	035340	163737	005572	001202	SUB	WD5XFR,\$REG10	
8237	035346	013737	001202	005600	MOV	\$REG10, LASTWC	; GET 2'S COMP OF WORD NO.
8238	035354	004737	034450		JSR	PC,FINADR	; COMPUTE ACTUAL PACK ADRS
8239	035360	104407			SAVREG		; SAVE RO-R5
8240	035362	013700	005574		MOV	FINCYL,RO	; GET CYL
8241	035366	113701	005576		MOVB	FINTRK,R1	; GET TRACK
8242	035372	113702	005577		MOVB	FINSEC,R2	; GET SECTOR
8243	035376	004737	040212		JSR	PC,BDSACK	; SEE IF THIS SECTOR LISTED BAD

8244	035402	104410			RESREG		;RESTORE R0-R5
8245	035404	032737	001000	005504	BIT	#BADSEC, RECODE	
8246	035412	001132			BNE	50\$	;BR IF LISTED- DON'T REPORT ERROR
8247	035414	005737	005542		TST	SCRACH	;CHECK THE ERROR COUNT
8248	035420	001024			BNE	36\$	;BR IF THIS IS NOT FIRST ERROR
8249	035422	105737	003144		TSTB	UBMPRS	;SEE IF UNIBUS MAP PRESENT
8250	035426	001411			BEQ	46\$	;BR IF NOT
8251	035430	013737	005270	001174	MOV	CRMPHD, \$REG5	;GET CURRENT MAP REG 0
8252	035436	013737	005266	001176	MOV	CRMPLO, \$REG6	
8253	035444	104116			ERROR	116	;DATA MISCOMPARE (11/70)
8254	035446	104117			ERROR	117	
8255	035450	000401			BR	48\$	
8256	035452	104034			ERROR	34	;TYPE HEADING FOR ERROR MSG
8257	035454	012737	177777	001174	MOV	#-1, \$REG5	;INIT CYL NO.
8258	035462	005037	001176		CLR	\$REG6	
8259	035466	005037	001200		CLR	\$REG7	
8260	035472	005237	005542		INC	SCRACH	;INCREMENT THE ERROR COUNT
8261	035476	032777	000001	143434	BIT	#BIT0, \$SWR	;SEE IF ALL ERRORS SHOULD BE REPORTED
8262	035504	001004			BNE	38\$	;BR TO REPORT ALL ERRORS
8263	035506	022737	000012	005542	CMP	#10, SCRACH	;SEE IF 10(DEC) ERRORS YET
8264	035514	002511			BLT	54\$	;BR IF ERROR LIMIT EXCEEDED
8265	035516	023737	001174	005574	CMP	\$REG5, FINCYL	;SEE IF DIFFERENT CYL
8266	035524	001010			BNE	42\$	;BR IF YES
8267	035526	123737	001176	005576	CMPB	\$REG6, FINTRK	;SEE IF DIFFERENT TRACK
8268	035534	001004			BNE	42\$	;BR IF YES
8269	035536	123737	001200	005577	CMPB	\$REG7, FINSEC	;SEE IF DIFFERENT SECTOR
8270	035544	001412			BEQ	44\$	;BR IF SAME PACK ADDRESS
8271	035546	013737	005574	001174	MOV	FINCYL, \$REG5	;SET NEW PACK ADRS FOR PRINTOUT
8272	035554	113737	005576	001176	MOVB	FINTRK, \$REG6	
8273	035562	113737	005577	001200	MOVB	FINSEC, \$REG7	
8274	035570	104115			ERROR	115	;TYPE NEW PACK ADDRESS
8275	035572	005437	001202		NEG	\$REG10	;GET WORD NO.
8276	035576	016337	177776	001204	MOV	-2(R3), \$REG11	;GET GOOD DATA
8277	035604	016137	177776	001206	MOV	-2(R1), \$REG12	;GET BAD DATA
8278	035612	013737	001202	001212	MOV	\$REG10, \$REG14	;COMPUTE PHYSICAL ADDRESS
8279	035620	005037	001210		CLR	\$REG13	
8280	035624	006137	001212		ROL	\$REG14	
8281	035630	006137	001210		ROL	\$REG13	
8282	035634	063737	005602	001212	ADD	PMA, \$REG14	
8283	035642	005537	001210		ADC	\$REG13	
8284	035646	063737	005604	001210	ADD	PMA+2, \$REG13	
8285	035654	010137	001214		MOV	R1, \$REG15	;GET VIRT. ADRS FOR PRINTOUT
8286	035660	162737	000002	001214	SUB	#2, \$REG15	
8287	035666	104063			ERROR	63	;TYPE GOOD AND BAD DATA, MEM. ADRS.
8288	035670	032777	000100	143242	BIT	#BIT6, \$SWR	;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8289	035676	001020			BNE	54\$	;BR IF JUST 1 ERROR SHOULD BE REPORTED
8290	035700	005737	005544		TST	PATRN	;SEE IF DEFAULT DATA TEST
8291	035704	001405			BEQ	40\$	;BR IF YES
8292	035706	005737	055716		TST	\$KT11	;SEE IF MEM MGT PRESENT
8293	035712	100002			BPL	40\$	;BR IF NOT PRESENT
8294	035714	000137	035250		JMP	25\$	;GO HANDLE MEM MGT
8295	035720	005302			DEC	R2	;DECREMENT WORD COUNTER
8296	035722	001406			BEQ	54\$	;BR IF ALL DONE
8297	035724	005304			DEC	R4	;DECREMENT PATTERN WORD COUNT
8298	035726	001002			BNE	53\$	;BR IF NOT DONE WITH PATTERN YET
8299	035730	000137	035306		JMP	30\$	;JUMP TO REPEAT THE PATTERN

N12

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 157
CMPBUF - SOFTWARE COMPARE DATASEQ 0435
SEQ 0156

8300 035734 000137 035312
 8301 035740 005737 055716
 8302 035744 100003
 8303 035746 042737 000001 177572
 8304 035754 005726
 8305 035756 104410
 8306 035760 000207
 8307
 8308
 8309
 8310
 8311
 8312
 8313
 8314
 8315
 8316
 8317
 8318
 8319
 8320
 8321
 8322 035762
 8323 035762 026537 000002 017754
 8324 035770 002551
 8325 035772 104407
 8326 035774 005037 003176
 8327 036000 005037 003200
 8328 036004 023765 017760 000002
 8329 036012 001004
 8330 036014 122765 000002 000005
 8331 036022 001537
 8332 036024 105737 003125
 8333 036030 001411
 8334
 8335 036032 012746 024000
 8336 036036 012746 036000
 8337 036042 012746 012000
 8338 036046 012746 000024
 8339 036052 000410
 8340
 8341 036054 012746 026000
 8342 036060 012746 041000
 8343 036064 012746 013000
 8344 036070 012746 000026
 8345
 8346 036074 012603
 8347 036076 116502 000004
 8348 036102 160203
 8349 036104 005002
 8350 036106 012705 000400
 8351 036112 005004
 8352 036114 004737 053204
 8353 036120 010337 003176
 8354 036124 016605 000012
 8355

53\$: JMP 34\$
 54\$: TST \$KT11
 BPL 56\$
 BIC \$BIT0,2\$SR0
 56\$: TST (SP)+
 RESREG
 RTS PC
 ;SEE IF MEM MGT PRESENT
 ;BR IF NOT PRESENT
 ;DISABLE MEM MGT
 ;POP THE STACK
 ;RESTORE R0-R5
 ;RETURN

 ;CHKLIM - CHECK CURRENT DATA TRANSFER LIMITS
 ;THIS SUBROUTINE DETERMINES IF THE NEXT DATA TRANSFER SHOULD BE
 ;ALLOWED WITH THE CURRENT PACK ADDRESS AND WORD COUNT. IF THE
 ;TRANSFER WOULD CAUSE OVERFLOW BEYOND CYL 632, TRACK 1, THE
 ;WORD COUNT IN P.WC(R5) MUST BE SCALED DOWN TO A VALID NUMBER.
 ;IF NO POSSIBLE TRANSFER CAN BE ALLOWED, RETURN IS MADE TO
 ;ADDRESS FOLLOWING THE CALL TO CHKLIM.
 ;CYL 1456 USED FOR RKN7
 ;CALL - JSR PC,CHKLIM
 ; ("NO TRANSFER" RETURN ADDRESS)

 CHKLIM: CMP P.CYLN(R5),LCMS ;SAVE R0-R5
 BLT 34\$;SEE IF SHOULD CHECK ADDR. LIMITS YET
 SAVREG ;BR TO RETURN IF NOT
 CLR SUMLO1 ;SAVE R0-R5
 CLR SUMHI1
 CMP LSTCYL,P.CYLN(R5) ;SEE IF ON LAST CYL
 BNE 8\$;BR IF NOT
 CMPB #2,P.TRCK(R5) ;SEE IF ON TRACK 2, CYL 632/1456
 BEQ 36\$;BR IF CAN'T DO TRANSFER
 8\$: TSTB FORMAT ;DETERMINE THE FORMAT
 BEQ 10\$;BR IF 22(DEC) SECTORS
 ;STORE CONSTANTS FOR 20(DEC) SECTORS
 MOV #10240,-(SP) ;NO. OF WORDS ON 2 TRACKS
 MOV #15360,-(SP) ;NO. OF WORDS PER CYL
 MOV #5120,-(SP) ;NO. OF WORDS PER TRACK
 MOV #20,-(SP) ;NO. OF SECTORS
 BR 12\$
 ;STORE CONSTANTS FOR 22(DEC) SECTORS
 10\$: MOV #11264,-(SP) ;NO. OF WORDS ON 2 TRACKS
 MOV #16896,-(SP) ;NO. OF WORDS PER CYL
 MOV #5632,-(SP) ;NO. OF WORDS PER TRACK
 MOV #22,-(SP) ;NO. OF SECTORS
 ;COMPUTE NO. OF WORDS LEFT ON THIS TRACK
 12\$: MOV (SP)+,R3 ;GET NO. OF SECTORS
 MOVB P.SECT(R5),R2 ;GET CURRENT SECTOR NO.
 SUB R2,R3 ;NO. OF SECTORS LEFT
 CLR R2 ;GET NO. OF WDS IN THESE SECTORS
 MOV #256,R5
 CLR R4
 JSR PC,M.DPIM
 MOV R3,SUMLO1 ;STORE THIS NO.
 MOV 12(SP),R5 ;RESTORE PARAM BLK ADDR
 ;COMPUTE NO. OF WORDS LEFT ON THIS CYLINDER

```

8356 036130 012703 000002      MOV      #2,R3      ; TRACK LIMIT = 2
8357 036134 023765 017760 000002  CMP      LSTCYL,P.CYLN(R5) ; SEE IF ON CYL 632/1456
8358 036142 001001          BNE      14$      ; BR IF NOT
8359 036144 005303          DEC      R3      ; DECREMENT TRACK LIMIT TO 1 FOR CYL 632/1456
8360 036146 116502 000005      14$:  MOVB     P.TRCK(R5),R2 ; GET CURRENT TRACK NO.
8361 036152 160203          SUB      R2,R3      ; GET NO. OF TRACKS LEFT
8362 036154 005002          CLR      R2      ; GET NO. OF WORDS IN THESE TRACKS
8363 036156 012605          MOV      (SP)+,R5 ; NO. OF WDS PER TRACK
8364 036160 004737 053204      JSR      PC,M.OPIM
8365 036164 060337 003176      ADD      R3,SUMLO1 ; ADD WORDS TO TOTAL
8366 036170 016605 000010      MOV      10(SP),R5 ; RESTORE PARAM BLK ADR
8367          ; COMPUTE NO. OF WORDS ON WHOLE CYLINDERS REMAINING
8368 036174 013703 017756      MOV      LCM1,R3 ; CYL LIMIT = 631/1455
8369 036200 166503 000002      SUB      P.CYLN(R5),R3 ; GET NO. OF WHOLE CYLS LEFT
8370 036204 100001          BPL      R3      16$:
8371 036206 005003          CLR      R2
8372 036210 005002          MOV      (SP)+,R5 ; GET NO. OF WDS PER CYL
8373 036212 012605          JSR      PC,M.OPIM ; COMPUTE NO. OF WDS IN THESE WHOLE CYLS
8374 036214 004737 053204      ADD      R3,SUMLO1 ; ADD THESE WDS TO TOTAL
8375 036220 063703 003176      ADC      R2
8376 036224 005502          ADD      SUMHI1,R2
8377 036226 063702 003200      MOV      6(SP),R5 ; RESTORE PARAM BLK ADDR
8378 036232 016605 000006      ; ADD THE NO. OF WDS ON TRACKS 0,1, CYL 632/1456
8379          CMP      LSTCYL,P.CYLN(R5) ; SEE IF CYL = 632/1456
8380 036236 023765 017760 000002  BNE      18$      ; BR IF NOT 632/1456
8381 036244 001002          TST      (SP)+ ; POP STACK
8382 036246 005726          BR      20$
8383 036250 000402          18$:  ADD      (SP)+,R3 ; ADD WDS ON TRKS 0,1, CYL 632/1456
8384 036252 062603          ADC      R2
8385 036254 005502          ; GET DESIRED WORD COUNT
8386          20$:  CLR      R0 ; WORD COUNT HI BITS
8387 036256 005000          MOV      P.WC(R5),R1 ; GET THE DESIRED WORD COUNT
8388 036260 016501 000012      NEG      R1
8389 036264 005401          BNE      22$      ; BR IF WORD COUNT NOT 65,536(DEC)
8390 036266 001001          INC      R0 ; SET HI WORD COUNT = 1
8391 036270 005200          ; SUBTRACT WORD COUNT FROM NO. OF WORDS LEFT TO DETERMINE IF OVERFLOW.
8392          ; NUMBER OF WORDS LEFT IS IN R2-R3, WORD COUNT IS IN R0-R1.
8393          22$:  SUB      R1,R3 ; LO PARTS
8394 036272 160103          SBC      R2,R3
8395 036274 005602          SUB      R0,R2 ; HI PARTS
8396 036276 160002          BPL      R3      ; BR IF WORD COUNT NOT > NO. OF WDS LEFT
8397 036300 100004          ; SCALE DOWN WORD COUNT TO AVOID OVERFLOW (SUBTRACT THE EXCESS)
8398          ADD      R3,R1 ; GET NEW WORD COUNT IN R1
8399 036302 060301          NEG      R1
8400 036304 005401          MOV      R1,P.WC(R5) ; SET NEW WORD COUNT IN PARAM BLOCK
8401 036306 010165 000012      ; RETURN
8402          30$:  RESREG
8403 036312 104410          ADD      #2,(SP) ; RESTORE R0-R5
8404 036314 062716 000002      RTS      PC ; FIX NORMAL RETURN PC
8405 036320 000207          34$:  PC
8406 036322 017616 000000      36$:  MOV      2(SP),(SP) ; EXIT
8407 036326 104410          RESREG ; FIX "NO TRANSFER" RETURN PC
8408 036330 000207          RTS      PC ; RESTORE R0-R5
8409          ; RETURN
8410
8411

```

```
8412      ;*****
8413      ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
8414      ;* IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVST. IF (↑Z)
8415      ;* WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
8416      ;* A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
8417      ;* IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
8418      ;* NO ACTION IS TAKEN.
8419      ;* CALL - JSR PC,CTLOUT
8420      ;* OR - CKEXIT
8421      ;*****
8422      036332 122737 000001 003120 CTLOUT: CMPB #1,TSTING ;SEE IF CURRENTLY RUNNING TESTS
8423      036340 001077      BNE 10$ ;BR IF NOT RUNNING TESTS
8424      036342 005737 005532      TST INTCHR ;SEE IF ANY TTY INPUT
8425      036346 001474      BEQ 10$ ;BR IF NO INPUT
8426      036350 105737 003116      TSTB MDFLAG ;SEE IF DEFAULT MODE RUN
8427      036354 001446      BEQ 12$ ;BR IF YES
8428      036356 122737 000003 005532      CMPB #003,INTCHR ;SEE IF (↑C) TYPED
8429      036364 001033      BNE 4$ ;BR IF NOT (↑C)
8430      036366 012700 013732      MOV #DRVST,RO ;SET RETURN ADDR = DRVST
8431      036372 105737 003134      2$: TSTB XOVLAB ;SEE IF XXDP CURRENTLY OVERLAID
8432      036376 001402      BEQ 6$ ;BR IF NOT
8433      036400 004737 027262      JSR PC,GETXDP ;RESTORE SAVED XXDP, IF NECESSARY
8434      036404 000005      6$: RESET ;RESET ALL DEVICES
8435      036406 005037 005532      CLR INTCHR ;CLEAR TTY CHAR BUFFER WORD
8436      036412 005037 001304      CLR $TIMES ;CLEAR THE ITER. COUNT
8437      036416 105037 003135      CLRB XDPSVD ;CLEAR THE XXDP SAVED FLAG
8438      036422 012737 042164 003046      MOV #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADDRESS
8439      036430 012706 001100      MOV #STACK,$P ;RESET THE STACK
8440      036434 112765 000113 000001      MOVB #RECAL,P.CMND(R5) ;SET RECAL COMMAND
8441      036442 004737 040316      JSR PC,DRVCL ;DO CLEANUP RECALIBRATE
8442      036446 005037 001102      CLR $TSTNM ;CLEAR THE TEST NO.
8443      036452 000110      JMP 2RO ;EXIT FROM TESTS
8444      036454 122737 000032 005532 4$: CMPB #032,INTCHR ;SEE IF (↑Z) TYPED
8445      036462 001016      BNE 7$ ;BR IF NOT (↑Z)
8446      036464 012700 015340      MOV #INPUTP,RO ;SET RETURN ADDR = INPUTP
8447      036470 000740      BR 2$ ;TAKE EXIT
8448      036472 122737 000003 005532 12$: CMPB #003,INTCHR ;SEE IF (↑C) TYPED
8449      036500 001007      BNE 7$ ;BR IF NOT
8450      036502 104401 010121      TYPE ,HLTRQD ;TYPE "HALT REQUESTED"
8451      036506 104401 010071      TYPE ,CNTRDY ;TYPE "PRESS CONT WHEN RDY"
8452      036512 012700 044134      MOV #HLTPRG,RO ;SET HALT ADDRESS
8453      036516 000725      BR 2$ ;TAKE EXIT
8454      036520 122737 000007 005532 7$: CMPB #007,INTCHR ;SEE IF (↑G) TYPED
8455      036526 001002      BNE 8$ ;BR IF NOT (↑G)
8456      036530 004737 026370      JSR PC,GTSWRG ;OPEN SOFTWARE SWR FOR MODIFICATION
8457      036534 004737 026040      8$: JSR PC,PREPKB ;ENABLE KBD INPUT AGAIN
8458      036540 000207      10$: RTS PC ;RETURN
8459
8460
8461
8462      ;*****
8463      ;* WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO DISK AND PERFORM
8464      ;* A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
8465      ;* ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
8466      ;* USED.
8467      ;*****
```

8468 036542 104407
8469
8470 036544 012700 063554
8471 036550 012701 000400
8472 036554 010320
8473 036556 005301
8474 036560 001375
8475
8476 036562 012765 063554 000010
8477 036570 012765 177400 000012
8478
8479 036576 112765 000123 000001
8480 036604 004737 040316
8481
8482 036610 112765 000131 000001
8483 036616 004737 040316
8484 036622 104410
8485 036624 000207
8486
8487
8488
8489
8490
8491
8492 036626 104407
8493 036630 005001
8494 036632 016500 000012
8495 036636 001002
8496 036640 005201
8497 036642 000401
8498 036644 075400
8499 036646 060241
8500 036650 006100
8501 036652 006101
8502 036654 060037 005762
8503 036660 005537 005764
8504 036664 060137 005764
8505 036670 104410
8506 036672 000207
8507
8508
8509
8510
8511
8512
8513
8514 036674
8515 036674 004737 027522
8516 036700 142765 000020 000007
8517 036706 112765 000121 000001
8518 036714 013765 017760 000002
8519 036722 112765 000002 000005
8520 036730 012703 000012
8521 036734 105737 003125
8522 036740 001403
8523 036742 105265 000004

```

WRTSEC: SAVREG          ;SAVE R0-R5
;LOAD THE R/W BUFFER WITH DATA
      MOV      #RWBUF, R0      ;BUFFER ADDRESS
      MOV      #400, R1       ;400(OCT) WORDS
4$:   MOV      R3, (R0)+       ;LOAD A BUFFER WORD
      DEC      R1              ;DECR COUNTER
      BNE     4$              ;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
      MOV      #WRDATA, P.CMND(R5) ;SET WRITE COMMAND
      JSR      PC, DRVCAL        ;WRITE THE DATA
;PERFORM WRITE CHECK
6$:   MOV      #WRTCHK, P.CMND(R5) ;SET WRITE CHECK COMMAND
      JSR      PC, DRVCAL        ;PERFORM WRITE CHECK
      RESREG          ;RESTORE R0-R5
      RTS             PC         ;RETURN

;*****
;INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
;*****
INCRMA: SAVREG          ;SAVE R0-R5
      CLR      R1
      MOV      P.WC(R5), R0      ;GET WORD COUNT
      BNE     4$              ;BR IF NOT 65,536(DEC)
      INC      R1              ;SET HI BIT
      BR      6$              ;CONTINUE
4$:   NEG      R0              ;GET TRUE WORD COUNT
6$:   CLC
      ROL      R0              ;DOUBLE THE WORD COUNT TO GET BYTES
      ROL      R1
      ADD      R0, MA          ;ADD IT TO COMPUTE NEW MA
      ADC      MA+2
      ADD      R1, MA+2
      RESREG          ;RESTORE R0-R5
      RTS             PC         ;RETURN

;*****
;REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
;*****
REDBSF: JSR      PC, INITSS      ;INIT THE S.S.
      BICB     #B.CFMT, P.CS1H(R5) ;SET FOR 22 SECTOR FORMAT
      MOV      #RDATA, P.CMND(R5) ;SET READ DATA COMMAND
      MOV      LSTCYL, P.CYLN(R5) ;SET CYL = 632/1456
      MOV      #2, P.TRACK(R5) ;SET TRACK = 2
      MOV      #10, R3         ;SET FACTORY BSF SECTOR LIMIT
      TSTB     FORMAT
      BEQ      6$              ;BR IF 22 SECTORS
      INCB     P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.

```

```

8524 036746 005203          INC      R3
8525 036750 012765 177400 000012 6$:  MOV      #-400,P.WC(R5) ;SET WORD COUNT FOR 1 SECTOR
8526 036756 012765 004232 000010  MOV      #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
8527 036764 012737 037140 003046 8$:  MOV      #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
8528 036772 105037 003141      CLR      WCEFLG ;INIT THE ERROR FLAG
8529 036776 004737 040316      JSR      PC,DRVCL ;READ THE FACTORY BSF
8530 037002 105737 003141      TSTB     WCEFLG ;SEE IF ANY ERRORS
8531 037006 001413          BEQ      12$ ;BR IF NOT
8532 037010 062765 000002 000004  ADD      #2,P.SECT(R5) ;CHECK NEXT SECTOR
8533 037016 126503 000004      CMPB     P.SECT(R5),R3 ;SEE IF LIMIT EXCEEDED YET
8534 037022 001360          BNE      8$ ;BR IF NOT YET
8535 037024 004737 041704 10$:  JSR      PC,REPSUP ;GATHER STATUS FOR PRINTOUT
8536 037030 104120          ERROR    120 ;ABORTING- BAD BSF READ
8537 037032 000137 044134      JMP      HLTPRG ;HALT- CAN'T PROCEED
8538 037036 012765 003232 000010 12$:  MOV      #BSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
8539 037044 110365 000004      MOV      R3,P.SECT(R5) ;SET STARTING SECTOR NO.
8540 037050 012703 000026      MOV      #22,R3 ;SET 22 SECTOR LIMIT
8541 037054 105737 003125      TSTB     FORMAT ;SEE IF 22 SECTOR FORMAT
8542 037060 001402          BEQ      14$ ;BR IF YES
8543 037062 012703 000023      MOV      #19,R3 ;SET 20 SECTOR LIMIT
8544 037066 012737 037140 003046 14$:  MOV      #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
8545 037074 105037 003141      CLR      WCEFLG ;INIT THE ERROR FLAG
8546 037100 004737 040316      JSR      PC,DRVCL ;READ THE SOFTWARE BSF
8547 037104 105737 003141      TSTB     WCEFLG ;SEE IF ANY ERRORS
8548 037110 001407          BEQ      16$ ;BR IF NOT
8549 037112 062765 000002 000004  ADD      #2,P.SECT(R5) ;CHECK NEXT SECTOR
8550 037120 126503 000004      CMPB     P.SECT(R5),R3 ;SEE IF LIMIT EXCEEDED YET
8551 037124 003760          BLE      14$ ;BR IF NOT YET
8552 037126 000736          BR       10$ ;REPORT ERROR AND ABORT
8553 037130 012737 042164 003046 16$:  MOV      #ERRHDL,A.ABNL ;RESTORE ERROR HANDLER ADRS
8554 037136 000207          RTS      PC ;RETURN
8555
8556 ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
8557 ;*POSSIBLE ERRORS IN READING BAD SECTORS.
8558 037140 032765 130600 000034 BDSCHD: BIT      #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
8559 037146 001002          BNE      4$ ;BR IF A READ ERROR OCCURRED
8560 037150 000137 042164          JMP      ERRHDL ;GO HANDLE OTHER ERROR
8561 037154 105237 003141 4$:  INCB     WCEFLG ;SET ERROR FLAG
8562 037160 000137 044342          JMP      RETNML ;TAKE NORMAL DRIVER RETURN
8563
8564
8565 ;*****
8566 ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
8567 ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
8568 ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
8569 ;*2 BSF'S.
8570 ;*****
8571
8572 037164 104407          TYPBSF: SAVREG ;SAVE R0-R5
8573 037166 005004          CLR      R4 ;INIT BSF FILE INDICATOR
8574 037170 012700 004242      MOV      #BSFACT+10,R0 ;ADRS OF DATA IN FACTORY BSF
8575 037174 104401 011423      TYPE     ,FACTBS ;TYPE "FACTORY"
8576 037200 104401 011450      TYPE     ,BDSCT ;TYPE "BAD SECTORS"
8577 037204 104401 061652 3$:  TYPE     ,DH6041 ;TYPE "CYLNDR TRACK SECTOR"
8578 037210 104401 001315      TYPE     ,$CRLF
8579 037214 005001          CLR      R1 ;INIT LIMIT COUNTER

```

F13

CZR6MDO RK611/06 SS VERIF 1
CZR6MDO.P11 10-JAN-78 11:48MACY11 3LA(1052) 10-JAN-78 12:16 PAGE 162
CMPBUF - SOFTWARE COMPARE DATASEQ 0440
SEQ 0161

8580 037216 005710
8581 037220 100007
8582 037222 005701
8583 037224 001032
8584 037226 104401 011470
8585 037232 104401 001315
8586 037236 000425
8587 037240 012046
8588 037242 104402
8589 037244 104401 012473
8590 037250 011003
8591 037252 105003
8592 037254 000303
8593 037256 010346
8594 037260 104402
8595 037262 104401 012473
8596 037266 111003
8597 037270 010346
8598 037272 104402
8599 037274 104401 001315
8600 037300 005720
8601 037302 005201
8602 037304 020127 000176
8603 037310 002742
8604 037312 005704
8605 037314 001006
8606 037316 005204
8607 037320 012700 003242
8608 037324 104401 011436
8609 037330 000725
8610 037332 104410
8611 037334 000207

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

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

FINMEM: SAVREG ; SAVE R0-R5
MOV P.CYLN(R5), R0 ; GET ORIG. CYL NO.
MOVB P.TRACK(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 R0-R5

8612
8613
8614
8615
8616
8617
8618
8619 037336 104407
8620 037340 016500 000002
8621 037344 116501 000005
8622 037350 116502 000004
8623 037354 005003
8624 037356 026500 000030
8625 037362 001006
8626 037364 126501 000027
8627 037370 001003
8628 037372 126502 000026
8629 037376 001404
8630 037400 005203
8631 037402 004737 037422
8632 037406 000763
8633 037410 000303
8634 037412 010337 005572
8635 037416 104410

G13

8636 037420 000207

RTS PC ;RETURN

8637
8638
8639
8640
8641
8642
8643
8644
8645 037422 012704 000025
8646 037426 105737 003125
8647 037432 001402
8648 037434 012704 000023
8649 037440 005202
8650 037442 020204
8651 037444 003407
8652 037446 005002
8653 037450 005201
8654 037452 020127 000002
8655 037456 003402
8656 037460 005001
8657 037462 005200
8658 037464 000207
8659
8660
8661
8662
8663
8664
8665
8666
8667
8668 037466 104407
8669 037470 004737 037336
8670 037474 016500 000030
8671 037500 116501 000027
8672 037504 116502 000026
8673 037510 004737 037422
8674 037514 010065 000002
8675 037520 110165 000005
8676 037524 110265 000004
8677 037530 013703 005572
8678 037534 062703 000400
8679 037540 060365 000012
8680 037544 005004
8681 037546 006103
8682 037550 006104
8683 037552 060337 005602
8684 037556 005537 005604
8685 037562 060437 005604
8686 037566 013765 005602 000010
8687 037574 013700 005604
8688 037600 042700 177774
8689 037604 150065 000007
8690 037610 005737 055716
8691 037614 100002

```

*****
;INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
;THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
;IN R2.
*****
INCRSC: MOV     #21, R4          ;SET SECTOR LIMIT
        TSTB    FORMAT        ;SEE IF 22 SECTOR FORMAT
        BEQ     4$            ;BR IF YES
        MOV     #19, R4        ;SET SECTOR LIMIT
4$:     INC      R2            ;INCR. SECTOR NO.
        CMP     R2, R4         ;SEE IF SECTOR LIMIT EXCEEDED
        BLE     6$            ;BR IF NOT
        CLR     R2            ;SET SECTOR = 0
        INC     R1            ;INCR. TRACK NO.
        CMP     R1, #2        ;SEE IF TRACK LIMIT EXCEEDED
        BLE     6$            ;BR IF NOT
        CLR     R1            ;SET TRACK = 0
        INC     R0            ;INCR. CYLINDER NO.
6$:     RTS      PC           ;RETURN

```

```

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

```

H13

CZR6MD0 RK611.06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 164
CMPBUF - SOFTWARE COMPARE DATASEQ 0442
SEQ 01638692 037616 004737 026604
8693 037622 104410
8694 037624 000207
8695
8696
8697
8698
8699
8700
870116\$: JSR PC,PREPAR ;UPDATE MEM MGT AND UNIBUS MAP
RESREG ;RESTORE R0-R5
RTS PC ;RETURN8702 037626 104407
8703 037630 012700 002632
8704 037634 012701 005560
8705 037640 016537 000012 005572
8706 037646 005437 005572
8707 037652 012737 063554 005602
8708 037660 005037 005604
8709 037664 005737 005544
8710 037670 001414
8711 037672 013737 005762 005602
8712 037700 013737 005764 005604
8713 037706 000405
8714 037710 104407
8715 037712 012700 005560
8716 037716 012701 002632
8717 037722 012702 000005
8718 037726 012021
8719 037730 005302
8720 037732 001375
8721 037734 104410
8722 037736 000207
8723
8724
8725
8726
8727
8728
8729
8730;*****
;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
GTPRMS: SAVREG ;SAVE R0-R5
MOV #SAVPRS,R0 ;ADRS OF SAVE AREA
MOV #PARMO+2,R1 ;ADRS OF PARAMS
GTO: MOV #5,R2 ;SET FOR 5 WORDS
6\$: MOV (R0)+,(R1)+ ;MOVE A WORD
DEC R2 ;SEE IF DONE YET
BNE 6\$;BR IF NOT YET
RESREG ;RESTORE R0-R5
RTS PC ;RETURN8731 037740 010446
8732 037742 005004
8733 037744 105037 003141
8734 037750 004737 040316
8735 037754 032737 000100 005504
8736 037762 001403
8737 037764 112737 000001 003141
8738 037772 032737 000002 005504
8739 040000 001406
8740 040002 005204
8741 040004 004737 037466
8742 040010 005765 000012
8743 040014 001355
8744 040016 005704
8745 040020 001411
8746 040022 004737 037710
8747 040026 004737 037626;*****
;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 DRVCL 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,DRVCL ;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 #BSERR,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 WDSXFR,PMA,PMA+2

8748 040032 005737 055716
8749 040036 100002
8750 040040 004737 026604
8751 040044 012604
8752 040046 000207

8\$: TST SKT11 ;SEE IF MEM MGT PRESENT
BPL 8\$;BR IF NOT
JSR PC,PREPAR ;PREPARE MEM MGT AND U.M.
MOV (SP)+,R4 ;RESTORE R4
RTS PC ;RETURN

8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8766

;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.

8767 040050 010237 001266
8768 040054 010337 001270
8769 040060 012637 001272
8770 040064 011402
8771 040066 012437 001264
8772 040072 062737 001000 001264
8773 040100 005003
8774 040102 062702 000010
8775 040106 005712
8776 040110 100430
8777 040112 020012
8778 040114 001406
8779 040116 062702 000004
8780 040122 020237 001264
8781 040126 002021
8782 040130 000766
8783 040132 005722
8784 040134 011237 001302
8785 040140 042737 174377 001302
8786 040146 123701 001303
8787 040152 001402
8788 040154 005722
8789 040156 000753
8790 040160 005203
8791 040162 012246
8792 040164 042716 177700
8793 040170 000746
8794 040172 010346
8795 040174 013702 001266
8796 040200 013703 001270
8797 040204 013746 001272
8798 040210 000204

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

8799
8800
8801
8802
8803

;SBTTL BAD SECTOR CHECK ROUTINE
;THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
;SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
;TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR

```

8804 ;*IS NOT LISTED THE FLAG IS RESET.
8805 ;*****
8806 040212 104407 BDSRCK: SAVREG
8807 040214 012737 004232 040226 MOV #BSFACT,1$ ;SET TABLE TO SEARCH
8808 040222 004437 040050 JSR R4,SRHTBS ;GO SEARCH IT
8809 040226 000000 1$: .WORD ;TABLE ADDRESS GOES HERE
8810 040230 012603 MOV (SP)+,R3 ;GET NUMBER OF BAD SECTORS, IF ANY
8811 040232 001015 BNE 6$ ;IF ANY, GO TEST WHICH ONES
8812 040234 023727 040226 003232 7$: CMP 1$,#BSSOFT ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
8813 040242 001404 BEQ 3$ ;IF YES-EXIT
8814 040244 012737 003232 040226 MOV #BSSOFT,1$ ;SET OTHER TABLE FOR SEARCH
8815 040252 000763 BR 2$ ;GO SEARCH IT
8816 040254 042737 001000 005504 3$: BIC #BADSEC,RECODE ;CLEAR BAD SECTOR BIT
8817 040262 104410 4$: RESREG
8818 040264 000207 RTS PC ;RETURN
8819 040266 022602 6$: CMP (SP)+,R2 ;THIS SECTOR IN TABLE?
8820 040270 001403 BEQ 8$ ;YES-GO SET BIT & EXIT
8821 040272 005303 DEC R3 ;DECREMENT BAD SECTOR COUNT
8822 040274 001374 BNE 6$ ;IF NOT ZERO-CHECK NEXT ENTRY
8823 040276 000756 BR 7$ ;ELSE GO SEARCH OTHER TABLE
8824 040300 052737 001000 005504 8$: BIS #BADSEC,RECODE ;SET BAD SECTOR BIT
8825 040306 005303 9$: DEC R3 ;CLEAR STACK OF OTHER BAD SECTOR
8826 040310 001764 BEQ 4$ ;NUMBER
8827 040312 005726 TST (SP)+
8828 040314 000774 BR 9$ ;EXIT
8829
8830
8831
8832 ;*****
8833 ;SBTTL CALL DRIVER ROUTINE
8834 ;*ENTRY JSR PC,DRVCL
8835 ;* WITH R5 POINTING TO PARAMETER BLOCK
8836 ;*RETURN RTS PC
8837 ;*
8838 ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
8839 ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
8840 ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
8841 ;*BLOCK WHEN THE ROUTINE IS CALLED.
8842 ;*
8843 ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
8844 ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
8845 ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
8846 ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
8847 ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
8848 ;*****
8849 040316 DRVCL:
8850 040316 104407 SAVREG
8851 040320 004737 032472 JSR PC,STALL ;SAVE R0-R5
8852 040324 004737 036332 JSR PC,CTLOUT ;PERFORM A STALL IF REQUIRED
8853 040330 105037 003123 CLRB DONE ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
8854 040334 105037 003137 CLRB DRNAFG ;CLEAR DONE FLAG
8855 040340 004737 040606 JSR PC,STRCMD ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
8856 040344 105737 003136 TSTB DULACS ;STORE PREV AND CURRENT COMMANDS
8857 040350 0015J2 BEQ 26$ ;SEE IF DUAL-ACCESS TEST
8858 040352 005077 140566 CLR #STKS ;BR IF NOT
8859 040356 113762 005510 000010 MOVB DRIVE,RKCS2(R2) ;DISABLE KBD INPUT
;SET DRIVE NO. IN RKCS2

```

K13

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 167
CALL DRIVER ROUTINESEQ 0445
SEQ 0166

8860	040364	012712	000001			MOV	#BIT0,(R2)	;SELECT THE DRIVE
8861	040370	032712	000200		20\$:	BIT	#BIT7,(R2)	;WAIT FOR C. READY
8862	040374	001775				BEQ	20\$	
8863	040376	032762	010000	000010		BIT	#BIT12,RKCS2(R2)	;SEE IF NED ERROR SET
8864	040404	001060				BNE	28\$	;BR IF YES
8865	040406	032762	000001	000012		BIT	#BIT0,RKDS(R2)	;SEE IF DRIVE IS AVAILABLE
8866	040414	001044				BNE	30\$	;BR IF YES
8867	040416	012712	100000			MOV	#100000,(R2)	;ISSUE CONTROLLER CLEAR
8868	040422	113700	005510			MOVB	DRIVE,R0	;GET DRIVE NO.
8869	040426	012703	000100			MOV	#100,R3	;SET OUTER COUNTER
8870	040432	005001			32\$:	CLR	R1	;INIT INNER COUNTER
8871	040434	005201			33\$:	INC	R1	;INCR INNER COUNTER
8872	040436	001027				BNE	22\$	
8873	040440	005303				DEC	R3	;DECR OUTER COUNTER
8874	040442	001373				BNE	32\$	;BR IF NO TIME-OUT YET
8875	040444	052737	000200	005504		BIS	#ABORT,RECODE	;SET ABORT FLAG
8876	040452	112737	000101	005233		MOVB	#SELDRV,PRVCMD+1	;SET PREV CMND = SELECT DRIVE
8877	040460	112765	000176	000001		MOVB	#CONCLR,P.CMND(R5)	;SET CURRENT CMND = CONTR CLEAR
8878	040466	113737	005510	005232		MOVB	DRIVE,PRVCMD	;SET DRIVE NO. FOR PREV. CMND
8879	040474	011265	000016			MOV	(R2),P.CS1(R5)	;GET CURRENT CONTENTS OF RKCS1
8880	040500	004737	050212			JSR	PC,I.CST1	;GO READ CONTROLLER REGS
8881	040504	004737	041704			JSR	PC,REPSUP	;SET UP REGS FOR PRINTOUT
8882	040510	104107				ERROR	107	;TYPE "DRIVE SIEZED BY OTHER PORT"
8883	040512	000137	043760			JMP	ALLTRM	;REPORT DRIVE SIEZED AND HALT
8884	040516	136062	003102	000017	22\$:	BITB	I.DRV(R0),RKASOF	;WAIT FOR DRIVE ATTENTION
8885	040524	001743				BEQ	33\$	;BR IF NO ATT'N YET
8886	040526	113762	005510	000010	30\$:	MOVB	DRIVE,RKCS2(R2)	;SET DRIVE NO. IN RKCS2
8887	040534	012712	000005			MOV	#5,(R2)	;ISSUE DRIVE CLEAR
8888	040540	032712	000200		24\$:	BIT	#BIT7,(R2)	;WAIT FOR CONTROLLER READY
8889	040544	001775				BEQ	24\$	
8890	040546	012712	100000		28\$:	MOV	#100000,(R2)	;ISSUE CONTROLLER CLEAR
8891	040552	004737	026040			JSR	PC,PREPKB	;ENABLE KBD INPUT
8892	040556	010537	040566		26\$:	MOV	R5,4\$	;GET PARAM BLOCK ADDRESS
8893	040562	004737	050570			JSR	PC,C.INIT	;CALL DRIVER
8894	040566	000000			4\$:	.WORD		;P.B. ADDRESS GOES HERE
8895	040570	004737	045140		6\$:	JSR	PC,W.WTCH	;CALL WATCH DOG
8896	040574	105737	003123			TSTB	DONE	;DONE SET?
8897	040600	001773				BEQ	6\$	;NO-LOOP
8898	040602	104410			12\$:	RESREG		;RESTORE R0-R5
8899	040604	000207				RTS	PC	;YES-RETURN
8900								
8901								
8902								
8903								
8904								
8905								
8906	040606	104407						
8907	040610	012701	005232			STRCMD: SAVREG		;SAVE R0-R5
8908	040614	012700	005246			MOV	#PRVCMD,R1	;ADDR OF PREV COMMAND STORAGE
8909						MOV	#COMSTR,R0	;ADDR OF CURRENT COMMAND STORAGE
8910	040620	012021				;STORE PREVIOUS COMMAND		
8911	040622	012021				MOV	(R0)+,(R1)+	
8912	040624	012021				MOV	(R0)+,(R1)+	
8913	040626	012021				MOV	(R0)+,(R1)+	
8914	040630	012021				MOV	(R0)+,(R1)+	
8915	040632	012021				MOV	(R0)+,(R1)+	

8916	040634	105737	003144		TSTB	UBMPRS		;SEE IF UNIBUS MAP PRESENT
8917	040640	001414			BEQ	4\$		;BR IF NOT
8918	040642	013737	005270	005264	MOV	CRMPHO,PRMPHO		;STORE PREV U.B. MAP REG 0
8919	040650	013737	005266	005262	MOV	CRMPLO,PRMPLO		
8920	040656	013737	170202	005270	MOV	2\$MAPH00,CRMPHO		;STORE CURRENT U.B. MAP REG 0
8921	040664	013737	170200	005266	MOV	2\$MAPLO0,CRMPLO		
8922								
8923	040672	012701	005246		4\$:	MOV	COMSTR,R1	
8924	040676	012521			MOV	(R5)+,(R1)+		;R5 POINTS TO DRIVER PARAM BLK
8925	040700	012521			MOV	(R5)+,(R1)+		
8926	040702	012521			MOV	(R5)+,(R1)+		
8927	040704	012521			MOV	(R5)+,(R1)+		
8928	040706	012521			MOV	(R5)+,(R1)+		
8929	040710	012521			MOV	(R5)+,(R1)+		
8930	040712	104410			RESREG			;RESTORE R0-R5
8931	040714	000207			RTS	PC		;RETURN
8932								
8933								
8934								
8935								
8936								
8937								
8938								
8939								
8940								
8941								
8942	040716	152737	000377	003123	ERRFRE:	BISB	377,DONE	;SET THE DONE FLAG
8943	040724	032737	100000	005504	BIT	ANYDER,RECODE		;TEST IF ANY DATA ERROR
8944	040732	0014C3			BEQ	2\$		;IF NO - DO ERROR RECOVERY PRINT TEST
8945	040734	105037	003126		CLRB	ERRCNT		;CLEAR ERROR COUNT
8946	040740	000413			BR	1\$		;EXIT
8947	040742	105737	003126		2\$:	TSTB	ERRCNT	;CHECK IF ANY ERRORS HAVE OCCURRED
8948	040746	001410			BEQ	1\$		;NO - SKIP TO EXIT
8949	040750	005037	001174		CLR	\$REG5		
8950	040754	113737	003126	001174	MOVB	ERRCNT,\$REG5		;GET RETRY COUNT
8951	040762	104101			ERROR	101		;PRINT RETRY SUCCESSFUL MESSAGE
8952	040764	105037	003126		CLRB	ERRCNT		;CLEAR ERROR COUNT
8953	040770	005037	005504		1\$:	CLR	RECODE	;CLEAR RECOVERY FLAGS
8954	040774	000207			RTS	PC		;RETURN
8955								
8956								
8957								
8958								
8959								
8960								
8961								
8962								
8963								
8964								
8965	040776	104407			TPERR:	SAVREG		
8966	041000	105237	003130		INCB	DRVERS		;INCR ERROR COUNT FOR THIS DRIVE
8967	041004	032737	000040	005770	BIT	BIT5,CS		;SEE IF DRIVE SHOULD BE DROPPED
8968								;IF ERROR LIMIT EXCEEDED
8969	041012	001412			BEQ	9\$		;BR IF NOT
8970	041014	123727	003130	000024	CMPB	DRVERS,20.		;SEE IF 20(DEC) ERRORS EXCEEDED
8971	041022	002406			BLT	9\$		;BR IF NOT

8972	041024	105037	003130		CLRB	DRVERS	; CLEAR DRIVE ERROR COUNT
8973	041030	104401	011310		TYPE	, DROPDR	; TYPE "DROPPING DRIVE"
8974	041034	000137	017026		JMP	NEWDRV	; PROCEED TO TEST NEXT DRIVE
8975	041040	032777	020000	140072 9\$:	BIT	#SW13, 2SWR	; INHIBIT ERROR TYPEOUTS?
8976	041046	001402			BEQ	6\$	; BR IF NO
8977	041050	000137	041454		JMP	20\$	
8978	041054	005000		6\$:	CLR	RO	; CLR RO FOR ERROR NUMBER
8979	041056	005005			CLR	RS	; INIT INDENT INDICATOR
8980	041060	005105			COM	RS	
8981	041062	113700	001114		MOVB	\$ITEMB, RO	; ENTER ERROR NUMBER
8982	041066	005300			DEC	RO	; FORM INDEX FOR ERROR TABLE
8983	041070	006300			ASL	RO	
8984	041072	006300			ASL	RO	
8985	041074	006300			ASL	RO	
8986	041076	062700	001400	1\$:	ADD	#ERRTB, RO	; FORM ADDRESS OF ERROR ENTRY
8987	041102	012037	041122		MOV	(RO)+, 2\$	; GET EM POINTER
8988	041106	001406			BEQ	3\$	; BRANCH IF THERE ISN'T ONE
8989	041110	104401	012457		TYPE	, CR2LF	
8990	041114	104401	056354		TYPE	, AS2SP2	; TYPE "** "
8991	041120	104401			TYPE		; TYPE ERROR MESSAGE (EM)
8992	041122	000000		2\$:	.WORD	0	; EM POINTER GOES HERE
8993	041124	012037	041300	3\$:	MOV	(RO)+, 4\$	; GET DH POINTER
8994	041130	001467			BEQ	5\$	; BR IF THERE ISN'T ONE
8995	041132	104401	001315		TYPE	, \$CRLF	
8996	041136	104401	056342		TYPE	, \$TSTMSG	; TYPE " TEST "
8997	041142	013746	001102		MOV	\$TSTNM, -(SP)	; GET TEST NO. ON STACK
8998	041146	104403			TYPOS		; TYPE IT
8999	041150	002			.BYTE	2	; 2 DIGITS
9000	041151	000			.BYTE	0	; SUPPRESS LEADING ZEROS
9001	041152	104401	012457		TYPE	, CR2LF	
9002	041156	032777	010000	137754	BIT	#BIT12, 2SWR	; REPORT DESCRIPTION ONLY ?
9003	041164	001133			BNE	20\$	; BR IF YES
9004	041166	104401	061023		TYPE	, DH105	; TYPE "PREVIOUS COMMAND :"
9005	041172	104401	001315		TYPE	, \$CRLF	
9006	041176	104401	061166		TYPE	, DH101+10	; TYPE PREV COMMAND HEADER
9007	041202	104401	001315		TYPE	, \$CRLF	
9008	041206	012701	000006		MOV	\$R1	; SIX COMMAND VALUES
9009	041212	012702	001236		MOV	\$REG26, R2	; STARTING ADDR OF PREV CMND VALUES
9010	041216	012246		30\$:	MOV	(R2)+, -(SP)	; PUT A WORD ON STACK
9011	041220	104402			TYPOC		; TYPE IT
9012	041222	104401	012473		TYPE	, SPACE2	; TYPE SEPARATORS
9013	041226	005301			DEC	R1	; SEE IF 7 VALUES TYPED YET
9014	041230	001372			BNE	30\$	; BR IF NOT
9015	041232	104401	001315		TYPE	, \$CRLF	
9016	041236	104401	012473		TYPE	, SPACE2	; INDENT
9017	041242	104401	061245		TYPE	, DH102	; TYPE HDR FOR BA DATA
9018	041246	104401	001315		TYPE	, \$CRLF	
9019	041252	104401	012473		TYPE	, SPACE2	; INDENT
9020	041256	012246			MOV	(R2)+, -(SP)	
9021	041260	104402			TYPOC		; TYPE PREV. HI BA BITS
9022	041262	104401	012473		TYPE	, SPACE2	
9023	041266	011246			MOV	(R2), -(SP)	
9024	041270	104402			TYPOC		; TYPE PREV. LO BA BITS
9025	041272	104401	012457		TYPE	, CR2LF	
9026	041276	104401			TYPE		; TYPE DATA HEADER
9027	041300	000000		4\$:	.WORD	0	; DH POINTER GOES HERE

```

9028 041302 104401 001315      TYPE      $CRLF
9029 041306 005005              CLR      R5
9030 041310 032777 010000 137622 5$:      BIT      12,2SWR      ; INIT INDENT INDICATOR
9031 041316 001056              BNE      20$      ; REPORT DESCRIPTION ONLY ?
9032 041320 012001              MOV      (R0)+,R1      ; BR IF YES
9033 041322 001454              BEQ      20$      ; GET DT POINTER
9034 041324 012000              MOV      (R0)+,R0      ; BRANCH IF THERE ARE NONE
9035 041326 012002              MOV      (R0)+,R2      ; GET OF POINTER
9036 041330 112003              10$:      MOV      (R0)+,R3      ; STORE NUMBER OF DH'S
9037 041332 105720              TST      R3      ; GET & STORE NUMBER OF DATA WORDS
9038 041334 005703              TST      R3      ; BUMP PAST FORMAT WORD
9039 041336 001417              BEQ      14$      ; TEST IF ANY DATA FOR THIS HEADER
9040 041340 005705              TST      R5      ; NO - SKIP DATA PRINT
9041 041342 001002              BNE      11$      ; SEE IF SHOULD INDENT
9042 041344 104401 012473      TYPE      $SPACE2      ; BR IF NOT
9043 041350 013146              11$:      MOV      2(R1)+,-(SP)      ; INDENT
9044 041352 104402              TYPOC      ; PUT FIRST DATA WORD ON STACK
9045 041354 005303              DEC      R3      ; TYPE IT
9046 041356 001403              BEQ      12$      ; MORE DATA WORDS
9047 041360 104401 012473      TYPE      $SPACE2      ; NO-BRANCH
9048 041364 000771              BR      11$      ; TYPE SEPARATORS
9049 041366 005702              12$:      TST      R2      ; LOOP
9050 041370 001402              BEQ      14$      ; SEE IF <CR>,<LF> NEEDED
9051 041372 104401 001315      TYPE      $CRLF      ; BR IF NOT
9052 041376 005302              14$:      DEC      R2      ; TYPE IT
9053 041400 003425              BLE      20$      ; MORE DH'S?
9054 041402 012037 041434      15$:      MOV      (R0)+,16$      ; NO-BRANCH
9055 041406 105710              TST      R3      ; GET NEXT DH POINTER
9056 041410 001004              BNE      34$      ; SEE IF ANY DATA FOR HDR
9057 041412 104401 001315      TYPE      $CRLF      ; BR IF YES
9058 041416 005005              CLR      R5      ; SKIP EXTRA LINE
9059 041420 000404              BR      36$      ; RE-INIT INDENT INDICATOR
9060 041422 005105              34$:      COM      R5      ; COMPLEMENT INDENT INDICATOR
9061 041424 001002              BNE      36$      ; BR IF NO INDENT REQUIRED
9062 041426 104401 012473      TYPE      $SPACE2      ; INDENT
9063 041432 104401              36$:      TYPE      0      ; TYPE DH
9064 041434 000000              16$:      .WORD      0      ; DH POINTER GOES HERE
9065 041436 104401 001315      TYPE      $CRLF
9066 041442 105710              TST      (R0)      ; TYPE A DT?
9067 041444 001331              BNE      10$      ; YES-BRANCH
9068 041446 062700 000002      ADJ      #2,R0      ; INCREMENT OF POINTER
9069 041452 000751              BR      14$      ; SEE IF END OF DF BLOCK
9070 041454 104410              20$:      RESREG
9071 041456 000207              RTS      PC
9072                                     ; *****
9073                                     ; SBTTL CONTROLLER ERROR REPORTER ROUTINE
9074                                     ; *ENTRY:      JSR PC, CONERR
9075                                     ; *RETURN:     RTS PC
9076                                     ; *
9077                                     ; *THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
9078                                     ; *REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
9079                                     ; *RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
9080                                     ; *BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
9081                                     ; *AT THIS TIME.
9082                                     ; *****
9083 041460 104407      CONERR: SAVREG      ;SAVE R0-PS

```

```

9084 041462 152737 000377 003123 BISB #377,DONE ;SET DONE FLAG
9085 041470 105237 003126 INCB ERRCNT ;INCREMENT ERROR COUNT
9086 041474 094737 044352 JSR PC, TOPROC ;LOAD RK REGS INTO $REGS
9087 041500 032737 000001 003052 BIT #BIT0,E.CONT ;ERROR 0?
9088 041506 001402 BEQ 1$ ;NO-BRANCH
9089 041510 104064 ERROR 64 ;CLEAR CONT DID NOT CLEAR ERROR
9090 041512 000470 BR 7$
9091 041514 032737 000002 003052 1$: BIT #BIT1,E.CONT ;ERROR 1?
9092 041522 001402 BEQ 2$ ;NO-BRANCH
9093 041524 104065 ERROR 65 ;NO ATTENTION IN ATTENTION SUM REG
9094 041526 000462 BR 7$
9095 041530 032737 000004 003052 2$: BIT #BIT2,E.CONT ;ERROR 2?
9096 041536 001407 BEQ 3$ ;NO-BRANCH
9097 041540 105737 003137 TSTB DRNAFG ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
9098 041544 001402 BEQ 15$ ;BR IF NOT
9099 041546 105237 003140 INCB REISSU ;SET FLAG TO RE-ISSUE COMMAND
9100 041552 104066 15$: ERROR 66 ;UNSOLICITED ATTENTION
9101 041554 000447 BR 7$
9102 041556 032737 000010 003052 3$: BIT #BIT3,E.CONT ;ERROR 3?
9103 041564 001402 BEQ 4$ ;NO-BRANCH
9104 041566 104067 ERROR 67 ;UNEXPECTED DATA TYPE ERROR
9105 041570 000441 BR 7$
9106 041572 032737 000020 003052 4$: BIT #BIT4,E.CONT ;ERROR 4?
9107 041600 001402 BEQ 5$ ;NO-BRANCH
9108 041602 104070 ERROR 70 ;ATTENTION DID NOT RESET WITH CLEAR
9109 041604 000433 BR 7$
9110 041606 032737 000040 003052 5$: BIT #BIT5,E.CONT ;ERROR 5?
9111 041614 001402 BEQ 6$ ;NO-BRANCH
9112 041616 104071 ERROR 71 ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
9113 041620 000425 BR 7$
9114 041622 032737 000400 003052 6$: BIT #BIT6,E.CONT
9115 041630 001401 BEQ 8$
9116 041632 104072 ERROR 72 ;DATA LATE WHEN UNLOADING HEADER
9117 041634 032737 001000 003052 8$: BIT #BIT9,E.CONT
9118 041642 001401 BEQ 9$
9119 041644 104073 ERROR 73 ;CONTROLLER ERROR DURING DRIVER SERVICE
9120 041646 032737 002000 003052 9$: BIT #BIT10,E.CONT
9121 041654 001401 BEQ 10$
9122 041656 104074 ERROR 74 ;DRIVE DETECTED PARITY ERROR
9123 041660 032737 100000 003052 10$: BIT #BIT15,E.CONT
9124 041666 001401 BEQ 11$
9125 041670 104052 ERROR 52 ;MULTIPLE DRIVE SELECT
9126 041672 104075 11$: ERROR 75 ;UNDEFINED ERROR
9127 041674 005037 003052 7$: CLR E.CONT ;CLEAR CONTROLLER ERROR WORD
9128 041700 000137 044152 JMP BGNRTY ;GO DO RETRY
9129 *****
9130 ;SBTTL REPORT SUPPORT ROUTINE
9131 ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
9132 ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
9133 ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
9134 REPSUP:
9135 041704 104407 SAVREG
9136 041706 005037 005506 CLR ERRCOM ;CLEAR ERROR COMMAND STORE
9137 041712 116537 000001 005506 MOVB P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
9138 041720 012700 001162 MOV #SREG0,R0 ;FOR REPORTING
9139 041724 016520 000002 MOV P.CYLN(R5),(R0)+

```

```
9140 041730 116520 000005      MOVB P.TRCK(R5),(R0)+
9141 041734 105020      CLRB (R0)+
9142 041736 116520 000004      MOVB P.SECT(R5),(R0)+
9143 041742 105020      CLRB (R0)+
9144 041744 016520 000012      MOV P.WC(R5),(R0)+
9145 041750 012700 001174      MOV #SREG5,R0
9146 041754 116503 000007      MOVB P.BAHI(R5),R3
9147 041760 042703 177774      BIC #177774,R3
9148 041764 010337 001256      MOV R3,SREG36
9149 041770 016537 000010      MOV P.BALO(R5),SREG37
9150 041776 016520 000016      MOV P.CS1(R5),(R0)+
9151 042002 016520 000020      MOV P.CS2(R5),(R0)+
9152 042006 016520 000030      MOV P.DCYL(R5),(R0)+
9153 042012 016520 000026      MOV P.DTS(R5),(R0)+
9154 042016 016520 000022      MOV P.WCR(R5),(R0)+
9155                                ;HI BA BITS
9156 042022 016520 000024      MOV P.BAR(R5),(R0)+
9157 042026 016520 000032      MOV P.ASOF(R5),(R0)+
9158 042032 016520 000036      MOV P.DS(R5),(R0)+
9159 042036 016520 000034      MOV P.ER(R5),(R0)+
9160 042042 016520 000040      MOV P.AOO(R5),(R0)+
9161 042046 016520 000042      MOV P.BOO(R5),(R0)+
9162 042052 016520 000044      MOV P.AOI(R5),(R0)+
9163 042056 016520 000046      MOV P.BOI(R5),(R0)+
9164 042062 016520 000050      MOV P.AIO(R5),(R0)+
9165 042066 016520 000052      MOV P.BIO(R5),(R0)+
9166 042072 016520 000054      MOV P.AII(R5),(R0)+
9167 042076 016520 000056      MOV P.BII(R5),(R0)+
9168                                ;STORE PREVIOUS COMMAND FOR PRINTOUT
9169 042102 012701 001236      MOV #SREG26,R1
9170 042106 012700 005232      MOV #PRVCMD,R0
9171 042112 112021      MOVB (R0)+,(R1)+
9172 042114 105021      CLRB (R1)+
9173 042116 112021      MOVB (R0)+,(R1)+
9174 042120 105021      CLRB (R1)+
9175 042122 012021      MOV (R0)+,(R1)+
9176 042124 116021 000001      MOVB 1(R0),(R1)+
9177 042130 105021      CLRB (R1)+
9178 042132 111021      MOVB (R0),(R1)+
9179 042134 105021      CLRB (R1)+
9180 042136 016021 000006      MOV 6(R0),(R1)+
9181 042142 116003 000003      MOVB 3(R0),R3
9182 042146 042703 177774      BIC #177774,R3
9183 042152 010321      MOV R3,(R1)+
9184 042154 016011 000004      MOV 4(R0),(R1)
9185 042160 104410      RESREG
9186 042162 000207      RTS PC
9187                                ;*****
9188                                ;SBTTL REPORT ERROR ROUTINE
9189                                ;* ENTRY JSR PC,ERRHDL
9190                                ;*RETURN RTS PC
9191                                ;*
9192                                ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
9193                                ;*IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
9194                                ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
9195                                ;*****
```

9196					ERRHDL:	SAVREG	
9197	042164	104407			BISB	#377,DONE	:SET DONE FLAG
9198	042166	152737	000377	003123	INCB	ERRCNT	:INCREMENT ERROR COUNT
9199	042174	105237	003126		CLR	RECODE	:CLEAR RECOVERY CODE WORD
9200	042200	005037	005504		BIT	#LEVZER,RECODE	:TEST IF 2ND LEVEL ERROR
9201	042204	032737	000400	005504	BEQ	52\$	:NO - SKIP PARAM BLOCK CHANGE
9202	042212	001402			MOV	#PARM1,R5	:ELSE SET R5 TO PARAMETER BLOCK 1
9203	042214	012705	002714		MOV	#RETANL,A.ABNL	:SET NOW ABNORMAL AND NORMAL RETURN FOR
9204	042220	012737	044324	003046	MOV	#RETNML,A.NORM	:DRIVER OPERATIONS IN ERROR PROCESSING
9205	042226	012737	044342	003044	JSR	PC,REPSUP	:GO SET UP REGISTERS FOR REPORT
9206	042234	004737	041704				:NOW BEGIN TESTING THE ERROR
9207							:BITS. THE SEQUENCE IN
9208							:WHICH THEY ARE TESTED IS
9209							:CONSIDERED SIGNIFICANT IN
9210							:THAT ERRORS OF A MORE
9211							:CATASTROPHIC NATURE ARE FIRST
9212							:TESTED.
9213							:
9214							:IF AN ERROR IS FOUND SET,
9215							:THAT ERROR IS REPORTED AND
9216							:THE REPORTING IS TERMINATED,
9217							:IF ADDITIONAL ERRORS ARE SET,
9218							:THE RK611 REGISTER PRINTOUTS'
9219							:WILL SHOW THIS BUT THE
9220							:REGISTER CONTENTS MUST BE
9221							:MANUALLY DECODED TO LOCATE THE
9222							:SECOND ERROR
9223							:SET R4 TO ERROR REGISTER
9224	042240	016504	000034		MOV	P.FR(R5),R4	:TEST UPE. IF UES-SET
9225	042244	032765	020000	000020	BIT	#UPE,P.CS2(R5)	:REPORT ERROR
9226	042252	001406			BEQ	1\$	:
9227	042254	104001			ERROR	1	:
9228	042256	052737	000200	005504	BIS	#ABORT,RECODE	:
9229	042264	000137	043660		JMP	37\$	:
9230	042270	032765	004000	000020	1\$: BIT	#NEM,P.CS2(R5)	;TEST NON-EXISTANT MEMORY
9231	042276	001406			BEQ	2\$	:
9232	042300	104002			ERROR	2	:
9233	042302	052737	000200	005504	BIS	#ABORT,RECODE	:
9234	042310	000137	043660		JMP	37\$	:
9235	042314	032765	010000	000020	2\$: BIT	#NED,P.CS2(R5)	;TEST NON-EXISTANT DRIVE
9236	042322	001412			BEQ	3\$	:
9237	042324	032765	000400	000020	BIT	#UFE,P.CS2(R5)	;TEST IF NED & UFE BOTH SET
9238	042332	001403			BEQ	38\$	:
9239	042334	104035			ERROR	35	;NED & UFE BOTH SET ERROR
9240	042336	000137	043660		JMP	37\$	:
9241	042342	104003			ERROR	3	;NED ONLY
9242	042344	000137	043660		JMP	37\$	:
9243	042350	032765	000400	000020	3\$: BIT	#UFE,P.CS2(R5)	;TEST UNIT FIELD ERROR
9244	042356	001412			BEQ	4\$	:
9245	042360	032765	010000	000020	BIT	#NED,P.CS2(R5)	;TEST IF UFE & NED BOTH SET
9246	042366	001403			BEQ	39\$	:NO-SKIP
9247	042370	104035			ERROR	35	:REPORT NED & UFE BOTH SET
9248	042372	000137	043660		JMP	37\$	:
9249	042376	104004			ERROR	4	:REPORT UFE ONLY
9250	042400	000137	043660		JMP	37\$	:
9251	042404	032765	000100	000014	4\$: BIT	#CMDTO,P.PRST(R5)	:

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 174
REPORT ERROR ROUTINE

SEQ 0452
SEQ 0173

9252	042412	001423			BEQ	5\$	
9253	042414	004737	044352		JSR	PC, TOPROC	; GO PROCESS TIMEOUT
9254	042420	032737	002000	005504	BIT	#TWOTOS, RECODE	; 2ND TIMEOUT IN TIMEOUT PROC?
9255	042426	001007			BNE	40\$	; YES - SKIP TO ERROR 56
9256	042430	032737	000400	005504	BIT	#LEV2ER, RECODE	; TEST IF LEVEL 2 ERROR
9257	042436	001006			BNE	41\$	; YES - SKIP TO ERROR 57
9258	042440	104005			ERROR	5	; ELSE MAKE FULL TIMEOUT REPORT
9259	042442	000137	043660		JMP	37\$	
9260	042446	104056		40\$:	ERROR	56	; TWO TIMEOUTS ERROR REPORT
9261	042450	000137	043660		JMP	37\$	
9262	042454	104057		41\$:	ERROR	57	; 2ND LEVEL ERROR REPORT
9263	042456	000137	042204		JMP	ER2ENT	; GO BUILD AND MAKE 2ND REPORT
9264	042462	032765	010000	000014	BIT	#DRVSZD, P.PRST(R5)	; SEE IF DRIVE SIEZED BY OTHER PORT
9265	042470	001403			BEQ	65\$	; BR IF DRIVE IS AVAILABLE
9266	042472	104107			ERROR	107	; DRIVE SIEZED BY OTHER PORT
9267	042474	000137	043660		JMP	37\$	
9268	042500	032765	020000	000016	BIT	#SPAR, P.CS1(R5)	; TEST D TO C PARITY ERROR
9269	042506	001406			BEQ	6\$	
9270	042510	104006			ERROR	6	
9271	042512	052737	004000	005504	BIS	#RCLREQ, RECODE	
9272	042520	000137	043660		JMP	37\$	
9273	042524	032704	000010	6\$:	BIT	#DRPAR, R4	; TEST DRIVE DETECTED PARITY ERROR
9274	042530	001406			BEQ	7\$	
9275	042532	104007			ERROR	7	
9276	042534	052737	004000	005504	BIS	#RCLREQ, RECODE	
9277	042542	000137	043660		JMP	37\$	
9278	042546	032765	000010	000036	BIT	#ACLO, P.DS(R5)	; TEST AC LOW
9279	042554	001403			BEQ	8\$	
9280	042556	104010			ERROR	10	
9281	042560	000137	043660		JMP	37\$	
9282	042564	032765	000020	000036	BIT	#SPDLSS, P.DS(R5)	; TEST SPEED LOSS
9283	042572	001403			BEQ	24\$	
9284	042574	104011			ERROR	11	
9285	042576	000137	043660		JMP	37\$	
9286	042602	032765	004000	000016	BIT	#CTO, P.CS1(R5)	; TEST FOR CONTROLLER TIMEOUT
9287	042610	001401			BEQ	25\$	
9288	042612	104027			ERROR	27	
9289	042614	032704	000001	25\$:	BIT	#ILC, R4	; TEST ILLEGAL FUNCTION CODE
9290	042620	001403			BEQ	10\$	
9291	042622	104012			ERROR	12	
9292	042624	000137	043660		JMP	37\$	
9293	042630	032765	002000	000020	BIT	#PGE, P.CS2(R5)	; TEST PROGRAMMING ERROR
9294	042636	001403			BEQ	11\$	
9295	042640	104013			ERROR	13	
9296	042642	000137	043660		JMP	37\$	
9297	042646	032704	000004	11\$:	BIT	#ILF, R4	; TEST ILLEGAL DRIVE FUNCTION
9298	042652	001403			BEQ	12\$	
9299	042654	104014			ERROR	14	
9300	042656	000137	043660		JMP	37\$	
9301	042662	032704	000040	12\$:	BIT	#DTYE, R4	; TEST DRIVE TYPE ERROR
9302	042666	001403			BEQ	13\$	
9303	042670	104015			ERROR	15	
9304	042672	000137	043660		JMP	37\$	
9305	042676	032704	000020	13\$:	BIT	#FMTE, R4	; TEST FORMAT ERROR
9306	042702	001403			BEQ	14\$	
9307	042704	104016			ERROR	16	

F14

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 175
REPORT ERROR ROUTINESEQ 0453
SEQ 0174

9308	042706	000137	043660			JMP	37\$	
9309	042712	032704	004000	14\$:		BIT	#WLE,R4	;TEST WRITE LOCK ERROR
9310	042716	001403				BEQ	15\$	
9311	042720	104017				ERROR	17	
9312	042722	000137	043660			JMP	37\$	
9313	042726	032704	040000	15\$:		BIT	#UNS,R4	;TEST DRIVE UNSAFE
9314	042732	001406				BEQ	16\$	
9315	042734	104020				ERROR	20	
9316	042736	052737	000200	005504		BIS	#ABORT,RECODE	
9317	042744	000137	043760			JMP	ALLTRM	
9318	042750	032704	000002	16\$:		BIT	#SKI,R4	;TEST SEEK INCOMPLETE
9319	042754	001406				BEQ	17\$	
9320	042756	104021				ERROR	21	
9321	042760	052737	004000	005504		BIS	#RCLREQ,RECODE	
9322	042766	000137	043660			JMP	37\$	
9323	042772	032704	001000	17\$:		BIT	#COE,R4	;TEST CYLINDER OVERFLOW
9324	042776	001406				BEQ	18\$	
9325	043000	104022				ERROR	22	
9326	043002	052737	004000	005504		BIS	#RCLREQ,RECODE	
9327	043010	000137	043660			JMP	37\$	
9328	043014	032704	002000	18\$:		BIT	#IDAE,R4	;TEST ILLEGAL CYLINDER
9329	043020	001406				BEQ	19\$	
9330	043022	104023				ERROR	23	
9331	043024	052737	004000	005504		BIS	#RCLREQ,RECODE	
9332	043032	000137	043660			JMP	37\$	
9333	043036	032765	000040	000036	19\$:	BIT	#DROT,P.DS(R5)	;TEST DRIVE OFF TRACK
9334	043044	001406				BEQ	20\$	
9335	043046	104024				ERROR	24	
9336	043050	052737	004000	005504		BIS	#RCLREQ,RECODE	
9337	043056	000137	043660			JMP	37\$	
9338	043062	032704	010000	20\$:		BIT	#DTE,R4	;TEST DRIVE TIMING ERROR
9339	043066	001406				BEQ	21\$	
9340	043070	104025				ERROR	25	
9341	043072	052737	000200	005504		BIS	#ABORT,RECODE	
9342	043100	000137	043660			JMP	37\$	
9343	043104	032765	100000	000020	21\$:	BIT	#DLT,P.CS2(R5)	;TEST DATA LATE
9344	043112	001403				BEQ	22\$	
9345	043114	104026				ERROR	26	
9346	043116	000137	043660			JMP	37\$	
9347	043122	032704	020000	22\$:		BIT	#OPI,R4	;TEST IF OPI ERROR
9348	043126	001457				BEQ	29\$	
9349	043130	052737	000010	005504		BIS	#OPIERR,RECODE	
9350	043136	105737	003121			TSTB	DERCNT	;TEST IF FIRST ERROR
9351	043142	001402				BEQ	50\$	
9352	043144	000137	043660			JMP	37\$	;NO - SKIP REPORT
9353	043150	032737	000400	005504	50\$:	BIT	#LEV2ER,RECODE	;TEST IF A SECOND LEVEL 2 ERROR
9354	043156	001403				BEQ	26\$	;HAS ALREADY OCCURRED
9355	043160	104054				ERROR	54	
9356	043162	000137	043660			JMP	37\$	;GET OUT OF ERROR REPORT
9357	043166	004737	045012	26\$:		JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
9358	043172	004737	044522			JSR	PC,RDHDO	;GET HEADER OF SECTOR 0
9359	043176	032737	000400	005504		BIT	#LEV2ER,RECODE	;TEST IF ERROR GETTING HDR
9360	043204	001025				BNE	28\$	;IF YES-GO MAKE ABBREVIATED REPORT
9361	043206	013702	003036			MOV	RKBAS,R2	;STORE HEADER 0 INTO REGISTERS
9362	043212	042762	000100	000000		BIC	#IE,RKCS1(R2)	;RESET INTERRUPT ENABLE
9363	043220	016237	000024	001202		MOV	RKDB(R2),SREG10	;FOR REPORTING

G14

CZR6M00 RK611/06 SS VERIF 1
CZR6M0.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 176
REPORT ERROR ROUTINESEQ 0454
SEQ 0175

9364	043226	016237	000024	001204	MOV	RKDB(R2), \$REG11	
9365	043234	016237	000024	001206	MOV	RKDB(R2), \$REG12	
9366	043242	032762	100000	000000	BIT	#CERR, RKCS1(R2)	; TEST IF ERROR DURING STORAGE
9367	043250	001343			BNE	27\$	
9368	043252	104030			ERROR	30	; MAKE OPI REPORT
9369	043254	000137	043660		JMP	37\$	
9370	043260	104054			28\$: ERROR	54	
9371	043262	000137	042204		JMP	ER2ENT	; GO MAKE 2ND LEVEL REPORT
9372	043266	032704	000400		29\$: BIT	#HVRC, R4	; TEST IF HVRC ERROR
9373	043272	001457			BEQ	23\$	
9374	043274	052737	000004	005504	BIS	#HVRCER, RECODE	
9375	043302	105737	003121		TSTB	DERCNT	; TEST IF FIRST ERROR
9376	043306	001402			BEQ	30\$	; YES - REPORT
9377	043310	000137	043660		JMP	37\$	; JUMP TO RETURN
9378	043314	032737	000400	005504	30\$: BIT	#LEV2ER, RECODE	; TEST IF A 2ND LEVEL ERROR HAS ALREADY
9379	043322	001403			BEQ	31\$	; OCCURRED. NO-SKIP EXIT
9380	043324	104055			51\$: ERROR	55	
9381	043326	000137	043660		JMP	37\$	; GET OUT OF ERROR REPORT
9382	043332	004737	045012		31\$: JSR	PC, BLDEXH	; GO BUILD EXPECTED HEADER
9383	043336	004737	044522		JSR	PC, RDHDO	; GO GET HDR 0
9384	043342	032737	000400	005504	BIT	#LEV2ER, RECODE	; TEST IF ERROR IN GETTING HDR
9385	043350	001025			BNE	32\$	; IF YES-GO MAKE ABBREVIATED REPORT
9386	043352	013702	003036		MOV	RKBAS, R2	; GET RK611 BASE ADDRESS
9387	043356	042762	000100	000000	BIC	#IE, RKCS1(R2)	; CLEAR INTERRUPT ENABLE
9388	043364	016237	000024	001202	MOV	RKDB(R2), \$REG10	; STORE OFF HEADER
9389	043372	016237	000024	001204	MOV	RKDB(R2), \$REG11	
9390	043400	016237	000024	001206	MOV	RKDB(R2), \$REG12	
9391	043406	032762	100000	000000	BIT	#CERR, RKCS1(R2)	; TEST IN ANY ERROR IN UNLOAD
9392	043414	001343			BNE	51\$	; IY YES - GO MAKE SHORT REPORT
9393	043416	104031			ERROR	31	; MAKE FULL REPORT
9394	043420	000137	043660		JMP	37\$	
9395	043424	104055			32\$: ERROR	55	; ABBREVIATED HVRC ERROR RPORT
9396	043426	000137	042204		JMP	ER2ENT	; GO REPORT 2ND LEVEL ERROR
9397	043432	032704	000200		23\$: BIT	#BSE, R4	; TEST FOR BAD SECTOR ERROR
9398	043436	001430			BEQ	33\$	; NO - SKIP
9399	043440	052737	000002	005504	BIS	#BSERR, RECODE	; SET ERROR FLAG
9400	043446	016500	000030		MOV	P.DCYL(R5), R0	; GET CYL NO.
9401	043452	116501	000027		MOVB	P.DTS+1(R5), R1	; GET TRACK NO.
9402	043456	042701	177774		BIC	#177774, R1	; CLEAR ALL BITS EXCEPT TRACK
9403	043462	016502	000026		MOV	P.DTS(R5), R2	; GET SECTOR IN ERROR
9404	043466	042702	177740		BIC	#177740, R2	; CLEAR ALL BITS EXCEPT SECTOR
9405	043472	004737	040212		JSR	PC, BDSRCK	; GO SEE IF THIS SECTOR LISTED
9406	043476	032737	001000	005504	BIT	#BADSEC, RECODE	; TEST RESULT
9407	043504	001065			BNE	37\$	; YES - EXIT, NO ERROR OR REPORT
9408	043506	104104			ERROR	104	; ELSE REPORT BAD BSE
9409	043510	042737	000002	005504	BIC	#BSERR, RECODE	; RESET BSE ERROR FLAG
9410	043516	000460			BR	37\$	; GO EXIT
9411	043520	032704	100000		33\$: BIT	#DCK, R4	; TEST IF DATA CHECK
9412	043524	001435			BEQ	36\$	
9413	043526	052737	000020	005504	BIS	#DCKERR, RECODE	; SET DATA CHECK ERROR IN RECOVERY CODE
9414	043534	032704	000100		BIT	#ECH, R4	; TEST IF ECC IS HARD. IF
9415	043540	001406			BEQ	34\$	; YES SET UNCORRECTABLE IN
9416	043542	052737	000040	005504	BIS	#ECCNC, RECODE	; RECOVERY FLAG AND A 0 IN
9417	043550	005037	001206		CLR	\$REG12	; REG12 TO INDICATE UNCORRECTABLE,
9418	043554	000403			BR	35\$	; A 1 IN REG 12 FOR CORRECTABLE
9419	043556	012737	000001	001206	34\$: MOV	#1, \$REG12	

```

9420 043564 105737 003121      35$: TSTB   DERCNT      ;TEST IF FIRST ERROR
9421 043570 001033              BNE     37$      ;NO SKIP REPORT
9422 043572 004737 044666      JSR     PC,GTPKAD    ;GO GET PACK ADDRESS OF ERROR
9423 043576 016537 000060      MOV     P.EPOS(R5),SREG10 ;STORE ECC POSITION &
9424 043604 016537 000062 001202 MOV     P.EPAT(R5),SREG11 ;PATTERN
9425 043612 104032              ERROR  32      ;REPORT DCK ERROR
9426 043614 000137 043660      JMP     37$
9427 043620 032765 040000 000020 36$: BIT     #WCE,P.CS2(R5) ;TEST WRITE CHECK ERROR
9428 043626 001414              BEQ     37$
9429 043630 042737 000200 005504 BIC     #ABORT,RECODE    ;CLEAR ABORT & SET WRITE
9430 043636 052737 000100 005504 BIS     #WCERR,RECODE    ;CHECK ERROR IN RECODE
9431 043644 105737 003121      TSTB   DERCNT      ;TEST IF FIRST ERROR
9432 043650 001003              BNE     37$      ;NO - SKIP
9433 043652 004737 044666      JSR     PC,GTPKAD    ;GO GET ADDRESS OF ERROR
9434 043656 104032              ERROR  33      ;REPORT WCE
9435
9436 043660 032765 000020 000014 37$: BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
9437 043666 001404              BEQ     43$
9438 043670 104036              ERROR  36
9439 043672 052737 000200 005504 BIS     #ABORT,RECODE
9440
9441 043700 032765 000040 000014 43$: BIT     #DRVDSK,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
9442 043706 001404              BEQ     44$
9443 043710 104037              ERROR  37
9444 043712 052737 000200 005504 BIS     #ABORT,RECODE
9445
9446 043720 032765 004000 000014 44$: BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
9447 043726 001404              BEQ     46$
9448 043730 104040              ERROR  40
9449 043732 052737 000200 005504 BIS     #ABORT,RECODE
9450
9451 043740 032765 000010 000014 46$: BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
9452 043746 001404              BEQ     ALLTRM
9453 043750 104042              ERROR  42
9454 043752 052737 000200 005504 BIS     #ABORT,RECODE
9455
9456      ;ALL ERRORS MUST EXIT THROUGH THIS POINT
9457
9458
9459 043760 012705 002630      ALLTRM: MOV     #PARMD,R5      ;RESTORE PARAMETER BLOCK SELECTION
9460 043764 012737 042164 003046      MOV     #ERRHDL,A.ABNL    ;RESTORE INTERRUPT VECTORS FOR RETRY
9461 043772 012737 040716 003044      MOV     #ERRFRE,A.NORM    ;DRIVER OPERATIONS, IF ANY
9462 044000 032737 000200 005504      BIT     #ABORT,RECODE    ;IF ABORT IS NOT SET AND
9463 044006 001043              BNE     48$      ;THE DRIVE IS READY (HAS NOT
9464                                ;CYCLED DOWN)
9465 044010 013702 003036      MOV     RKBAS,R2      ;GET BASE ADDRESS
9466 044014 032762 000200 000012      BIT     #RDY,RKDS(R2)    ;TEST IF DRIVE READY SET
9467 044022 001004              BNE     47$      ;RECALIBRATE REQUIRED BIT IS SET
9468 044024 052737 000200 005504      BIS     #ABORT,RECODE    ;ELSE ABORT WITH ABORT MESSAGE
9469 044032 000431              BR      48$
9470 044034 032737 004000 005504 47$: BIT     #RCLREQ,RECODE    ;IF RECALIBRATE IS REQUIRED
9471 044042 001443              BEQ     BGNRTY    ;FOR RETRY SET UP PARAM
9472 044044 112737 000113 000001      MOVB    #RECAL,P.CMND    ;BLOCK TO DO IT.
9473 044052 012737 044324 003046      MOV     #RETANL,A.ABNL
9474 044060 004737 040316      JSR     PC,DRVCAL
9475 044064 012737 042164 003046      MOV     #ERRHDL,A.ABNL ;RESTORE ERROR RETURN

```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 178
REPORT ERROR ROUTINE

SEQ 0456
SEQ 0177

```

9476 044072 032737 000400 005504      BIT      #LEV2ER,RECODE ; IF AN ERROR OCCURRED IN THE
9477 044100 001424                      BEQ      BGNRTY      ; RECAL ATTEMP SET ABORT
9478 044102 052737 000200 005504      BIS      #ABORT,PECODE ; PRINT THE RECAL ERROR MESSAGE, AND
9479 044110 104060                      ERROR    60          ; GO REPORT DETAILS
9480 044112 000137 042204                      JMP      ER2ENT
9481 044116 104061      48$:      ERROR    61          ; REPORT ABORT-RETRY FAILED
9482 044120 105037 003130                      CLRB     DRVERS    ; DROP DRIVE & GO TO NEW DRIVE
9483 044124 105037 003126                      CLRB     ERRCNT
9484 044130 000137 017026                      JMP      NEWDRV
9485
9486      ; THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
9487      ; IF THE DRIVE READY BIT IS RESET.
9488
9489      044134 000005      HLTPRG: RESET      ; DISABLE ALL DEVICES
9490      044136 000000                      HALT      ; HALT THE CPU
9491      044140 105037 003126                      CLRB     ERRCNT    ; CLEAR ERROR COUNT
9492      044144 000005                      RESET     ; RESET ALL DEVICES
9493      044146 000137 012606                      JMP      CMSTRT
9494
9495
9496      ; THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
9497      ; HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
9498      ; THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
9499      ; RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
9500      ; FLAG IS SET AND PROGRAM HALTS.
9501
9502 044152 032737 000136 005504 BGNRTY: BIT      #BSERR!HVR CER!OPIERR!DCKERR!WCERR,RECODE ; TEST IF ANY DATA ERROR
9503 044160 001404                      BEQ      3$
9504 044162 052737 100000 005504 9$:      BIS      #ANYDER,RECODE
9505 044170 000453                      BR        2$
9506
9507 044172 105737 003143      3$:      TSTB     NORTRY      ; SEE IF "NO-RETRY" FLAG SET
9508 044176 001371                      BNE      9$          ; BR IF YES
9509 044200 032737 001000 005504      BIT      #BADSEC,RECODE ; TEST IF BAD SECTOR FLAG SET
9510 044206 001044                      BNE      2$          ; IF YES-EXIT TO CALLER
9511 044210 123737 003126 003127      CMPB     ERRCNT,ERRLMT ; BEGIN RETRY IF ERROR COUNT HAS
9512 044216 001012                      BNE      1$          ; NOT BEEN EXCEEDED
9513 044220 005037 001174                      CLR      $REG5
9514 044224 113737 003126 001174      MOVB     ERRCNT,$REG5 ; GET ERROR RETRY COUNT
9515 044232 104102                      ERROR    102        ; REPORT RETRY UNSUCCESSFUL
9516 044234 052737 000200 005504      BIS      #ABORT,RECODE ; SET ABORT & QUIT
9517 044242 000734                      BR        HLTPRG
9518 044244 013702 003036      1$:      MOV      RKBAS,R2      ; GET RK BASE ADDRESS
9519 044250 112765 000177 000001      MOVB     #SUBCLR,P.CMND(R5) ; SET UP TO CLEAR SUBSYSTEM
9520 044256 004737 040316                      JSR      PC,DRVCAL ; GO DO IT
9521 044262 012700 005246                      MOV      #COMSTR,R0 ; GO AND REESTABLISH THE COMMAND
9522 044266 012025                      MOV      (R0)+,(R5)+ ; AS IT WAS ENTERED INTO THE
9523 044270 012025                      MOV      (R0)+,(R5)+ ; PARAMETER BLOCK
9524 044272 012025                      MOV      (R0)+,(R5)+
9525 044274 012025                      MOV      (R0)+,(R5)+
9526 044276 012025                      MOV      (R0)+,(R5)+
9527 044300 012025                      MOV      (R0)+,(R5)+
9528 044302 012705 002630                      MOV      #PARMO,R5
9529 044306 012737 000000 177776      MOV      #PRO,2#PSW ; LOWER PRIORITY TO ALLOW INTERRUPT
9530 044314 004737 040316                      JSR      PC,DRVCAL ; CALL DRIVER
9531 044320 104410      2$:      RESREG ; IF RETURN GETS HERE, NO ERROR

```

J14

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 179
REPORT ERROR ROUTINESEQ 0457
SEQ 0178

```

9532 044322 000207          RTS      PC          ;OCCURRED, RECOVERY WAS SUCCESSFUL
9533
9534
9535 044324 152737 000377 003123 RETANL: BISB    #377,DONE      ;SET DONE
9536 044332 052737 000400 005504          BIS    #LEV2ER,RECODE ;SET LEVEL TWO ERROR
9537 044340 000207          RTS      PC
9538 044342 152737 000377 003123 RETNML: BISB    #377,DONE      ;SET DONE
9539 044350 000207          RTS      PC
9540
9541
9542
9543
9544
9545
9546
9547 044352
9548 044352 104407
9549 044354 013702 003036          SAVREG
9550 044360 012701 001174          MOV      RKBAS,R2
9551 044364 016221 000000          MOV      #SREG5,R1      ;SET UP R1 FOR RK REGISTER STORAGE
9552 044370 016221 000010          MOV      RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
9553 044374 016221 000020          MOV      RKCS2(R2),(R1)+ ;REGISTERS
9554 044400 016221 000006          MOV      RKDC(R2),(R1)+
9555 044404 016221 000002          MOV      RKDA(R2),(R1)+
9556 044410 016221 000004          MOV      RKWC(R2),(R1)+
9557 044414 016221 000016          MOV      RK9A(R2),(R1)+
9558 044420 016221 000012          MOV      RKASOF(R2),(R1)+
9559 044424 016221 000014          MOV      RKDS(R2),(R1)+
9560 044430 005000          MOV      RKER(R2),(R1)+
9561
9562
9563 044432 012705 002714          CLR      R0
9564 044436 112765 000141 000001          MOV      R0
9565 044444 004737 040316          ;THIS CODE WILL ATTEMPT TO
9566 044450 032765 000100 000014          ;RETRIEVE THE STATUS FROM THE
9567 044456 001403          ;DRIVE.
9568 044460 052737 002000 005504          MOV      #PARM1,R5      ;SET UP TO USE PARM1
9569 044466 062705 000040          MOV      #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
9570 044472 012521          JSR      PC,DRVCL      ;CALL DRIVER
9571 044474 012521          BIT      #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
9572 044476 012521          BEQ      1$
9573 044500 012521          ;NO - SKIP
9574 044502 012521          ;SET TWO TIMEOUTS FLAG
9575 044504 012521          ADD      #P.ADD,R5      ;BUMP R5 TO POINT TO DRIVE STATUS
9576 044506 012521          MOV      (R5)+,(R1)+ ;MOVE ALL THE DRIVE STATUS INTO THE
9577 044510 012521          MOV      (R5)+,(R1)+ ;TEMP REGS FOR REPORTING.
9578
9579 044512 012705 002630          MOV      #PARM0,R5      ;RESTORE PARM 0
9580 044516 104410          RESREG
9581 044520 000207          RTS      PC
9582
9583
9584
9585
9586
9587

```

```
9588 044522          RDH00: SAVREG
9589 044522      104407      MOV      P.DTS(R5),R1      ;STORE TRACK AND SECTOR
9590 044524      016501      MOV      P.B10(R5),R0      ;GET THE CYLINDER ADRS
9591 044530      016500      BIC      #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
9592 044534      042700      ASR      R0      ;OFF UNUSED BITS AND POSITION
9593 044540      006200      ASR      R0      ;FOR USE AS THE DESIRED
9594 044542      006200      ASR      R0      ;CYLINDER IN THE READ
9595 044544      006200      ASR      R0      ;HEADER COMMAND.
9596 044546      006200      MOV      #PARM1,R5      ;SET UP TO USE P.B. 1
9597 044550      012705      MOV      R1,P.SECT(R5)      ;INSERT TRK,SECT FOR READ HDR
9598 044554      010165      MOV      R0,P.CYLN(R5)      ;SET CYL NO.
9599 044560      010065      MOV      #SUBCLR,P.CMND(R5) ;SET S.S. CLEAR CMND
9600 044564      112765      MOV      #PRO,2#PSW      ;ALLOW INTERRUPTS
9601 044572      012737      JSR      PC,DRVCL      ;CLEAR THE S.S.
9602 044600      004737      MOV      #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
9603 044604      112765      JSR      PC,DRVCL      ;DO READ HEADER
9604 044612      004737      MOV      HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT = 0
9605 044616      013712      BIT      #BIT7,(R2)      ;CHECK FOR C. READY IN CSI
9606 044622      032712      BEQ      4$      ;BR IF NOT RDY YET
9607 044626      001775      MOV      HOLD4,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
9608 044630      013712      BIT      #BIT7,(R2)      ;CHECK FOR C. READY IN CSI
9609 044634      032712      BEQ      6$      ;BR IF NOT RDY YET
9610 044640      001775      BICB      #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
9611 044642      142765      BISB      FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
9612 044650      153765      MOV      #PARMO,R5      ;RESTORE P.B. 0 ADDRESS
9613 044656      012705      RESREG      ;RESTORE R0-R5
9614 044662      104410      RTS      PC      ;RETURN
9615 044664      000207
9616
9617
9618
9619
9620
9621
9622 044666      016537      000030      001174      ;*****
9623 044674      005037      001176      ;SBTTL GET PACK ADDRESS ROUTINE
9624 044700      005037      001200      ;*****
9625 044704      116537      000026      001200      GTPKAD: MOV      P.DCYL(R5),SREG5 ;GET CYLINDER NUMBER
9626 044712      116537      000027      001176      CLR      SREG6      ;CLEAR REGISTERS FOR
9627 044720      005737      001200      CLR      SREG7      ;TRACK & SECTOR STORAGE
9628                                MOV      P.DTS(R5),SREG7 ;STORE THE TRACK AND SECTOR
9629                                MOV      P.DTS+1(R5),SREG6
9630                                TST      SREG7      ;ADJUST THE ADDRESS CONTAINED IN
9631                                ;THE RK REGISTERS FOR THE AUTOMATIC
9632                                ;INCREMENT
9633 044724      001403      BEQ      1$
9634 044726      005337      DEC      SREG7
9635 044732      000426      BR      5$
9636 044734      032765      010000      000016      1$: BIT      #CFMT,P.CS1(R5)
9637 044742      001404      BEQ      2$
9638 044744      012737      000023      001200      MOV      #19.,SREG7
9639 044752      000403      BR      3$
9640 044754      012737      000025      001200      2$: MOV      #21.,SREG7
9641 044762      005737      001176      3$: TST      SREG6
9642 044766      001403      BEQ      4$
9643 044770      005337      DEC      SREG6
9644 044774      000405      BR      5$
9645 044776      012737      000002      001176      4$: MOV      #2,SREG6
9646 045004      005337      001174      DEC      SREG5
9647 045010      000207      5$: RTS      PC
```

CZR6M00 RK611 06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 181
GET PACK ADDRESS ROUTINE

L14

SEQ 0459
SEQ 0180

```
9644
9645
9646
9647
9648
9649
9650
9651
9652
9653 045012 104407
9654 045014 016537 000030 001174
9655 045022 016501 000026
9656 045026 042701 174377
9657 045032 006201
9658 045034 006201
9659 045036 006201
9660
9661 045040 016537 000026 001176
9662 045046 042737 177740 001176
9663 045054 060137 001176
9664 045060 052737 140000 001176
9665 045066 032765 010000 000016
9666 045074 001403
9667 045076 052737 001000 001176
9668 045104 013737 001174 001200 23$:
9669 045112 013701 001176
9670 045116 043737 001176 001200
9671 045124 043701 001174
9672 045130 050137 001200
9673 045134 104410
9674 045136 000207
9675

*****
: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
;BIT.
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
RTS PC
```

CZR6MDO RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 182
RK611/RK06-RK07 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.10)

SEQ 0460
SEQ 0181

M14

.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 SIMULTEOUSLY. FOR SEQUENTIAL
* OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
*
*CALL JSR PC,W.WTCH
* RETURN IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
*
* OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
* BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
* IN THE PROGRAM DEVICE STATUS REGISTER OF THE
* APPROPRIATE PARAMETER BLOCK WILL BE SET.
*

9676
9677
9678
9679
9680
9681
9682
9683
9684
9685
9686
9687
9688
9689
9690
9691
9692
9693
9694
9695
9696
9697
9698
9699
9700
9701
9702
9703
9704
9705
9706
9707
9708
9709
9710
9711
9712
9713
9714
9715
9716
9717
9718 045140 010546
9719 045142 010446
9720 045144 010346
9721 045146 010246
9722 045150 013746 177776
9723 045154 005337 003056
9724 045160 001034
9725 045162 013737 003060 003056
9726 045170 105737 003100
9727 045174 001426
9728 045176 013737 003042 177776
9729 045204 013702 003036
9730 045210 005337 003114
9731 045214 001016

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

N14

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 183
*WATCH-DOG TIMER

SEQ 046!
SEQ 0182

9732	045216	105037	003100		CLRB	W.TIME	;RESET TIMING INDICATOR
9733	045222	013705	003112		MOV	PBLKT,R5	;LOAD ADDRESS OF PARAMETER BLOCK
9734							;TABLE FOR INDEXING
9735	045226	052765	000100	000014	BIS	*CMDTO.P.PRST(R5)	;SET COMMAND TIME OUT
9736	045234	020537	003054		CMP	R5,0.WAIT	;CHECK IF DRIVER IS WAITING FOR
9737							;COMMAND COMPLETION
9738	045240	001002			BNE	SS	;NO, DO NOT ALTER WAITING FOR
9739							;COMMAND COMPLETION
9740	045242	005037	003054		CLR	0.WAIT	;CLEAR WAIT FOR COMMAND COMPLETION
9741	045246	004737	050522	SS:	JSR	PC,R.ABNL	;BRANCH TO ERROR ROUTINE
9742	045252	012637	177776	20\$:	MOV	(SP)+,PS	;RESTORE PSW
9743	045256	012602			MOV	(SP)+,R2	;RESTORE R2
9744	045260	012603			MOV	(SP)+,R3	;RESTORE R3
9745	045262	012604			MOV	(SP)+,R4	;RESTORE R4
9746	045264	012605			MOV	(SP)+,R5	;RESTORE R5
9747	045266	000207			RTS	PC	;RETURN

B15

.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

9748
9749
9750
9751
9752
9753
9754
9755
9756
9757
9758
9759
9760
9761
9762
9763
9764
9765
9766
9767
9768
9769
9770
9771
9772
9773
9774
9775
9776
9777
9778
9779
9780
9781
9782
9783
9784
9785
9786
9787
9788
9789
9790
9791
9792 045270 010546
9793 045272 010446
9794 045274 010346
9795 045276 010246
9796 045300 010146
9797 045302 010046
9798 045304 013702 003036
9799 045310 016237 000010 003002
9800 045316 032737 001000 003002
9801 045324 001407
9802 045326 052737 100000 003052
9803 045334 004737 050546

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

```

9804 045340 000137 047520      JMP      I.ATRN      ;RETURN
9805
9806 045344 105737 003074      1$:    TSTB     I.ISRL      ;CHECK IF INTERRUPT OR RELEASE
9807 045350 001410              BEQ      6$          ;NO, CHECK IF DRIVE AVAILABLE
9808 045352 100403              BMI      5$          ;CHECK IF RELEASE COMMAND
9809 045354 105037 003074      CLRB     I.ISRL      ;YES, CLEAR FLAG
9810 045360 000473              BR       I.ICO       ;CONTINUE PROCESSING INTERRUPT
9811
9812 045362 105037 003074      5$:    CLRB     I.ISRL      ;CLEAR FLAG
9813 045366 000137 046502      JMP      I.ATTN     ;GO PROCESS DRIVE ATTENTIONS
9814
9815 045372 032737 010400 003002 6$:    BIT      #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
9816                                ;UNIT FIELD ERROR
9817 045400 001413              BEQ      7$          ;NO, WAIT FOR DUAL ACCESS INTERRUPT
9818 045402 013704 003002      MOV      T.CS2,R4    ;LOAD R4 FOR DRIVE NUMBER
9819 045406 042704 177770      BIC      #IC<DRVMSK>,R4 ;KEEP DRIVE BITS
9820 045412 013705 003112      MOV      PBLKT,R5    ;STORE PARAMETER BLOCK ADDRESS
9821 045416 016237 000000 003000      MOV      RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
9822 045424 000137 045712      JMP      I.ERRC     ;REPORT ERROR
9823
9824 045430 016237 000012 003020 7$:    MOV      RKDS(R2),T.DS    ;STORE STATUS REGISTER FOR COMPARISON
9825 045436 032737 000001 003020      BIT      #DRA,T.DS    ;CHECK IF DRIVE SEIZED BY OTHER
9826                                ;PORT
9827 045444 001041              BNE      I.IOO       ;NO, CONTINUE PROCESSING INTERRUPT
9828
9829                                ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
9830 045446 032737 164000 003002      BIT      #DLT!WCE!UPE!NEM,T.CS2
9831
9832 045454 001007              BNE      10$         ;INDICATE ERROR
9833 045456 016237 000014 003016      MOV      RKER(R2),T.ER    ;STORE ERROR REGISTER
9834
9835                                ;CHECK FOR DATA TRANSFER ERROR TYPE ERROR
9836 045464 032737 125700 003016      BIT      #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
9837
9838 045472 001407              BEQ      11$         ;NO, WAIT FOR RELEASE OF RK06 DRIVE
9839
9840 045474 052737 000010 003052 10$:   BIS      #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
9841 045502 004737 050546      JSR      PC,R.CONT    ;REPORT ERROR
9842 045506 000137 047520      JMP      I.ATRN     ;RESTORE REGISTERS
9843
9844 045512 105037 003100      11$:   CLRB     W.TIME      ;RESET TIMING ON THIS DRIVE
9845 045516 005037 003114      CLR      W.DRV      ;CLEAR TIMING COUNT FOR THIS DRIVE
9846 045522 013705 003112      MOV      PBLKT,R5    ;LOAD R5 WITH PARAMETER BLOCK
9847                                ;ADDRESS
9848 045526 052765 010000 000014      BIS      #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
9849                                ;PROGRAM DRIVE STATUS REGISTER
9850 045534 005037 003054      CLR      O.WAIT     ;CLEAR WAIT FOR COMMAND COMPLETION
9851 045540 004737 050522      JSR      PC,R.ABNL    ;INDICATE ABNORMAL TERMINATION
9852 045544 000137 047520      JMP      I.ATRN     ;GO RESTORE REGISTERS
9853
9854 045550 013705 003054      I.IOO:  MOV      O.WAIT,R5    ;LOAD PARAMETER BLOCK ADDRESS INTO R5
9855 045554 001002              BNE      2$          ;IS COMMAND WAITING PROCESSING
9856                                ;YES, DO PROCESSING
9857 045556 000137 046502      JMP      I.ATTN     ;NO, PROCESS ATTENTION
9858
9859 045562 013704 003002      2$:    MOV      T.CS2,R4    ;STORE RKCS2 FOR DRIVE NUMBER

```

```

9860 045566 042704 177770          BIC      #1C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
9861
9862
9863 045572 126504 000000          CMPB     P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
9864 045576 001401 000000          BEQ      3$ ;YES, CONTINUE
9865 045600 000000 000000          HALT     ;NO, DRIVER ERROR
9866 045602 122765 000164 000001 3$:  CMPB     #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
9867 045610 001002 000000          BNE      10$ ;NO, EXECUTE NORMAL DATA TRANSFER
9868 045612 000137 046150          JMP      I.HDAL ;GO EXECUTE SPECIAL HEADER SEQUENCE
9869
9870 045616 005037 003054          10$:  CLR      O.WAIT ;CLEAR WAIT FOR COMMAND COMPLETION
9871 045622 005037 003114          CLR      W.DRV ;CLEAR WATCH-DOG TIME
9872 045626 105037 003100          CLRB    W.TIME ;RESET TIMING ON THIS DRIVE
9873 045632 016237 000000 003000  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
9874 045640 032737 100000 003000  BIT      #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
9875 045646 001021 000000          BNE      I.ERRC ;YES, PROCESS ERROR
9876 045650 016237 000016 003014  MOV      RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9877 045656 133737 003101 003015  BITB     INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
9878 045664 001004 000000          BNE      15$ ;YES, REPORT ERROR
9879 045666 004737 050534          JSR      PC,R.NORM ;INDICATE NORMAL RETURN
9880 045672 000137 047520          JMP      I.ATRN ;RESTORE REGISTERS
9881
9882 045676 052765 000010 000014 15$:  BIS      #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
9883
9884 045704 004737 050170          I.ERRA: JSR      PC,I.CSTS ;STORE CONTROLLER STATUS
9885 045710 000405 000000          BR       I.ERR ;STORE PATTERN AND POSITION INFORMATION
9886
9887 045712 013765 003000 000016 I.ERRC: MOV      T.CS1,P.CS1(R5) ;GET ERROR RKCS1
9888 045720 004737 050212 000000 JSR      PC,I.CS1+1 ;GET REST OF CONTROLLER STATUS
9889 045724 016265 000032 000062 I.ERR:  MOV      RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
9890 045732 016265 000030 000060  MOV      RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
9891 045740 004037 047536          JSR      RO,I.CCLR ;CLEAR CONTROLLER
9892 045744 047520 000000          I.ATRN  ;ERROR RETURN
9893 045746 032765 010400 000020  BIT      #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
9894                                     UNIT FIELD ERROR
9895 045754 001046 000000          BNE      5$ ;YES, REPORT ERROR
9896 045756 004037 050274          JSR      RO,I.STAT ;GATHER DRIVE STATUS
9897 045762 047520 000000          I.ATRN  ;ERROR RETURN
9898 045764 112737 000005 003000  MOVB     #DR.CLR,T.CS1 ;LOAD COMMAND
9899 045772 004037 047620          JSR      RO,I.ISSU ;ISSUE DRIVE CLEAR
9900 045776 047520 000000          I.ATRN  ;ERROR RETURN
9901 046000 133737 003101 003015  BITB     INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
9902 046006 001407 000000          BEQ      2$ ;NO, INDICATE DRIVE ERROR
9903 046010 052737 000020 003052  BIS      #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
9904                                     WITH CLEAR
9905 046016 004737 050546          JSR      PC,R.CONT ;REPORT CONTROLLER ERROR
9906 046022 000137 047520          JMP      I.ATRN ;GO RESTORE REGISTERS
9907
9908 046026 032737 040000 003024 2$:  BIT      #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE CLEARED
9909 046034 001403 000000          BEQ      3$ ;YES, CHECK FAULT
9910 046036 052765 000040 000014  BIS      #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
9911 046044 032737 001000 003026 3$:  BIT      #S.PAR,T.MR3 ;CHECK IF DRIVE PARITY ERROR
9912 046052 001407 000000          BEQ      5$ ;NO, INDICATE ABNORMAL TERMINATION
9913 046054 052737 002000 003052  BIS      #E.DPAR,E.CONT ;SET DRIVE PARITY ERROR
9914 046062 004737 050546          JSR      PC,R.CONT ;INDICATE CONTROLLER ERROR
9915 046066 000137 047520          JMP      I.ATRN ;RETURN

```

```

9916
9917 046072 032765 000020 000014 5$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
9918 046100 001017 10$ BNE 10$ ;YES, GO REPORT ERROR
9919 046102 032737 020000 003024 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
9920 046110 001413 10$ BEQ 10$ ;NO, REPORT ERROR
9921 046112 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
9922 046120 113737 003101 003100 MOVB INTMSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
9923 046126 013737 003064 003114 MOV W.8SEC,W.DRV
9924 046134 000137 047520 JMP I.RTRN ;GO RESTORE REGISTERS
9925
9926 046140 004737 050522 10$: JSR PC,R.ABNL ;GO REPORT ERROR
9927 046144 000137 047520 JMP I.RTRN ;GO RESTORE REGISTERS
9928
9929 .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
9930
9931 046150 016237 000000 003000 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
9932 ERROR
9933 046156 032737 100000 003000 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
9934 046164 001422 5$ BEQ 5$ ;NO, CHECK FOR ATTENTION
9935
9936 046166 005037 003054 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
9937 046172 105037 003100 CLRB W.TIME ;RESET TIMING ON DRIVE
9938 046176 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT
9939 046202 013765 003000 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
9940 046210 004737 050212 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
9941 046214 004037 047536 JSR RD,I.CCLR ;CLEAR CONTROLLER
9942 046220 047520 I.RTRN ;ERROR RETURN
9943 046222 004737 050522 JSR PC,R.ABNL ;INDICATE ERROR RETURN
9944 046226 000137 047520 JMP I.RTRN ;RESTORE REGISTERS
9945
9946 046232 016237 000016 003014 5$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9947 046240 133737 003101 003015 BITB INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
9948 046246 001410 7$ BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
9949 046250 005037 003054 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
9950 046254 105037 003100 CLRB W.TIME ;RESET TIMING ON DRIVE
9951 046260 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT
9952 046264 000137 045704 JMP I.ERRA ;GO REPORT ERROR
9953
9954 046270 013701 003070 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
9955 046274 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
9956 046300 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
9957 046304 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
9958 046310 010137 003070 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
9959 046314 016237 000010 003002 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
9960 046322 032737 100000 003002 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
9961 046330 001055 35$ BNE 35$ ;YES, REPORT ERROR
9962 046332 005337 003072 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
9963 046336 001026 25$ BNE 25$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
9964 046340 005037 003054 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
9965 046344 005037 003114 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
9966 046350 105037 003100 CLRB W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
9967 046354 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
9968 046362 112737 000001 003000 MOVB #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
9969 046370 004037 047620 JSR RD,I.ISSU ;GET SECTOR COUNT
9970 046374 047520 I.RTRN ;ERROR RETURN
9971 046376 013765 003026 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

F15

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 188
*READ ALL HEADERS INTERRUPT SEQUENCE

SEQ 0466
SEQ 0187

```

9972 046404 004737 050534 JSR PC.R.NORM ;INDICATE NORMAL TERMINATION
9973 046410 000137 047520 JMP I.ATRN ;RESTORE REGISTERS
9974
9975 046414 016562 000002 000020 25$: MOV P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
9976 046422 016562 000004 000006 MOV P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
9977 046430 116565 000007 000017 MOV B P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
9978 046436 042765 165777 000016 BIC #C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
9979 ; DRIVE TYPE
9980 046444 112765 000125 000016 MOV B #RDHEAD,P.CS1(R5) ;STORE COMMAND ISSUED
9981 046452 016562 000016 000000 MOV P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
9982 046460 000137 047520 JMP I.ATRN ;RESTORE REGISTERS
9983
9984 046464 052737 000400 003052 35$: BIS #E.DLT,E.CONT ;SET DATA LATE WHILE UNLOADING HEADER
9985 046472 004737 050546 JSR PC.R.CONT ;REPORT ERROR
9986 046476 000137 047520 JMP I.ATRN ;RESTORE REGISTERS
9987
9988 .SBTTL *DRIVE ATTENTION SCANNER
9989
9990 046502 016237 000000 003000 I ATTN: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
9991 ; REGISTER 1 FOR COMPARISON
9992 046510 032737 100000 003000 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR OCCURRED
9993 046516 001441 BEQ SS ;NO, CHECK IF ATTENTION
9994
9995 ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
9996 046520 032737 164000 003002 BIT #DLT!WCE!UPE!NEM,T.CS2
9997
9998 046526 001007 BNE 1$ ;INDICATE ERROR
9999 046530 016237 000014 003016 MOV RKER(R2),T.ER ;STORE ERROR REGISTER
10000
10001 ;CHECK FOR DATA TRANSFER ERROR TYPE
10002 046536 032737 125700 003016 BIT #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10003
10004 046544 001407 BEQ 2$ ;NO DATA TRANSFER ERROR
10005
10006 046546 052737 000010 003052 1$: BIS #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
10007 046554 004737 050546 JSR PC.R.CONT ;REPORT ERROR
10008 046560 000137 047520 JMP I.ATRN ;RESTORE REGISTERS
10009
10010 046564 013704 003002 2$: MOV T.CS2,R4 ;SAVE CS2 FOR REGISTER NUMBER
10011 046570 042704 177770 BIC #C<DRVMSK>,R4 ;STRIP OFF JUNK
10012 046574 105037 003100 CLR B W.TIME ;CLEAR WATCH DOG TIMER
10013 046600 005037 003114 CLR W.DRV ;RESET TIMER VALUE
10014 046604 013705 003112 MOV PBLKT,R5 ;STORE PARAMETER BLOCK ADDRESS IN R5
10015
10016 ;
10017 ;
10018 046610 042765 000006 000014 BIC #DRVPOS!DRVPDT,P.PRST(R5)
10019
10020 046616 000137 045712 JMP I.ERRC ;GO REPORT ERROR
10021
10022 046622 032737 040000 003000 5$: BIT #DI,T.CS1 ;CHECK IF ANY DRIVE ATTENTION
10023 046630 001002 BNE 6$ ;YES, PROCESS INTERRUPT
10024 046632 000137 047520 JMP I.ATRN ;RESTORE REGISTERS
10025
10026 046636 016237 000016 003014 6$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10027 046644 105737 003015 TST B T.ASOF+1 ;CHECK IF ANY ATTENTIONS SET

```

10028	046650	001007			BNE	7\$	;YES GO PROCESS INTERRUPT
10029	046652	052737	000002	003052	BIS	#E.NOAT,E.CONT	;SET NO ATTENTION IN ATTENTION SUMMARY
10030	046660	004737	050546		JSR	PC.R.CONT	;GO REPORT ERROR
10031	046664	000137	047520		JMP	I.RTRN	;GO RESTORE REGISTERS
10032							
10033	046670	133737	003101	003015	7\$: BITB	INTMSK,T.ASOF+1	;CHECK IF DESIRED INTERRUPT
10034	046676	001007			BNE	8\$	;YES, GO PROCESS IT
10035	046700	052737	000004	003052	BIS	#E.UATT,E.CONT	;SET UNSOLICATED ATTENTION
10036	046706	004737	050546		JSR	PC.R.CONT	;GO REPORT ERROR
10037	046712	000137	047520		JMP	I.RTRN	;GO RESTORE REGISTERS
10038							
10039	046716	013705	003112		8\$: MOV	PBLKT,R5	;STORE PARAMETER BLOCK TABLE
10040	046722	116504	000000		MOV	P.DRVN(R5),R4	;STORE DRIVE NUMBER
10041	046726	032765	020000	000014	BIT	#E.UNLD,P.PRST(R5)	;CHECK IF DRIVE UNLOADING
10042	046734	001402			BEQ	11\$	;NO, CONTINUE
10043	046736	000137	047420		JMP	I.UNLD	;SERVICE DRIVE IN POSITION AFTER ERROR
10044							
10045	046742	042765	000002	000014	11\$: BIC	#DRVPOS,P.PRST(R5)	;RESET DRIVE POSITIONING
10046	046750	005062	000026		CLR	RKMR1(R2)	;CLEAR MAINTENANCE REGISTER 1
10047	046754	112737	000001	003000	MOV	#DR.SEL,T.CS1	;LOAD COMMAND
10048	046762	004037	047620		JSR	RD,I.ISSU	;SELECT DRIVE WITH ATTENTION HIGH
10049	046766	047520			I.RTRN		;ERROR RETURN
10050	046770	013765	003026	000042	MOV	T.MR3,P.B00(R5)	;STORE STATUS BYTE 00 MESS B
10051	046776	032765	000200	000042	BIT	#S.FLT,P.B00(R5)	;CHECK IF DRIVE FAULT
10052	047004	001401			BEQ	12\$	;NO, CHECK FOR DRIVE STATUS CHANGE
10053	047006	000461			BR	I.AERR	;PROCESS ERROR
10054							
10055	047010	013765	003024	000040	12\$: MOV	T.MR2,P.A00(R5)	;STORE MAINTENANCE REGISTER 2
10056	047016	032765	040000	000040	BIT	#S.DSC,P.A00(R5)	;CHECK FOR DRIVE STATUS CHANGE
10057	047024	001004			BNE	13\$	;YES, PROCESS DRIVE STATUS CHANGE
10058	047026	052765	004000	000014	BIS	#NODSC,P.PRST(R5)	;SET NO DRIVE STATUS CHANGE
10059	047034	000446			BR	I.AERR	;PROCESS ERROR
10060							
10061	047036	112737	000005	003000	13\$: MOV	#DR.CLR,T.CS1	;LOAD COMMAND
10062	047044	004037	047620		JSR	RD,I.ISSU	;CLEAR DRIVE STATUS CHANGE
10063	047050	047520			I.RTRN		;ERROR RETURN
10064	047052	013765	003014	000032	MOV	T.ASOF,P.ASOF(R5)	;STORE ATTENTION SUMMARY
10065	047060	133765	003101	000033	BITB	INTMSK,P.ASOF+1(R5)	;CHECK IF ATTENTION RESET
10066	047066	001407			BEQ	15\$	;YES, CONTINUE INTERRUPT PROCESSING
10067	047070	052737	000020	003052	BIS	#E.CLAT,E.CONT	;SET ATTENTION DID NOT RESET
10068							;WITH DRIVE CLEAR
10069	047076	004737	050546		JSR	PC.R.CONT	;FLAG ERROR
10070	047102	000137	047520		JMP	I.RTRN	;RESTORE REGISTERS
10071							
10072	047106	013765	003024	000040	15\$: MOV	T.MR2,P.A00(R5)	;STORE MAINTENANCE REGISTER 2
10073	047114	032765	040000	000040	BIT	#S.DSC,P.A00(R5)	;CHECK IF DRIVE STATUS CHANGE
10074							;RESET
10075	047122	001404			BEQ	16\$	;YES, CONTINUE INTERRUPT PROCESSING
10076	047124	052765	000040	000014	BIS	#DRVDSC,P.PRST(R5)	;SET DRIVE STATUS CHANGE DID NOT CLEAR
10077	047132	000407			BR	I.AERR	;GO PROCESS ERROR
10078							
10079	047134	105037	003100		16\$: CLRB	W.TIME	;RESET TIMING ON THIS DRIVE
10080	047140	005037	003114		CLR	W.DRV	;CLEAR DRIVE TIMING COUNT
10081	047144	004737	050534		JSR	PC.R.NORM	;REPORT SUCCESSFUL COMMAND COMPLETION
10082	047150	000563			BR	I.RTRN	;RESTORE REGISTERS
10083							

*ATTENTION ERROR HANDLER

.SBTTL *ATTENTION ERROR HANDLER

```

10084
10085
10086 047152 042765 000004 000014 I.AERR: BIC #DRV PDT,F.PRST(R5) :RESET POSITIONING IN PROGRESS BECAUSE
10087                                     OF DATA TRANSFER
10088 047160 105037 003100 CLR W.TIME :CLEAR TIMING FOR THIS DRIVE
10089 047164 005037 003114 CLR W.DRV :RESET WATCH-DOG TIME
10090 047170 042765 177741 000016 BIC #177741,P.CS1(R5) :KEEP COMMAND ISSUED
10091 047176 042737 000036 003000 BIC #36,T.CS1 :KEEP CURRENT CONTROLLER STATUS
10092 047204 053765 003000 000016 BIS T.CS1,P.CS1(R5) :MAKE GOOD MESSAGE
10093 047212 013765 003002 000020 MOV T.CS2,P.CS2(R5) :STORE CONTROLLER REGISTERS
10094 047220 013765 003004 000022 MOV T.WCR,P.WCR(R5)
10095 047226 013765 003006 000024 MOV T.BA,P.BAR(R5)
10096 047234 013765 003010 000026 MOV T.DA,P.DTS(R5)
10097 047242 013765 003012 000030 MOV T.DC,P.DCYL(R5)
10098 047250 013765 003014 000032 MOV T.ASOF,P.AS(R5)
10099 047256 013765 003016 000034 MOV T.ER,P.ER(R5)
10100 047264 013765 003020 000036 MOV T.DS,P.DS(R5)
10101 047272 004037 050274 JSR RO,I.STAT :GATHER DRIVE STATUS
10102 047276 047520 I.RTRN :ERROR RETURN
10103 047300 112737 000005 003000 MOVB #DR.CLR,T.CS1 :LOAD COMMAND
10104 047306 004037 047620 JSR RO,I.ISSU :CLEAR DRIVE ERRORS
10105 047312 047520 I.RTRN :ERROR RETURN
10106 047314 133737 003101 003015 BITB INTMSK,T.ASOF+1 :CHECK IF ATTENTION RESET
10107 047322 001407 BEQ 2$ :YES, FLAG DRIVE ERROR
10108 047324 052737 000020 003052 BIS #E.CLAT,E.CONT :SET ATTENTION DID NOT RESET
10109 047332 004737 050546 JSR PC,R.CONT :REPORT ERROR
10110 047336 000137 047520 JMP I.RTRN :RESTORE REGISTERS
10111
10112 047342 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) :CHECK IF AHAARD DRIVE ERROR
10113 047350 001017 BNE 10$ :YES, REPORT ERROR
10114 047352 032737 020000 003024 BIT #S.PIP,T.MR2 :CHECK IF DRIVE IS UNLOADING
10115 047360 001413 BEQ 10$ :NO, REPORT ERROR
10116 047362 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) :SET DRIVE UNLOADING DUE TO ERROR
10117 047370 113737 003101 003100 MOVB INTMSK,W.TIME :SET TIMING ON THIS DRIVE
10118 047376 013737 003064 003114 MOV W.BSEC,W.DRV :LOAD 8 SECONDS FOR CYCLE UP TIME
10119 047404 000137 047520 JMP I.RTRN :RESTORE REGISTERS
10120
10121 047410 004737 050522 10$: JSR PC,R.ABNL :REPORT ERROR
10122 047414 000137 047520 JMP I.RTRN :RESTORE REGISTERS
10123
10124 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
10125
10126 047420 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) :CLEAR DRIVE UNLOADING BECAUSE OF ERROR
10127 047426 112737 000005 003000 MOVB #DR.CLR,T.CS1 :LOAD IN DRIVE CLEAR
10128 047434 004037 047620 JSR RO,I.ISSU :GO ISSUE DRIVE CLEAR
10129 047440 047520 I.RTRN :ERROR RETURN
10130 047442 136437 003101 003015 BITB INTMSK(R4),T.ASOF+1 :CHECK IF ATTENTION CLEARED
10131 047450 001406 BEQ 15$ :YES, CONTINUE
10132 047452 012737 000020 003052 MOV #E.CLAT,E.CONT :SET ATTENTION DID NOT RESET
10133 047460 004737 050546 JSR PC,R.CONT :REPORT ERROR
10134 047464 000415 BR I.RTRN :RESTORE REGISTERS
10135
10136 047466 032737 040000 003024 15$: BIT #S.DSC,T.MR2 :CHECK IF DRIVE STAUS CHANGE RESET
10137 047474 001403 BEQ 20$ :YES, CONTINUE
10138 047476 052765 000040 000014 BIS #DRV DSC,P.PRST(R5) :SET DRIVE STAUS CHANGE DID NOT CLEAR
10139 047504 105037 003100 20$: CLRB W.TIME :RESET TIMING ON THIS DRIVE

```

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 191
*ERROR CAUSING DRIVE TO UNLOAD

SEQ 0469
SEQ 0190

10140	047510	005037	003114	CLR	W.DRV	;CLEAR TIME COUNT
10141	047514	004737	050522	JSR	PC,R.ABNL	;REPORT ERROR
10142						
10143	047520	012600		I.RTRN: MOV	(SP)+,R0	;RESTORE R0
10144	047522	012601		MOV	(SP)+,R1	;RESTORE R1
10145	047524	012602		MOV	(SP)+,R2	;RESTORE R2
10146	047526	012603		MOV	(SP)+,R3	;RESTORE R3
10147	047530	012604		MOV	(SP)+,R4	;RESTORE R4
10148	047532	012605		MOV	(SP)+,R5	;RESTORE R5
10149	047534	003002		RTI		;RETURN
10150						

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 192
*CONTROLLER CLEAR ROUTINE

SEQ 0470
SEQ 0191

.SBTTL *CONTROLLER CLEAR ROUTINE

THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT
CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
E.CCLR SET IN E.CONT.

REGISTER USE

R2 ADDRESS OF RK06 REGISTERS
R5 ADDRESS OF PARAMETER BLOCK

*CALL JSR R0,I.CCLR
<ADDRESS OF ERROR RETURN>
RETURN

10151								
10152								
10153								
10154								
10155								
10156								
10157								
10158								
10159								
10160								
10161								
10162								
10163								
10164								
10165								
10166								
10167								
10168								
10169								
10170								
10171								
10172	047536	012762	100000	000000	I.CCLR:	MOV	#CCLR,RKCS1(R2)	; CLEAR CONTROLLER
10173	047544	016237	000000	003000		MOV	RKCS1(R2),T.CS1	; STORE COMMAND AND STATUS REGISTER 1
10174	047552	032737	100000	003000		BIT	#CERR,T.CS1	; CHECK IF CONTROLLER CLEAR DID
10175								; CLEAR ERROR
10176	047560	001407				BEQ	SS	; YES, RETURN TO DRIVER PROCESSING
10177	047562	052737	000001	003052		BIS	#E.CCLR,E.CONT	; SET CLEAR CONTROLLER DID NOT CLEAR ERROR
10178	047570	004737	050546			JSR	PC,R.CONT	; REPORT CONTROLLER ERROR
10179	047574	011000				MOV	(R0),R0	; SET UP ERROR RETURN
10180	047576	000200				RTS	R0	; RETURN
10181								
10182	047600	012762	000100	000000	SS:	MOV	#IE,RKCS1(R2)	; SET INTERRUPT ENABLE
10183	047606	112737	177777	003074		MOVB	#-1,I.ISRL	; SET INTERRUPT ENABLE ISSUED
10184	047614	005720				TST	(R0)+	; ADJUST FOR NORMAL RETURN
10185	047616	000200				RTS	R0	; RETURN

K15

CZR6MDO RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 193
*COMMAND ISSUED BY DRIVER SERVICE ROUTINESEQ 0471
SEQ 0192

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

```

```

;CHECK IF ANY CONTROLLER ERROR IS SET
2$:      BIT      #CTO!SPAR, T.CS1
        BNE      7$
        BIT      #UFE!PGE!NEM!NED!UPE!WCE!DLT, T.CS2
        BNE      7$
        BIT      #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK, T.ER
        BNE      7$

```

```

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

```

```

10186
10187
10188
10189
10190
10191
10192
10193
10194
10195
10196
10197
10198
10199
10200
10201
10202
10203
10204
10205
10206
10207
10208
10209
10210
10211
10212
10213
10214 047620 013746 003000
10215 047624 005037 003002
10216 047630 116537 000000 003002
10217 047636 013762 003002 000010
10218 047644 116537 000007 003001
10219 047652 142737 177753 003001
10220
10221 047660 013762 003000 000000
10222 047666 105762 000000
10223 047672 100375
10224 047674 004737 050042
10225 047700 032737 100000 003000
10226 047706 001437
10227 047710 032737 001000 003002
10228 047716 001406
10229 047720 052737 100000 003052
10230 047726 004737 050546
10231 047732 000440
10232
10233
10234 047734 032737 024000 003000 2$:
10235 047742 001027
10236 047744 032737 176400 003002
10237 047752 001023
10238 047754 032737 131761 003016
10239 047762 001017
10240
10241 047764 122716 000005

```

L15

CZR6M00 RK611 06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 194
*COMMAND ISSUED BY DRIVER SERVICE ROUTINE

SEQ 0472
SEQ 0193

10242	047770	001003			BNE	3\$	;NO, DO NOT SET DRIVE HARD ERROR
10243	047772	052765	000020	000014	BIS	#DRVHRD,P.PRST(R5)	;SET HARD DRIVE ERROR
10244	050000	004037	047536		JSR	RO,I.CCLR	;GO ISSUE A CONTROLLER CLEAR
10245	050004	050034			10\$		;ERROR RETURN
10246	050006	012762	000100	000000	5\$: MOV	#IE,RKCS1(R2)	;SET INTERRUPT ENABLE
10247	050014	005726			TST	(SP)+	;ADJUST STACK
10248	050016	005720			TST	(RO)+	;ADJUST RO FOR NORMAL RETURN
10249	050020	000200			RTS	RO	;RETURN
10250							
10251	050022	052737	001000	003052	7\$: BIS	#E.CERR,E.CONT	;SET CONTROLLER ERROR DURING
10252							; DRIVER SERVICING
10253	050030	004737	050546		JSR	PC,R.CONT	;REPORT ERROR
10254	050034	005726			10\$: TST	(SP)+	;ADJUST STACK
10255	050036	011000			MOV	(RO),RO	;ADJUST RO FOR ERROR RETURN
10256	050040	000200			RTS	RO	;RETURN

CZR6MDO RK611 06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 195
*STORE RK611 UNIBUS REGISTERS

SEQ 0473
SEQ 0194

.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
;
;   REGISTER          USE
;   -----          ---
;
;   R2                ADDRESS OF RK611 REGISTERS
;*****

```

10257
10258
10259
10260
10261
10262
10263
10264
10265
10266
10267
10268
10269
10270
10271
10272
10273
10274
10275
10276
10277
10278
10279
10280
10281
10282
10283
10284
10285
10286
10287
10288

050042	016237	000000	003000
050050	016237	000010	003002
050056	016237	000002	003004
050064	016237	000004	003006
050072	016237	000006	003010
050100	016237	000012	003020
050106	016237	000014	003016
050114	016237	000016	003014
050122	016237	000020	003012
050130	016237	000026	003022
050136	016237	000034	003024
050144	016237	000036	003026
050152	016237	000030	003030
050160	016237	000032	003032
050166	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

```

10318	050170	042765	177741	000016	I.CSTS: BIC	#177741,P.CS1(R5)	;CLEAR ALL BITS EXCEPT FUNCTION
10320							;OF LAST COMMAND ISSUED
10321	050176	042737	000036	003000	BIC	#36,T.CS1	;CLEAR FUNCTION OF CS1 STATUS
10322	050204	053765	003000	000016	BIS	T.CS1,P.CS1(R5)	;GENERATE CS1 STATUS INFORMATION
10323	050212	016265	000010	000020	I.CST1: MOV	RKCS2(R2),P.CS2(R5)	;STORE COMMAND AND STATUS REGISTER 2
10324	050220	016265	000002	000022	MOV	RKWC(R2),P.WCR(R5)	;STORE WORD COUNT REGISTER
10325	050226	016265	000004	000024	MOV	RKBA(R2),P.BAR(R5)	;STORE BUS ADDRESS REGISTER
10326	050234	016265	000006	000026	MOV	RKDA(R2),P.DTS(R5)	;STORE DESIRED TRACK AND SECTOR
10327	050242	016265	000012	000036	MOV	RKDS(R2),P.DS(R5)	;STORE DRIVE STATUS REGISTER
10328	050250	016265	000014	000034	MOV	RKER(R2),P.ER(R5)	;STORE ERROR REGISTER
10329	050256	016265	000016	000032	MOV	RKASOF(R2),P.ASOF(R5)	;STORE ATTENTION SUMMARY AND
10330							;OFFSET
10331	050264	016265	000020	000030	MOV	RKDCYL(R2),P.DCYL(R5)	;STORE CYLINDER ADDRESS
10332	050272	000207			RTS	PC	;RETURN
10333							

Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380</
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----------

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 198
*GATHER DRIVE STATUS

SEQ 0476
SEQ 0197

10390	050472	052737	002000	003052	BIS	#E.DPAR,E.CONT	;INDICATE BAD PARITY DETECTED BY DRIVE
10391	050500	004737	050546		JSR	PC,R.CONT	;REPORT ERROR
10392	050504	011000		3\$:	MOV	(RO),RO	;LOAD RO FOR ERROR RETURN
10393	050506	000200			RTS	RO	;RETURN
10394							
10395	050510	052765	001000	000014	5\$:	BIS	#PBSVAL,P.PRST(R5)
10396	050516	005720			TST	(RO),+	;SET PARAMETER BLOCK STATUS VALID
10397	050520	000200			RTS	RO	;ADJUST RO FOR NORMAL RETURN
10398							;RETURN

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 199
*COMMON DRIVER RETURNS

D16

SEQ 0477
SEQ 0198

.SBTTL *COMMON DRIVER RETURNS

R.ABNL:	CLRB	INTMSK	;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	JSR	PC,QA.ABNL	;INDICATE ABNORMAL RETURN
	RTS	PC	;RETURN
R.NORM:	CLRB	INTMSK	;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	JSR	PC,QA.NORM	;INDICATE NORMAL RETURN
	RTS	PC	;RETURN
R.CONT:	CLRB	INTMSK	;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	CLRB	W.TIME	;RESET WATCH DOG TIMING ON THIS DRIVE
	CLR	W.DRV	;CLEAR TIMING COUNT FOR THIS DRIVE
	JSR	PC,QA.CONT	;INDICATE CONTROLLER ERROR RETURN
	RTS	PC	;RETURN

10399			
10400			
10401	050522	105037	003101
10402	050526	004777	132314
10403	050532	000207	
10404			
10405	050534	105037	003101
10406	050540	004777	132300
10407	050544	000207	
10408			
10409	050546	105037	003101
10410	050552	105037	003100
10411	050556	005037	003114
10412	050562	004777	132262
10413	050566	000207	

.SBTTL *COMMAND INITATOR

```

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

```

```

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

```

```

10414
10415
10416
10417
10418
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 050570 010546
10446 050572 010446
10447 050574 010346
10448 050576 010246
10449 050600 010146
10450 050602 010046
10451 050604 013746 177776
10452 050610 013737 003042 177776
10453 050616 017605 000016
10454 050622 062766 000002 000016
10455 050630 016504 000000
10456 050634 042704 177770
10457 050640 010537 003112
10458 050644 116437 003102 003101
10459 050652 116437 003102 003100
10460 050660 013737 003062 003114
10461
10462 050666 013702 003036
10463
10464
10465
10466
10467
10468
10469

```

10470	050672	042765	075176	000014		BIC	*↑C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)	
10471								
10472	050700	010500				MOV	R5,R0	;STORE PARAMETER BLOCK ADDRESS
10473	050702	062700	000016			ADD	*P.CS1,R0	;CALCULATE FIRST LOCATION TO BE CLEARED
10474	050706	010501				MOV	R5,R1	;STORE PARAMETER BLOCK ADDRESS
10475	050710	062701	000062			ADD	*P.EPAT,R1	;CALCULATE LAST LOCATION TO BE CLEARED
10476								
10477	050714	005020			1\$:	CLR	(R0)+	;CLEAR RETURN PARAMETER
10478	050716	020001				CMP	R0,R1	;CHECK IF FINISHED
10479	050720	101775				BLOS	1\$	;NO, CLEAR NEXT RETURN PARAMETER
10480	050722	105037	003074			CLRB	I.ISRL	;CLEAR RELEASE OR INTERRUPT ISSUED
10481	050726	010465	000020			MOV	R4,P.CS2(R5)	;STORE DRIVE NUMBER
10482	050732	005062	000026			CLR	RKMRI(R2)	;CLEAR RK06 MAINTENANCE REGISTER 1
10483	050736	132765	000040	000001		BITB	*BITS,P.CMND(R5)	;CHECK IF SPECIAL COMMAND
10484	050744	001402				BEQ	3\$	;NO, PROCESS
10485	050746	000137	051462			JMP	C.SPEC	;JUMP TO SPECIAL COMMAND PROCESSOR
10486								
10487	050752	122765	000107	000001	3\$:	CMPB	*UNLOAD,P.CMND(R5)	;CHECK IF POSITIONING COMMAND
10488								START SPINDLE
10489								RECALIBRATE
10490								OFFSET
10491								SEEK
10492								UNLOAD
10493								
10494	050760	101174				BHI	25\$	;NO, DRIVE COMMAND
10495								SELECT DRIVE
10496								PACK ACKNOWLEDGE
10497								CLEAR
10498								
10499	050762	122765	000117	000001		CMPB	*SEEK,P.CMND(R5)	;CHECK IF DATA TRANSFER
10500	050770	103540				BLO	20\$	;YES, DATA TRANSFER COMMAND
10501								READ DATA
10502								WRITE DATA
10503								READ HEADER
10504								WRITE HEADER
10505								WRITE CHECK
10506	050772	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2)	;LOAD DRIVE NUMBER
10507	051000	052765	000002	000014		BIS	*DRVPOS,P.PRST(R5)	;SET DRIVE POSITIONING
10508	051006	005037	003054			CLR	O.WAIT	;CLEAR WAIT FOR COMMAND
10509	051012	122765	000117	000001		CMPB	*SEEK,P.CMND(R5)	;CHECK IF SEEK
10510	051020	001007				BNE	5\$	;NO, CHECK FOR OFFSET
10511	051022	016562	000002	000020		MOV	P.CYLN(R5),RKDCY(R2)	;LOAD CYLINDER ADDRESS
10512	051030	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	;LOAD SECTOR AND TRACK
10513	051036	000431				BR	8\$	;GO ISSUE COMMAND
10514								
10515	051040	122765	000115	000001	5\$:	CMPB	*OFFSET,P.CMND(R5)	;CHECK IF OFFSET
10516	051046	001007				BNE	6\$	;NO, CHECK FOR UNLOAD
10517	051050	116565	000006	000032		MOVB	P.OFST(R5),P.ASOF(R5)	;STORE OFFSET
10518	051056	016562	000032	000016		MOV	P.ASOF(R5),RKASOF(R2)	;LOAD OFFSET REGISTER
10519	051064	000416				BR	8\$	;GO ISSUE COMMAND
10520								
10521	051066	122765	000111	000001	6\$:	CMPB	*SRTSPL,P.CMND(R5)	;CHECK IF START SPINDLE
10522	051074	001003				BNE	7\$	;NO, CHECK IF RECAL
10523	051076	013737	003066	003114		MOV	W.MIN,W.DRV	;LOAD WATCH DOG TIME FOR 1 MINUTE
10524	051104	122765	000113	000001	7\$:	CMPB	*RECAL,P.CMND(R5)	;CHECK IF RECAL
10525	051112	001003				BNE	8\$	;NO, CONTINUE

G16

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 202
*COMMAND INITIATORSEQ 0480
SEQ 0201

10526	051114	013737	003064	003114		MOV	W.8SEC,W.DRV	:LOAD RECAL TIME FOR 8 SECONDS
10527	051122	116565	000007	000017	8\$:	MOV	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10528	051130	042765	165777	000016		BIC	#1C<CFMT!CDT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT
10529								:AND DRIVE TYPE
10530	051136	116565	000001	000016		MOV	P.CMND(R5),P.CS1(R5)	:MOVE COMMAND INTO CS1
10531	051144	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:RESET WRITE FOR WRITE CHECK
10532	051152	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IN NO CHECK MODE
10533	051160	001533				BEQ	30\$	:NO, SKIP CLEAR OF INTERRUPT ENABLE
10534	051162	042765	000100	000016		BIC	#IE,P.CS1(R5)	:CLEAR INTERRUPT ENABLE
10535	051170	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
10536	051176	004737	045140		10\$:	JSR	PC.W.WTCH	:CALL WATCH DOG TIMER
10537	051202	016237	000000	003000		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REGISTER 1
10538	051210	032737	000200	003000		BIT	#RDY,T.CS1	:WAIT FOR READY
10539	051216	001767				BEQ	10\$	
10540	051220	032737	100000	003000		BIT	#CERR,T.CS1	:CHECK FOR ERROR
10541	051226	001011				BNE	15\$	:YES, GIVE NORMAL RETURN
10542	051230	004737	045140		11\$:	JSR	PC.W.WTCH	:CALL WATCH DOG TIMER
10543	051234	016237	000016	003014		MOV	RKASOF(R2),T.ASOF	:STORE ATTENTION SUMMARY
10544	051242	133737	003101	003015		BIT	INTMSK,T.ASOF+1	:CHECK IF INTERRUPT HAS OCCURRED
10545	051250	001767				BEQ	11\$	:WAIT FOR DRIVE INTERRUPT
10546	051252	105037	003100		15\$:	CLRB	W.TIME	:RESET TIMING ON THIS DRIVE
10547	051256	005037	003114			CLR	W.DRV	:CLEAR DRIVE TIMING COUNT
10548	051262	004737	050534			JSR	PC.R.NORM	:INDICATE COMMAND IS FINISHED
10549	051266	000137	052442			JMP	C.ATRN	:RESTORE REGISTERS
10550								
10551	051272	016562	000010	000004	20\$:	MOV	P.BALO(R5),RKBA(R2)	:LOAD BUS ADDRESS REGISTER
10552	051300	016562	000012	000002		MOV	P.WC(R5),RKWC(R2)	:LOAD WORD COUNT REGISTER
10553	051306	016562	000002	000020		MOV	P.CYLN(R5),RKDCYL(R2)	:LOAD CYLINDER ADDRESS REGISTER
10554	051314	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	:LOAD SECTOR AND TRACK NUMBER
10555	051322	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	:CHECK IF WRITE CHECK COMMAND
10556	051330	001010				BNE	25\$	:NO, GO ISSUE THE COMMAND
10557	051332	032765	000200	000014		BIT	#W.WCK,P.PRST(R5)	:CHECK IF WRITE COMMAND SHOULD BE ISSUED
10558	051340	001404				BEQ	25\$	:NO, GO ISSUE THE COMMAND
10559	051342	012765	000123	000016		MOV	#WRDATA,P.CS1(R5)	:ISSUE WRITE COMMAND
10560	051350	000406				BR	26\$	:GO ISSUE COMMAND
10561								
10562	051352	116565	000001	000016	25\$:	MOV	P.CMND(R5),P.CS1(R5)	:MOVE COMMAND INTO CS1
10563	051360	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	:RESET WRITE FOR WRITE CHECK
10564	051366	116565	000007	000017	26\$:	MOV	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
10565	051374	142765	177750	000017		BICB	#1C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5)	:CLEAR ALL BITS EXCEPT
10566								:FORMAT, DRIVE TYPE, AND BUS ADDRESS
10567								:BITS 16-17
10568	051402	010537	003054			MOV	R5,0.WAIT	:LOAD WAITING FOR COMMAND
10569	051406	032765	100000	000014		BIT	#DTBA11,P.PRST(R5)	:CHECK IF INHIBIT BUS ADDRESS INCREMENT
10570	051414	001403				BEQ	27\$	:NO, LOAD CS2
10571	051416	052765	000020	000020		BIS	#BA1,P.CS2(R5)	:SET INHIBIT BUS ADDRESS INCREMENT
10572	051424	016562	000020	000010	27\$:	MOV	P.CS2(R5),RKCS2(R2)	:LOAD CS2
10573	051432	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IN NO CHECK MODE
10574	051440	001403				BEQ	30\$	:NO, SKIP CLEAR OF INTERRUPT ENABLE
10575	051442	042765	000100	000016		BIC	#IE,P.CS1(R5)	:CLEAR INTERRUPT ENABLE
10576	051450	016562	000016	000000	30\$:	MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
10577	051456	000137	052442			JMP	C.ATRN	:RESTORE REGISTERS
10578								
10579						.SBTTL	*SPECIAL COMMAND PROCESSING	
10580								
10581	051462	122765	000141	000001		C.SPEC: CMPB	#RSTAT,P.CMND(R5)	:CHECK IF READ DRIVE STATUS

H16

CZR6MDD PK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 203
*SPECIAL COMMAND PROCESSINGSEQ 0481
SEQ 0202

10582	051470	001132			BNE	10\$	: NO, PROCESS OTHER COMMANDS
10583	051472	016562	000020	000010	MOV	P.CS2(R5),RKCS2(R2)	: LOAD CS2 FOR COMMAND
10584	051500	116565	000007	000017	MOVB	P.CS1H(P5),P.CS1+1(R5)	: STORE BITS 8-15 OF CS1
10585	051506	042765	165777	000016	BIC	*C<CFMT!CDT>,P.CS1(R5)	: CLEAR ALL BITS EXCEPT FORMAT
10586							: AND DRIVE TYPE
10587	051514	112765	000001	000016	MOVB	*DR.SEL,P.CS1(R5)	: STORE COMMAND
10588	051522	016562	000016	000000	MOV	P.CS1(R5),RKCS1(R2)	: ISSUE COMMAND
10589	051530	004737	045140		JSR	PC.W.WTCH	: CALL WATCH-DOG TIMER
10590	051534	016265	000000	000016	MOV	RKCS1(R2),P.CS1(R5)	: STORE COMMAND AND STATUS REG. 1
10591	051542	032765	000200	000016	BIT	*RDY,P.CS1(R5)	: WAIT FOR READY
10592	051550	001767			BEQ	2\$	
10593	051552	004737	050212		JSR	PC.I.CST1	: STORE CONTROLLER REGISTERS
10594	051556	016265	000034	000040	MOV	RKMR2(R2),P.A00(R5)	: STORE STATUS BYTE 00 MESSAGE A
10595	051564	016265	000036	000042	MOV	RKMR3(R2),P.B00(R5)	: STORE STATUS BYTE 00 MESSAGE B
10596	051572	032765	100000	000016	BIT	*CERR,P.CS1(R5)	: CHECK IF CONTROLLER ERROR
10597	051600	001436			BEQ	6\$	: NO, GATHER DRIVE STATUS
10598	051602	105037	003100		CLRB	W.TIME	: RESET WATCH DOG TIMING ON THIS DRIVE
10599	051606	005037	003114		CLR	W.DRV	: CLEAR WATCH DOG COUNT
10600	051612	032765	000400	000014	BIT	*NOCHK,P.PRST(R5)	: CHECK IF NO CHECK MODE
10601	051620	001043			BNE	8\$	: YES, INDICATE NORMAL RETURN
10602	051622	032765	001000	000020	BIT	*MDS,P.CS2(R5)	: CHECK IF MULTIPLE DRIVE SELECT
10603	051630	001043			BNE	9\$	: YES, INDICATE CONTROLLER ERROR
10604	051632	004037	047536		JSR	RD,I.CCLR	: CLEAR ERROR
10605	051636	052442			C.ATRN		: ERROR RETURN
10606	051640	032765	010400	000020	BIT	*NED!UFE,P.CS2(R5)	: CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
10607	051646	001007			BNE	5\$	: REPORT ERROR
10608	051650	032765	000001	000036	BIT	*DRA,P.DS(R5)	: CHECK IF DRIVE AVAILIABLE
10609	051656	001003			BNE	5\$	: YES, REPORT ERROR
10610	051660	052765	010000	000014	BIS	*DRVSZD,P.PRST(R5)	: INDICATE DRIVE IS SEIZED BY OTHER PORT
10611	051666	004737	050522		JSR	PC.R.ABNL	: INDICATE ABNORMAL RETURN
10612	051672	000137	052442		JMP	C.ATRN	: RESTORE REGISTERS
10613							
10614	051676	004037	050274		JSR	RD,I.STAT	: GATHER DRIVE STATUS
10615	051702	052442			C.ATRN		: ERROR RETURN
10616	051704	105037	003100		CLRB	W.TIME	: STOP WATCH-DOG TIMING ON DRIVE
10617	051710	005037	003114		CLR	W.DRV	: RESET WATCH-DOG TIME
10618	051714	032765	000400	000014	BIT	*NOCHK,P.PRST(R5)	: CHECK IF NO CHECK MODE
10619	051722	001402			BEQ	8\$	: NO, REPORT ERROR
10620	051724	005062	000000		CLR	RKCS1(R2)	: CLEAR INTERRUPT ENABLE
10621	051730	004737	050534		JSR	PC.R.NORM	: REPORT COMMAND COMPLETE
10622	051734	000137	052442		JMP	C.ATRN	: RESTORE REGISTERS
10623							
10624	051740	052737	100000	003052	9\$: BIS	*E.MDS,E.CONT	: SET MULTIPLE DRIVE SELECT
10625	051746	004737	050546		JSR	PC.R.CONT	: INDICATE CONTROLLER ERROR
10626	051752	000137	052442		JMP	C.ATRN	
10627							
10628	051756	122765	000140	000001	10\$: CMPB	*RELEAS,P.CMND(R5)	: CHECK IF RELEASE COMMAND
10629	051764	001040			BNE	13\$	: NO, CHECK IF READ ALL HEADERS
10630	051766	010537	003054		MOV	RS,O.WAIT	: STORE PARAMETER BLOCK ADDRESS IN
10631							: WAIT FOR COMMAND
10632	051772	052765	000010	000020	BIS	*RLS,P.CS2(R5)	: SET RELEASE BIT
10633	052000	016562	000020	000010	MOV	P.CS2(R5),RKCS2(R2)	: LOAD CS2 FOR DESELECT
10634	052006	112737	000001	003074	MOVB	*I.I.ISRL	: SET FLAG FOR RELEASE COMMAND
10635	052014	116565	000007	000017	MOVB	P.CS1H(R5),P.CS1+1(R5)	: STORE BITS 8-15 OF CS1
10636	052022	042765	165777	000016	BIC	*C<CFMT!CDT>,P.CS1(R5)	: CLEAR ALL BITS EXCEPT FORMAT
10637							: AND DRIVE TYPE

10638	052030	112765	000101	000016		MOV	#SELDIV,P.CS1(R5) ;STORE COMMAND
10639	052036	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10640	052044	001403				BEQ	11\$;NO DO NOT RESET INTERRUPT ENABLE
10641	052046	042765	000100	000016		BIC	#IE,P.CS1(R5) ;RESET INTERRUPT ENABLE
10642	052054	016562	000016	000000	11\$:	MOV	P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10643	052062	000137	052442			JMP	C.ATRN ;RESTORE REGISTERS
10644							
10645	052066	122765	000164	000001	13\$:	CMPB	#RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
10646	052074	001053				BNE	30\$;NO CHECK IF CONTROLLER CLEAR
10647	052076	010537	003054			MOV	R5,0.WAIT ;SET WAITING FOR COMMAND COMPLETION
10648	052102	016537	000010	003070		MOV	P.BALO(R5),HDR.A0 ;LOAD HEADER ADDRESS
10649	052110	132765	000020	000007		BITB	#B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
10650	052116	001404				BEQ	14\$;YES LOAD 22 IN HEADER COUNT
10651	052120	012737	000024	003072		MOV	#20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
10652	052126	000403				BR	22\$;GO ISSUE READ HEADER COMMAND
10653							
10654	052130	012737	000026	003072	14\$:	MOV	#22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
10655	052136	016562	000002	000020	22\$:	MOV	P.CYL(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10656	052144	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
10657	052152	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10658	052160	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10659	052166	042765	165777	000016		BIC	#1C(CFMT:CDT),P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
10660							AND FORMAT
10661	052174	112765	000125	000016		MOVB	#RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
10662	052202	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10663	052210	001027				BNE	34\$;YES, INDICATE ILLEGAL DRIVER COMMAND
10664	052212	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
10665	052220	000137	052442			JMP	C.ATRN ;RESTORE REGISTERS
10666							
10667	052224	122765	000176	000001	30\$:	CMPB	#CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
10668	052232	001012				BNE	32\$;NO CHECK IF SUBSYSTEM CLEAR
10669	052234	004037	047536			JSR	RD,I.CCLR ;CLEAR CONTROLLER
10670	052240	052442				C.ATRN	ERROR RETURN
10671	052242	032765	000400	000014		BIT	#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10672	052250	001472				BEQ	40\$;NO INDICATE NORMAL RETURN
10673	052252	005062	000000			CLR	RKCS1(R2) ;RESET INTERRUPT ENABLE
10674	052256	000467				BR	40\$;INDICATE NORMAL RETURN
10675							
10676	052260	122765	000177	000001	32\$:	CMPB	#SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
10677	052266	001406				BEQ	36\$;YES, CLEAR SUBSYSTEM
10678	052270	052737	000100	003052	34\$:	BIS	#E.IILD,E.CONT ;SET ILLEGAL DRIVER COMMAND
10679	052276	004737	050546			JSR	PC,R.CONT ;REPORT ERROR
10680	052302	000457				BR	C.ATRN ;RESTORE REGISTERS
10681							
10682	052304	012762	000040	000010	36\$:	MOV	#SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
10683	052312	016265	000000	000016		MOV	RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
10684	052320	032765	100000	000016		BIT	#CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
10685	052326	001406				BEQ	37\$;NO FINISH COMMAND
10686	052330	052737	000001	003052		BIS	#BITO,E.CONT ;SET CLEAR SUBSYSTEM DID NOT CLEAR
10687							CONTROLLER ERROR
10688	052336	004737	050546			JSR	PC,R.CONT ;REPORT ERROR
10689	052342	000437				BR	C.ATRN ;RESTORE REGISTERS
10690							
10691	052344	013746	003060		37\$:	MOV	W.MILI,-(SP) ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
10692							TO DISAPPEAR
10693	052350	016265	000000	000016	38\$:	MOV	RKCS1(R2),P.CS1(R5) ;STORE CS1

```

10694 052356 032765 040000 000016 BIT #DI,P.CS1(R5) ;CHECK IF ATTENTIONS CLEARED
10695 052364 001411 BEQ 39$ ;YES, FINISH COMMAND
10696 052366 005316 DEC (SP) ;DECREMENT 16 MILLISECOND COUNT
10697 052370 001367 BNE 38$ ;CHECK DRIVE INTERRUPT AGAIN
10698 052372 005726 TST (SP)+ ;ADJUST STACK
10699 052374 052737 000040 003052 BIS #E.SCLR,E.CONT ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10700 ;DRIVE ATTENTIONS
10701 052402 004737 050546 JSR PC,R.CONT ;REPORT ERROR
10702 052406 000415 BR C.ATRN ;RESTORE REGISTER
10703
10704 052410 005726 39$: TST (SP)+ ;ADJUST STACK
10705 052412 032765 000400 000014 BIT #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10706 052420 001010 BNE C.ATRN ;YES, RESTORE REGISTERS
10707 052422 112737 177777 003074 MOVB #-1,I.ISRL ;SET INTERRUPT ENABLE SET
10708 052430 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
10709 052436 004737 050534 40$: JSR PC,R.NORM ;INDICATE NORMAL TERMINATION
10710
10711 052442 012637 177776 C.ATRN: MOV (SP)+,PS ;RESTORE PSW
10712 052446 012600 MOV (SP)+,R0 ;RESTORE R0
10713 052450 012601 MOV (SP)+,R1 ;RESTORE R1
10714 052452 012602 MOV (SP)+,R2 ;RESTORE R2
10715 052454 012603 MOV (SP)+,R3 ;RESTORE R3
10716 052456 012604 MOV (SP)+,R4 ;RESTORE R4
10717 052460 012605 MOV (SP)+,R5 ;RESTORE R5
10718 052462 000207 RTS PC ;RETURN
10719
10720 .SBTTL OCTAL TO BINARY CONVERSION ROUTINE
10721
10722 ;*****
10723 ; THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10724 ; WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL
10725 ; IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10726 ; ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
10727 ;*****
10728 ;CALL
10729 ; MOV <ADDRESS OF ASCII STRING>,-(SP)
10730 ; JSR PC,OCTBIN
10731 ; <ADDRESS OF ERROR RETURN>
10732 ; RETURN
10733 ;*****
10734
10735
10736 052464 010046 OCTBIN: MOV R0,-(SP) ;SAVE R0
10737 052466 010146 MOV R1,-(SP) ;SAVE R1
10738 052470 010246 MOV R2,-(SP) ;SAVE R2
10739 052472 016600 000010 MOV 10(SP),R0 ;GET ADDRESS OF ASCII STRING
10740 052476 005001 CLR R1 ;CLEAR DATA WORDS
10741 052500 005002 CLR R2
10742 052502 112046 2$: MOVB (R0)+,-(SP) ;PICK THIS CHARACTER
10743 052504 001423 BEQ 3$ ;IF ZERO GET OUT
10744 052506 121627 000054 CMPB (SP),#'. ;CHECK IF COMMA
10745 052512 001420 BEQ 3$ ;IF COMMA GET OUT
10746 052514 122716 000060 CMPB #'0,(SP) ;MAKE SURE THIS CHARACTER IS
10747 052520 003030 BGT 4$ ; AN OCTAL DIGIT
10748 052522 122716 000067 CMPB #'7,(SP)
10749 052526 002425 BLT 4$

```

K16

CZRMDO RK611/06 SS VERIF 1
CZRMDO.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 206
OCTAL TO BINARY CONVERSION ROUTINE

SEQ 0484
SEQ 0205

```

10750 052530 006301      ASL      R1      : *2
10751 052532 006102      ROL      R2
10752 052534 006301      ASL      R1      : *4
10753 052536 006102      ROL      R2
10754 052540 006301      ASL      R1      : *8
10755 052542 006102      ROL      R2
10756 052544 042716 177770 BIC      #1C7,(SP)    ;STRIP THE ASCII JUNK
10757 052550 062601      ADD      (SP)+,R1    ;ADD THIS DIGIT
10758 052552 000753      BR       2$          ;LOOP
10759 052554 005726      3$:  TST      (SP)+    ;CLEAN PARTIAL FROM STACK
10760 052556 010166 000010  MOV      R1,10(SP) ;SAVE RESULT
10761 052562 010237 052616  MOV      R2,$HIOCT
10762 052566 012602      MOV      (SP)+,R2    ;RESTORE R2
10763 052570 012601      MOV      (SP)+,R1    ;RESTORE R1
10764 052572 012600      MOV      (SP)+,R0    ;RESTORE R0
10765 052574 062716 000002  ADD      #2,(SP)  ;ADJUST RETURN
10766 052600 000207      RTS      PC          ;RETURN
10767
10768 052602 005726      4$:  TST      (SP)+    ;CLEAN UP PARTIAL FROM STACK
10769 052604 012602      MOV      (SP)+,R2    ;RESTORE R2
10770 052606 012601      MOV      (SP)+,R1    ;RESTORE R1
10771 052610 012600      MOV      (SP)+,R0    ;RESTORE R0
10772 052612 013616      MOV      2(SP)+,(SP) ;PUT ADDRESS OF ERROR ROUTINE ON STACK
10773 052614 000207      RTS      PC          ;GO PROCESS ERROR
10774 052616 000000      $HIOCT: .WORD 0      ;HIGH ORDER BITS GO HERE
10775      .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
10776
10777      ;*****
10778      ;THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
10779      ;WITH A RANGE OF 0 TO 2(+33)-1.
10780      ;CALL:
10781      ;      JSR      PC,$RAND      ;CALL THE ROUTINE
10782      ;      RETURN                    ;RETURN HERE THE RANDOM
10783      ;                                ;NUMBER WILL BE IN
10784      ;                                ;$HINUM,$LONUM
10785
10786      $RAND:
10787      MOV      R0,-(SP)    ;PUSH R0 ON STACK
10788      MOV      R1,-(SP)    ;PUSH R1 ON STACK
10789      MOV      R2,-(SP)    ;PUSH R2 ON STACK
10790      MOV      $LONUM,R0   ;SET R0 WITH LOW
10791      MOV      $HINUM,R1   ;SET R1 WITH HIGH
10792      MOV      #-7,R2     ;SET SHIFT COUNT
10793      1$:  ASL      R0      ;SHIFT R0 LEFT AND
10794      ROL      R1          ;ROTATE CARRY INTO R1 AND
10795      INC      R2          ;CHECK FOR DONE
10796      BNE     1$          ;CONTINUE SHIFT LOOP
10797      ADD      $LONUM,R0   ;ADD NUMBER TO MAKE X 129
10798      ADC      R1          ;PROPOGATE CARRY
10799      ADD      $HINUM,R1  ;ADD NUMBER TO MAKE X 129
10800      ADD      #1057,R0   ;ADD LOW CONSTANT
10801      ADC      R1          ;PROPOGATE CARRY
10802      ADD      #47401,R1  ;ADD HIGH CONSTANT
10803      MOV      R0,$LONUM  ;SAVE R0
10804      MOV      R1,$HINUM  ;SAVE R1
10805      MOV      (SP)+,R2   ;POP STACK INTO R2

```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 207
RANDOM NUMBER GENERATOR ROUTINE

L16

SEQ 0485
SEQ 0206

```
10806 052710 012601      MOV      (SP)+,R1      ;; POP STACK INTO R1
10807 052712 012600      MOV      (SP)+,R0      ;; POP STACK INTO R0
10808 052714 000207      RTS       PC          ;; RETURN
10809 052716 176543      $HINUM: .WORD 176543
10810 052720 123456      $LONUM: .WORD 123456
10811                                .SBTTL  TYPE ROUTINE
10812
10813      ;;*****
10814      ;;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10815      ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10816      ;;NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10817      ;;NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10818      ;;NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10819      ;;
10820      ;;CALL:
10821      ;;1) USING A TRAP INSTRUCTION
10822      ;;      TYPE      ,MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
10823      ;;OR
10824      ;;      TYPE
10825      ;;      MESADR
10826      ;;
10827
10828 052722 105737 001157      $TYPE:  TSTB      $TPFLG      ;; IS THERE A TERMINAL?
10829 052726 100002      BPL        1$      ;; BR IF YES
10830 052730 000000      HALT      ;; HALT HERE IF NO TERMINAL
10831 052732 000430      BR        3$      ;; LEAVE
10832 052734 010045      1$:      MOV      R0,-(SP)      ;; SAVE R0
10833 052736 017600 000002      MOV      22(SP),R0      ;; GET ADDRESS OF ASCIZ STRING
10834 052742 122737 000001 001340      CMPB      #APTENV,$ENV      ;; RUNNING IN APT MODE
10835 052750 001011      BNE        62$      ;; NO, GO CHECK FOR APT CONSOLE
10836 052752 132737 000100 001341      BITB      #APTSPool,$ENVm      ;; SPOOL MESSAGE TO APT
10837 052760 001405      BEQ        62$      ;; NO, GO CHECK FOR CONSOLE
10838 052762 010037 052772      MOV      R0,61$      ;; SETUP MESSAGE ADDRESS FOR APT
10839 052766 004737 055420      JSR      PC,$ATY3      ;; SPOOL MESSAGE TO APT
10840 052772 000000      61$:      .WORD      0      ;; MESSAGE ADDRESS
10841 052774 132737 000040 001341      62$:      BITB      #APTCsup,$ENVm      ;; APT CONSOLE SUPPRESSED
10842 053002 001003      BNE        60$      ;; YES, SKIP TYPE OUT
10843 053004 112046      2$:      MOVb      (R0)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
10844 053006 001005      BNE        4$      ;; BR IF IT ISN'T THE TERMINATOR
10845 053010 005726      TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
10846 053012 012600      60$:      MOV      (SP)+,R0      ;; RESTORE R0
10847 053014 062716 000002      3$:      ADD      #2,(SP)      ;; ADJUST RETURN PC
10848 053020 000002      RTI          ;; RETURN
10849 053022 122716 000011      4$:      CMPB      #HT,(SP)      ;; BRANCH IF <HT>
10850 053026 001430      BEQ        8$      ;;
10851 053030 122716 000200      CMPB      #CRLF,(SP)      ;; BRANCH IF NOT <CRLF>
10852 053034 001006      BNE        5$      ;;
10853 053036 005726      TST      (SP)+      ;; POP <CR><LF> EQUIV
10854 053040 104401      TYPE      ;; TYPE A CR AND LF
10855 053042 001315      $CRLF
10856 053044 105017 053200      CLRB      $CHARCNT      ;; CLEAR CHARACTER COUNT
10857 053050 000755      BR        2$      ;; GET NEXT CHARACTER
10858 053052 004737 053134      5$:      JSR      PC,$TYPEC      ;; GO TYPE THIS CHARACTER
10859 053056 123726 001156      6$:      CMPB      $FILLC,(SP)+      ;; IS IT TIME FOR FILLER CHARS.?
10860 053062 001350      BNE        2$      ;; IF NO GO GET NEXT CHAR.
10861 053064 013746 001154      MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
```

M16

CZR6MDO RK611 06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 208
TYPE ROUTINE

SEQ 0486
SEQ 0207

```

10862
10863 053070 105366 000001      7$:   DECB   1(SP)      ;; AND THE NULL CHAR.
10864 053074 002770              BLT    6$           ;; DOES A NULL NEED TO BE TYPED?
10865 053076 004737 053134      JSR    PC,$TYPEC      ;; BR IF NO--GO POP THE NULL OFF OF STACK
10866 053102 105337 053200      DECB   $CHARCNT      ;; GO TYPE A NULL
10867 053106 000770              BR     7$           ;; DO NOT COUNT AS A COUNT
10868
10869
10870      :HORIZONTAL TAB PROCESSOR
10871 053110 112716 000040      8$:   MOVB   #' (SP)      ;; REPLACE TAB WITH SPACE
10872 053114 004737 053134      9$:   JSR    PC,$TYPEC      ;; TYPE A SPACE
10873 053120 132737 000007 053200  BITB   #',$CHARCNT      ;; BRANCH IF NOT AT
10874 053126 001372              BNE    9$           ;; TAB STOP
10875 053130 005726              TST    (SP)+         ;; POP SPACE OFF STACK
10876 053132 000724              BR     2$           ;; GET NEXT CHARACTER
10877 053134 105777 126010      $TYPEC: TSTB   2$TPS      ;; WAIT UNTIL PRINTER IS READY
10878 053140 100375              BPL    $TYPEC
10879 053142 116677 000002 126002  MOVB   2(SP),2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
10880 053150 122766 000015 000002  CMPB   #CR,2(SP)      ;; IS CHARACTER A CARRIAGE RETURN?
10881 053156 001003              BNE    1$           ;; BRANCH IF NO
10882 053160 105037 053200      CLRB   $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
10883 053164 000406              BR     $TYPEX      ;; EXIT
10884 053166 122766 000012 000002 1$:   CMPB   #LF,2(SP)      ;; IS CHARACTER A LINE FEED?
10885 053174 001402              BEQ    $TYPEX      ;; BRANCH IF YES
10886 053176 105227              INCB   (PC)+         ;; COUNT THE CHARACTER
10887 053200 000000      $CHARCNT: .WORD 0           ;; CHARACTER COUNT STORAGE
10888 053202 000207      $TYPEX: RTS    PC
10889
10890
10891
10892
10893      ;*****
10894      ;SBTTL DOUBLE-PRECISION MULTIPLY SUBROUTINE
10895      ;SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
10896      ;USES ALL REGISTERS (R0-R5)
10897      ;
10898      ;ENTER WITH JSR PC,M.DPIM
10899      ;MULTIPLIER IN R2-R3
10900      ;MULTIPLICAND IN R4-R5
10901      ;PRODUCT RETURNED IN R0-R1-R2-R3
10902      ;*****
10903
10904 053204 005000      M.DPIM: CLR    R0           ;CLEAR HI ORDER WORDS
10905 053206 005001      CLR    R1
10906 053210 012746 000041      MOV    #41,-(SP)      ;MOVE 33 (DEC) TO COUNTER
10907 053214 006000      M.DP01: ROR    R0
10908 053216 006001      ROR    R1
10909 053220 006002      ROR    R2           ;SHIFT TO ADD
10910 053222 006003      ROR    R3
10911 053224 103003      BCC    M.DP02      ;NO CARRY NO ADD
10912 053226 060501      ADD    R5,R1
10913 053230 005500      ADC    R0           ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
10914 053232 060400      ADD    R4,R0      ;PRODUCT
10915 053234 005316      M.DP02: DEC    2$P           ;DECREMENT COUNTER
10916 053236 001366      BNE    M.DP01
10917 053240 005726      TST    (SP)+         ;REMOVE THE COUNTER

```

10918 053242 000207
10919
10920
10921
10922
10923
10924
10925
10926
10927
10928
10929
10930
10931
10932
10933 053244 012746 000040
10934 053250 010146
10935 053252 010546
10936 053254 005466 000002
10937 053260 005416
10938 053262 005666 000002
10939 053266 061601
10940 053270 005500
10941 053272 066600 000002
10942 053276 103445
10943 053300 005046
10944 053302 006103
10945 053304 006102
10946 053306 006101
10947 053310 006100
10948 053312 005716
10949 053314 001410
10950 053316 005016
10951 053320 066601 000002
10952 053324 005500
10953 053326 005516
10954 053330 066600 000004
10955 053334 000404
10956 053336 060501
10957 053340 005500
10958 053342 005516
10959 053344 060400
10960 053346 005516
10961 053350 005716
10962 053352 001401
10963 053354 005203
10964 053356 005366 000006
10965 053362 003347
10966 053364 006003
10967 053366 103404
10968 053370 060501
10969 053372 005500
10970 053374 060400
10971 053376 000241
10972 053400 006103
10973 053402 062706 000010

RTS PC

```

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

```

10974 053406 000242
10975 053410 000207
10976 053412 062706 000006
10977 053416 000262
10978 053420 000207
10979
10980
10981
10982
10983
10984
10985
10986
10987
10988
10989
10990
10991
10992

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 ASCIIZ NUMBER.
*CALL

* MOV #PNTR, -(SP) ;; POINTER TO LOW WORD OF BINARY NUMBER
* JSR PC, @#SDB20 ;; CALL THE ROUTINE
* RETURN ;; THE ADDRESS OF THE FIRST ASCIIZ CHAR. IS ON THE STACK

10993 053422 104407
10994 053424 016601 000002
10995 053430 012705 053541
10996 053434 012704 000014
10997 053440 012703 177770
10998 053444 012100
10999 053446 012101
11000 053450 005002
11001 053452 110245
11002 053454 010002
11003 053456 005304
11004 053460 003007
11005 053462 001405
11006 053464 005205
11007 053466 010566 000002
11008 053472 104410
11009 053474 000207
11010 053476 006203
11011 053500 006001
11012 053502 006000
11013 053504 006001
11014 053506 006000
11015 053510 006001
11016 053512 006000
11017 053514 040302
11018 053516 062702 000060
11019 053522 000753
11020 053524 000016

SDB20: SAVREG ;; SAVE ALL REGISTERS
MOV 2(SP), R1 ;; PICKUP THE POINTER TO LOW WORD
MOV #SOCTVL+13., R5 ;; POINTER TO DATA TABLE
MOV #12., R4 ;; DO ELEVEN CHARACTERS
MOV #C7, 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) ;; ASCIIZ 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 #0, R2 ;; MAKE THIS CHAR. ASCII
BR 1\$;; GO PUT IT IN THE DATA TABLE
SOCTVL: .BLKB 14. ;; RESERVE DATA TABLE
.SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

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

* MOV #PNTR, -(SP) ;; POINTER TO LOW WORD OF BINARY NUMBER
* JSR PC, @#SDB20

11021
11022
11023
11024
11025
11026
11027
11028
11029

001

CZR6M00 RK611-06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 211
DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

SEQ 0489
SEQ 0210

```

11030 ;* RETURN ;:THE FIRST ADDRESS OF ASCIIZ
11031 ;:IS ON THE STACK
11032
11033
11034 053542 104407 $0820: SAVREG ;:SAVE REGISTERS
11035 053544 016602 000002 MOV 2(SP),R2 ;:PICKUP THE DATA POINTER
11036 053550 012700 053722 MOV $DECVL,R0 ;:GET ADDRESS OF "SDECVL" STRING
11037 053554 010066 000002 MOV R0,2(SP) ;:PUT ADDRESS OF ASCIIZ STRING ON STACK
11038 053560 012201 MOV (R2)+,R1 ;:PICKUP THE BINARY NUMBER
11039 053562 012202 MOV (R2)+,R2
11040 053564 012737 000012 053640 MOV #10,4$ ;:SET UP TO DO 10 CONVERSIONS
11041 053572 012704 053652 MOV $STNPWR,R4 ;:ADDRESS OF TEN POWER
11042 053576 012705 053654 MOV $STNPWR+2,R5
11043 053602 005003 1$: CLR R3 ;:CLEAR PARTIAL
11044 053604 161401 2$: SUB (R4),R1 ;:SUBTRACT TEN POWER
11045 053606 005602 SBC R2
11046 053610 161502 SUB (R5),R2
11047 053612 002402 BLT 3$ ;:BR IF TEN POWER TO LARGE
11048 053614 005203 INC R3 ;:ADD 1 TO PARTIAL
11049 053616 000772 BR 2$ ;:LOOP
11050 053620 062401 3$: ADD (R4)+,R1 ;:RESTORE SUBTRACTED VALUE
11051 053622 005502 ADC R2
11052 053624 062402 ACJ (R4)+,R2
11053 053626 022525 CMP (R5)+,(R5)+ ;:MOVE TO NEXT TEN POWER
11054 053630 052703 000060 BIS #0,R3 ;:CHANGE PARTIAL TO ASCII
11055 053634 110320 MOVB R3,(R0)+ ;:SAVE IT
11056 053636 005327 DEC (PC)+ ;:DONE?
11057 053640 000000 4$: .WORD 0
11058 053642 001357 BNE 1$ ;:BR IF NO
11059 053644 105020 CLRB (R0)+ ;:TERMINATOR
11060 053646 104410 RESREG ;:RESTORE REGISTERS
11061 053650 000207 RTS PC ;:RETURN
11062 053652 145000 $STNPWR: 145000 ;:1.0E09
11063 053654 035632 35632 ;:1.0E08
11064 053656 160400 160400 ;:1.0E07
11065 053660 002765 2765 ;:1.0E06
11066 053662 113200 113200 ;:1.0E05
11067 053664 000230 230 ;:1.0E04
11068 053666 041100 041100 ;:1.0E03
11069 053670 000017 17 ;:1.0E02
11070 053672 103240 103240 ;:1.0E01
11071 053674 000001 1 ;:1.0E00
11072 053676 023420 23420 ;:1.0E00
11073 053700 000000 0 ;:1.0E00
11074 053702 001750 1750 ;:1.0E00
11075 053704 000000 0 ;:1.0E00
11076 053706 000144 144 ;:1.0E00
11077 053710 000000 0 ;:1.0E00
11078 053712 000012 12 ;:1.0E00
11079 053714 000000 0 ;:1.0E00
11080 053716 000001 1 ;:1.0E00
11081 053720 000000 0 ;:1.0E00
11082 053722 000014 $DECVL: .BLKB 12 ;:RESERVE STORAGE FOR ASCIIZ STRING
11083 .SBTTL TYPE NUMERICAL ASCIIZ STRING SUPPRESS LEADING ZEROS
11084 ;:*****
11085

```

EO1

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 212
TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS

SEQ 0490
SEQ 0211

```

11086 ;*THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
11087 ;*LEADING NUMBERS.
11088 ;*CALL
11089 ;*      MOV      #NUMADR, -(SP)      ;; FIRST ADDRESS OF ASCII STRING
11090 ;*      JSR      PC, @#SSUPRS
11091
11092
11093 $SUPRS: MOV      RO, -(SP)          ;; SAVE RO
11094         MOV      4(SP), RO          ;; PICKUP THE POINTER
11095 1$:      TSTB     (RO)              ;; TERMINATE OR?
11096         BEQ      2$                ;; BR IF YES
11097         CMPB     #'0, (RO)+        ;; IS THIS AN ASCII "0" ?
11098         BEQ      1$                ;; BR IF YES
11099 2$:      DEC      RO                ;; BACKUP BY "1"
11100         MOV      RO, 3$           ;; SAVE FOR TYPING
11101         TYPE     PC               ;; GO TYPE
11102 3$:      .WORD    0                ;; ASCII POINTER GOES HERE
11103         MOV      (SP)+, RO         ;; RESTORE RO
11104         MOV      (SP)+, (SP)       ;; RESTORE THE STACK
11105         RTS      PC               ;; RETURN
11106 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11107
11108 ;*****
11109 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11110 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11111 ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11112 ;*CALL:
11113 ;*      MOV      NUM, -(SP)          ;; NUMBER TO BE TYPED
11114 ;*      TYPOS     N                  ;; CALL FOR TYPEOUT
11115 ;*      .BYTE     N                  ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11116 ;*      .BYTE     M                  ;; M=1 OR 0
11117 ;*                                     ;; 1=TYPE LEADING ZEROS
11118 ;*                                     ;; 0=SUPPRESS LEADING ZEROS
11119 ;*
11120 ;*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11121 ;*STYPOS OR STYPOC
11122 ;*CALL:
11123 ;*      MOV      NUM, -(SP)          ;; NUMBER TO BE TYPED
11124 ;*      TYPON     N                  ;; CALL FOR TYPEOUT
11125 ;*
11126 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11127 ;*CALL:
11128 ;*      MOV      NUM, -(SP)          ;; NUMBER TO BE TYPED
11129 ;*      TYPOC     N                  ;; CALL FOR TYPEOUT
11130 ;*
11131 11131 053776 017646 000000 054221 STYPOS: MOV      @2(SP), -(SP)      ;; PICKUP THE MODE
11132 11132 054002 116637 000001 054221 MOV      1(SP), $OFILL      ;; LOAD ZERO FILL SWITCH
11133 11133 054010 112637 054223 054221 MOV      (SP)+, $OMODE+1    ;; NUMBER OF DIGITS TO TYPE
11134 11134 054014 062716 000002 054221 ADD      #2, (SP)          ;; ADJUST RETURN ADDRESS
11135 11135 054020 000406 054221 BR          STYPON
11136 11136 054022 112737 000001 054221 STYPOC: MOV      #1, $OFILL      ;; SET THE ZERO FILL SWITCH
11137 11137 054030 112737 000006 054223 MOV      #6, $OMODE+1    ;; SET FOR SIX(6) DIGITS
11138 11138 054036 112737 000005 054220 STYPON: MOV      #5, $OCNT      ;; SET THE ITERATION COUNT
11139 11139 054044 010346 054220 MOV      R3, -(SP)        ;; SAVE R3
11140 11140 054046 010446 054220 MOV      R4, -(SP)        ;; SAVE R4
11141 11141 054050 010546 054220 MOV      R5, -(SP)        ;; SAVE R5

```

F01

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 213
BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0491
SEQ 0212

11142	054052	113704	054223	MOV	\$OMODE+1,R4	;;GET THE NUMBER OF DIGITS TO TYPE
11143	054056	005404		NEG	R4	
11144	054060	062704	000006	ADD	#6,R4	;;SUBTRACT IT FOR MAX. ALLOWED
11145	054064	110437	054222	MOV	R4,\$OMODE	;;SAVE IT FOR USE
11146	054070	113704	054221	MOV	\$OFILL,R4	;;GET THE ZERO FILL SWITCH
11147	054074	016605	000012	MOV	12(SP),R5	;;PICKUP THE INPUT NUMBER
11148	054100	005003		CLR	R3	;;CLEAR THE OUTPUT WORD
11149	054102	006105		ROL	R5	;;ROTATE MSB INTO "C"
11150	054104	000404		BR	3\$	;;GO DO MSB
11151	054106	006105		ROL	R5	;;FORM THIS DIGIT
11152	054110	006105		ROL	R5	
11153	054112	006105		ROL	R5	
11154	054114	010503		MOV	R5,R3	
11155	054116	006103		ROL	R3	;;GET LSB OF THIS DIGIT
11156	054120	105337	054222	DEC	\$OMODE	;;TYPE THIS DIGIT?
11157	054124	100016		BPL	7\$	;;BR IF NO
11158	054126	042703	177770	BIC	#177770,R3	;;GET RID OF JUNK
11159	054132	001002		BNE	4\$	;;TEST FOR 0
11160	054134	005704		TST	R4	;;SUPPRESS THIS 0?
11161	054136	001403		BEQ	5\$	;;BR IF YES
11162	054140	005204		INC	R4	;;DON'T SUPPRESS ANYMORE 0'S
11163	054142	052703	000060	BIS	#'0,R3	;;MAKE THIS DIGIT ASCII
11164	054146	052703	000040	BIS	#',R3	;;MAKE ASCII IF NOT ALREADY
11165	054152	110337	054216	MOV	R3,\$\$	;;SAVE FOR TYPING
11166	054156	104401	054216	TYPE	8\$	;;GO TYPE THIS DIGIT
11167	054162	105337	054220	DEC	\$OCNT	;;COUNT BY 1
11168	054166	003347		BGT	2\$	;;BR IF MORE TO DO
11169	054170	002402		BLT	6\$	;;BR IF DONE
11170	054172	005204		INC	R4	;;INSURE LAST DIGIT ISN'T A BLANK
11171	054174	000744		BR	2\$	;;GO DO THE LAST DIGIT
11172	054176	012605		MOV	(SP)+,R5	;;RESTORE R5
11173	054200	012604		MOV	(SP)+,R4	;;RESTORE R4
11174	054202	012603		MOV	(SP)+,R3	;;RESTORE R3
11175	054204	016666	000002 000004	MOV	2(SP),4(SP)	;;SET THE STACK FOR RETURNING
11176	054212	012616		MOV	(SP)+,(SP)	
11177	054214	000002		RTI		;;RETURN
11178	054216	000		.BYTE	0	;;STORAGE FOR ASCII DIGIT
11179	054217	000		.BYTE	0	;;TERMINATOR FOR TYPE ROUTINE
11180	054220	000		.BYTE	0	;;OCTAL DIGIT COUNTER
11181	054221	000		.BYTE	0	;;ZERO FILL SWITCH
11182	054222	000000		.WORD	0	;;NUMBER OF DIGITS TO TYPE
11183				.SBTTL	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE	
11184						
11185						
11186						
11187						
11188						
11189						
11190						
11191						
11192						
11193						
11194						
11195	054224					
11196	054224	010046		MOV	R0,-(SP)	;;PUSH R0 ON STACK
11197	054226	010146		MOV	R1,-(SP)	;;PUSH R1 ON STACK

 ;THIS ROUTINE IS USED TO CHANGE A 15-BIT BINARY NUMBER TO A 5-DIGIT
 ;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 ;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 ;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 ;REPLACED WITH SPACES.
 ;CALL:
 ;* MOV NUM,-(SP) ;;PUT THE BINARY NUMBER ON THE STACK
 ;* TYPDS ;;GO TO THE ROUTINE

\$TYPDS: MOV R0,-(SP) ;;PUSH R0 ON STACK
 MOV R1,-(SP) ;;PUSH R1 ON STACK

GO1

CZR6MDO RK611/06 SS VERIF 1
CZR6MDO.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 214
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0492
SEQ 0213

11198	054230	010246			MOV	R2,-(SP)	;; PUSH R2 ON STACK
11199	054232	010346			MOV	R3,-(SP)	;; PUSH R3 ON STACK
11200	054234	010546			MOV	R5,-(SP)	;; PUSH R5 ON STACK
11201	054236	012746	020200		MOV	#20200,-(SP)	;; SET BLANK SWITCH AND SIGN
11202	054242	016605	000020		MOV	20(SP),R5	;; GET THE INPUT NUMBER
11203	054246	100004			BPL	1\$	;; BR IF INPUT IS POS.
11204	054250	005405			NEG	R5	;; MAKE THE BINARY NUMBER POS.
11205	054252	112766	000055	000001	MOVB	#'-,1(SP)	;; MAKE THE ASCII NUMBER NEG.
11206	054260	005000			CLR	R0	;; ZERO THE CONSTANTS INDEX
11207	054262	012703	054440		MOV	#SDBLK,R3	;; SETUP THE OUTPUT POINTER
11208	054266	112723	000040		MOVB	#',(R3)+	;; SET THE FIRST CHARACTER TO A BLANK
11209	054272	005002			CLR	R2	;; CLEAR THE BCD NUMBER
11210	054274	016001	054430		MOV	\$DTBL(R0),R1	;; GET THE CONSTANT
11211	054300	160105			SUB	R1,R5	;; FORM THIS BCD DIGIT
11212	054302	002402			BLT	4\$	;; BR IF DONE
11213	054304	005202			INC	R2	;; INCREASE THE BCD DIGIT BY 1
11214	054306	000774			BR	3\$	
11215	054310	060105			ADD	R1,R5	;; ADD BACK THE CONSTANT
11216	054312	005702			TST	R2	;; CHECK IF BCD DIGIT=0
11217	054314	001002			BNE	5\$	;; FALL THROUGH IF 0
11218	054316	105716			TSTB	(SP)	;; STILL DOING LEADING 0'S?
11219	054320	100407			BMI	7\$	;; BR IF YES
11220	054322	106316			ASLB	(SP)	;; MSD
11221	054324	103003			BCC	6\$	;; BR IF NO
11222	054326	116663	000001	177777	MOVB	1(SP),-1(R3)	;; YES--SET THE SIGN
11223	054334	052702	000060		BIS	#'0,R2	;; MAKE THE BCD DIGIT ASCII
11224	054340	052702	000040		BIS	#',R2	;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
11225	054344	110223			MOVB	R2,(R3)+	;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
11226	054346	005720			TST	(R0)+	;; JUST INCREMENTING
11227	054350	020027	000010		CMP	R0,#10	;; CHECK THE TABLE INDEX
11228	054354	002746			BLT	2\$	;; GO DO THE NEXT DIGIT
11229	054356	003002			BGT	8\$	;; GO TO EXIT
11230	054360	010502			MOV	R5,R2	;; GET THE LSD
11231	054362	000764			BR	6\$	;; GO CHANGE TO ASCII
11232	054364	105726			TSTB	(SP)+	;; WAS THE LSD THE FIRST NON-ZERO?
11233	054366	100003			BPL	9\$	;; BR IF NO
11234	054370	116663	177777	177776	MOVB	-1(SP),-2(R3)	;; YES--SET THE SIGN FOR TYPING
11235	054376	105013			CLRB	(R3)	;; SET THE TERMINATOR
11236	054400	012605			MOV	(SP)+,R5	;; POP STACK INTO R5
11237	054402	012603			MOV	(SP)+,R3	;; POP STACK INTO R3
11238	054404	012602			MOV	(SP)+,R2	;; POP STACK INTO R2
11239	054406	012601			MOV	(SP)+,R1	;; POP STACK INTO R1
11240	054410	012600			MOV	(SP)+,R0	;; POP STACK INTO R0
11241	054412	104401	054440		TYPE	\$SDBLK	;; NOW TYPE THE NUMBER
11242	054416	016666	000002	000004	MOV	2(SP),4(SP)	;; ADJUST THE STACK
11243	054424	012616			MOV	(SP)+,(SP)	
11244	054426	000002			RTI		;; RETURN TO USER
11245	054430	023420			\$DTBL:	10000.	
11246	054432	001750				1000.	
11247	054434	000144				100.	
11248	054436	000012				10.	
11249	054440	000004			\$SDBLK:	.BLKW 4	
11250					.SBTTL	ERROR HANDLER ROUTINE	
11251							
11252							
11253							

;;*****
;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.

```

11254      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11255      ;*AND GO TO TYPERR ON ERROR
11256      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11257      ;*SW15=1      HALT ON ERROR
11258      ;*SW13=1      INHIBIT ERROR TYPEOUTS
11259      ;*SW10=1      BELL ON ERROR
11260      ;*SW09=1      LOOF ON ERROR
11261      ;*CALL
11262      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
11263
11264      054450      105237      001103      $ERROR:
11265      054450      001775      7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
11266      054454      013777      001102      124456      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
11267      054456      032777      002000      124446      MOV      $TSTNM,$DISPLAY      ;;DISPLAY TEST NUMBER AND ERROR FLAG
11268      054464      001402      1$      BIT      $BIT10,$SWR      ;;BELL ON ERROR?
11269      054472      104401      001310      BEQ      1$      ;;NO - SKIP
11270      054474      005237      001112      TYPE      $BELL      ;;RING BELL
11271      054500      011637      001116      1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
11272      054504      162737      000002      001116      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
11273      054516      117737      124374      001114      SUB      #2,$ERRPC
11274      054524      032777      020000      124406      MOVB      $ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
11275      054532      004737      040776      BIT      $BIT13,$SWR      ;;SKIP TYPEOUT IF SET
11276      054540      104401      001315      BNE      20$      ;;SKIP TYPEOUTS
11277      054544      122737      000001      001340      JSR      PC,TYPERR      ;;GO TO USER ERROR ROUTINE
11278      054552      113737      001114      054566      TYPE      $CRLF
11279      054554      000777      124342      20$:      CMPB      $APTENV,$ENV      ;;RUNNING IN APT MODE
11280      054556      000000      001000      124330      BNE      2$      ;;NO SKIP APT ERROR REPORT
11281      054558      013716      001110      MOV      $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
11282      054560      004737      055430      JSR      PC,$ATY4      ;;REPORT FATAL ERROR TO APT
11283      054562      000777      21$:      .BYTE      0
11284      054564      000777      22$:      .BYTE      0
11285      054566      000777      2$:      BR      22$      ;;APT ERROR LOOP
11286      054568      000001      100001      TST      $SWR      ;;HALT ON ERROR
11287      054570      000000      000000      BPL      3$      ;;SKIP IF CONTINUE
11288      054572      013716      001110      HALT      ;;HALT ON ERROR!
11289      054574      001402      013716      001306      3$:      BIT      $BIT09,$SWR      ;;LOOP ON ERROR SWITCH SET?
11290      054576      001402      013716      001306      BEQ      4$      ;;BR IF NO
11291      054578      001402      013716      001306      MOV      $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
11292      054580      001402      013716      001306      TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
11293      054582      001402      013716      001306      BEQ      5$      ;;BR IF NONE
11294      054584      001402      013716      001306      MOV      $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
11295      054586      022737      025534      000042      5$:      CMP      $SENDAD,$#42      ;;ACT-11 AUTO-ACCEPT?
11296      054588      001001      000000      BNE      6$      ;;BRANCH IF NO
11297      054590      000000      6$:      HALT      ;;YES
11298      054592      000002      6$:      RTI      ;;RETURN
11299      054594      000002      .SBTTL      TTY INPUT ROUTINE
11300      054596      000002      ;*****
11301      054598      000002      .ENABL      LSB
11302      054600      000002      .DSABL      LSB
11303
11304
11305
11306
11307
11308
11309

```

```

11310 ;*****
11311 ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11312 ;CALL:
11313 ;RDCHR
11314 ;RETURN HERE
11315 ;INPUT A SINGLE CHARACTER FROM THE TTY
11316 ;CHARACTER IS ON THE STACK
11317 ;WITH PARITY BIT STRIPPED OFF
11318 $RDCHR: MOV (SP),-(SP) ;PUSH DOWN THE PC
11319 054544 011646 000004 000002 1$: MOV 4(SP),2(SP) ;SAVE THE PS
11320 054654 105777 124264 000004 1$: TSTB 2$TKS ;WAIT FOR
11321 054660 100375 000004 000004 BPL 1$ ;A CHARACTER
11322 054662 117766 124260 000004 MOVB 2$TKB,4(SP) ;READ THE TTY
11323 054670 042766 177600 000004 BIC #1C<177>,4(SP) ;GET RID OF JUNK IF ANY
11324 054676 026627 000004 000023 CMP 4(SP),#23 ;IS IT A CONTROL-S?
11325 054704 001013 000004 000023 BNE 3$ ;BRANCH IF NO
11326 054706 105777 124232 2$: TSTB 2$TKS ;WAIT FOR A CHARACTER
11327 054712 100375 000004 000023 BPL 2$ ;LOOP UNTIL ITS THERE
11328 054714 117746 124226 000004 MOVB 2$TKB,-(SP) ;GET CHARACTER
11329 054720 042716 177600 000021 BIC #1C177,(SP) ;MAKE IT 7-BIT ASCII
11330 054724 022627 000021 000021 CMP (SP)+,#21 ;IS IT A CONTROL-Q?
11331 054730 001366 000004 000140 BNE 2$ ;IF NOT DISCARD IT
11332 054732 000750 000004 000140 BR 1$ ;YES, RESUME
11333 054734 026627 000004 000175 3$: CMP 4(SP),#140 ;IS IT UPPER CASE?
11334 054742 002407 000004 000175 BLT 4$ ;BRANCH IF YES
11335 054744 026627 000004 000175 CMP 4(SP),#175 ;IS IT A SPECIAL CHAR?
11336 054752 003003 000004 000004 BGT 4$ ;BRANCH IF YES
11337 054754 042766 000040 000004 BIC #40,4(SP) ;MAKE IT UPPER CASE
11338 054762 000002 000004 000004 4$: RTI ;GO BACK TO USER
11339 054764 052536 005015 000 $CNTLU: .ASCIZ /1U/15>12> ;CONTROL "U"
11340 054771 136 006507 000012 $CNTLG: .ASCIZ /1G/15>12> ;CONTROL "G"
11341 054776 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
11342 055004 020075 000 000
11343 055007 040 047040 053505 $MNEW: .ASCIZ / NEW = /
11344 055014 036440 000040
11345
11346
11347
11348 .SBTTL POWER DOWN AND UP ROUTINES
11349
11350 ;POWER DOWN ROUTINE
11351 055020 012737 055032 000024 $PWRDN: MOV #PWRUP,PWRVEC ;SET VECTOR FOR POWER UP
11352 055026 000000 000000 000024 HALT ;HANG UP
11353 055030 000776 000000 000024 BR -2
11354
11355 ;POWER UP ROUTINE
11356 055032 005037 055104 $PWRUP: CLR $PWRCT ;WAIT LOOP FOR TTY TO COME UP
11357 055036 005237 055104 4$: INC $PWRCT
11358 055042 001375 000004 000024 BNE 4$
11359 055044 012737 055020 000024 MOV #PWRDN,PWRVEC ;SET VECTOR FOR POWER DOWN
11360 055052 012737 000340 000026 MOV #PR7,PWRVEC+2 ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11361 055060 012737 000340 000036 MOV #PR7,TRAPVEC+2
11362 055066 012706 001100 000036 MOV #STACK,SP ;RE-INITIALIZE THE STACK
11363 055072 104401 055106 TYPE ,PWRMSG ;TYPE "POWER FAILED"
11364 055076 000005 000005 RESET ;CLEAR THE UNIBUS
11365 055100 000177 124002 JMP 2$LPADR ;RESTART THE CURRENT TEST

```

J01

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 217
POWER DOWN AND UP ROUTINESSEQ 0495
SEQ 0216

11366 055101 000000
11367 055106 005015 047520 042527
11368 055114 020122 040506 046111
11369 055122 042105 005015 000
11370 055130

\$PWRC: .WORD 0
\$PWMSG: .ASCIZ <15><12>/POWER FAILED/<15><12>

.EVEN

11371
11372
11373
11374

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

11375
11376
11377

11378 055130 105737 001103
11379 055134 001406
11380 055136 032777 001000 123774
11381 055144 001402
11382 055146 013716 001110
11383 055152 000002

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

11384
11385
11386

.SBTTL SCOPE HANDLER ROUTINE

11387
11388
11389

* 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:

11390
11391
11392

*SW14=1 LOOP ON TEST
*SW11=1 INHIBIT ITERATIONS
*SW09=1 LOOP ON ERROR
*CALL

11393
11394
11395

* SCOPE ; ;SCOPE=IOT

11396
11397
11398

11399
11400

11401 055154 005737 001304
11402 055160 001404
11403 055162 032777 040000 123750
11404 055170 001101
11405 055172 000416

\$SCOPE: TST \$TIMES ;CHECK CURRENT ITERATION NUMBER
BEQ \$XTSTR ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
BIT #BIT14,\$SWR ;LOOP ON PRESENT TEST?
BNE \$OVER ;YES IF SW14=1

11406
11407
11408

*****START OF CODE FOR THE XOR TESTER*****
\$XTSTR: BR 6\$;IF RUNNING ON THE "XOR" TESTER CHANGE
THIS INSTRUCTION TO A "NOP" (NOP=240)

11409 055174 013746 000004
11410 055200 012737 055220 000004
11411 055206 005737 177060
11412 055212 012637 000004

MOV \$ERRVEC, -(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
MOV \$5,\$ERRVEC ;SET FOR TIMEOUT
TST \$177060 ;TIME OUT ON XOR?
MOV (SP)+,\$ERRVEC ;RESTORE THE ERROR VECTOR

11413 055216 000450
11414 055220 022626
11415 055222 012637 000004
11416 055226 000413

5\$: CMP (SP)+,(SP)+ ;GO TO THE NEXT TEST
MOV (SP)+,\$ERRVEC ;CLEAR THE STACK AFTER A TIME OUT
7\$;RESTORE THE ERROR VECTOR
BR 7\$;LOOP ON THE PRESENT TEST

11417 055230 105737 001103
11418 055234 001421
11419 055236 123737 001115 001103

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?

11420 055244 101015
11421 055246 032777 001000 123664

K01

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 218
SCOPE HANDLER ROUTINESEQ 0496
SEQ 0217

```

11422 055254 001404      BEQ      4$      ;;BR IF NO
11423 055256 013737 001110 001106 7$: MOV      $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
11424 055264 000443      BR      $OVER
11425 055266 105037 001103      CLR      $ERFLG      ;;ZERO THE ERROR FLAG
11426 055272 005037 001304      CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11427 055276 000412      BR      1$      ;;ESCAPE TO THE NEXT TEST
11428 055300 032777 004000 123632 3$: BIT      $BIT11,$SWR      ;;INHIBIT ITERATIONS?
11429 055306 001006      BNE      1$      ;;BR IF YES
11430 055310 005237 001104      INC      $ICNT      ;;INCREMENT ITERATION COUNT
11431 055314 023737 001304 001104  CMP      $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
11432 055322 002024      AGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
11433 055324 012737 000001 001104 1$: MOV      $1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
11434 055332 013737 055410 001304  MOV      $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
11435 055340 105237 001102      $SVLAD: INCB     $TSTNM      ;;COUNT TEST NUMBERS
11436 055344 113737 001102 001324  MOV      $TSTNM,$TESTN      ;;SET TEST NUMBER IN APT MAILBOX
11437 055352 011637 001106      MOV      (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
11438 055356 011637 001110      MOV      (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
11439 055362 005037 001306      CLR      $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11440 055366 112737 000001 001115  MOV      $1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11441 055374 013777 001102 123540 $OVER: MOV      $TSTNM,$DISPLAY      ;;DISPLAY TEST NUMBER
11442 055402 013716 001106      MOV      $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
11443 055406 000002      RTI      ;;FIXES PS
11444 055410 003720      $MXCNT: 2000.      ;;MAX. NUMBER OF ITERATIONS
11445      .SBTTL  APT COMMUNICATIONS ROUTINE
11446
11447 *****
11448 055412 112737 000001 055656 $ATY1: MOV      $1,$FFLG      ;;TO REPORT FATAL ERROR
11449 055420 112737 000001 055654 $ATY3: MOV      $1,$MFLG      ;;TO TYPE A MESSAGE
11450 055426 000403      BR      $ATYC
11451 055430 112737 000001 055656 $ATY4: MOV      $1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
11452 055436      $ATYC:
11453 055436 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
11454 055440 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
11455 055442 105737 055654      TSTB     $MFLG      ;;SHOULD TYPE A MESSAGE?
11456 055446 001450      BEQ      5$      ;;IF NOT: BR
11457 055450 122737 000001 001340  CMPB     $APTENV,$ENV      ;;OPERATING UNDER APT?
11458 055456 001031      BNE      3$      ;;IF NOT: BR
11459 055460 132737 000100 001341  BITB     $APTPOOL,$ENVM      ;;SHOULD SPOOL MESSAGES?
11460 055466 001425      BEQ      3$      ;;IF NOT: BR
11461 055470 017600 000004      MOV      $4(SP),R0      ;;GET MESSAGE ADDR.
11462 055474 062766 000002 000004  ADD      $2,$4(SP)      ;;BUMP RETURN ADDR.
11463 055502 005737 001320 1$: TST      $MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
11464 055506 001375      BNE      1$      ;;IF NOT: WAIT
11465 055510 010037 001334      MOV      R0,$MSGAD      ;;PUT ADDR IN MAILBOX
11466 055514 105720 2$: TSTB     (R0)+      ;;FIND END OF MESSAGE
11467 055516 001376      BNE      2$
11468 055520 163700 001334      SUB      $MSGAD,R0      ;;SUB START OF MESSAGE
11469 055524 006200      ASR      R0      ;;GET MESSAGE LNTH IN WORDS
11470 055526 010037 001336      MOV      R0,$MSGGLT      ;;PUT LENGTH IN MAILBOX
11471 055532 012737 000004 001320  MOV      $4,$MSGTYPE      ;;TELL APT TO TAKE MSG.
11472 055540 000413      BR      5$
11473 055542 017637 000004 055566 3$: MOV      $4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
11474 055550 062766 000002 000004  ADD      $2,$4(SP)      ;;BUMP RETURN ADDRESS
11475 055556 013746 177776      MOV      177776,-(SP)      ;;PUSH 177776 ON STACK
11476 055562 004737 052722      JSR      PC,$TYPE      ;;CALL TYPE MACRO
1 477 055566 000000 4$: .WORD 0

```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 219
APT COMMUNICATIONS ROUTINE

LO1

SEQ 0497
SEQ 0218

```
11478 055570 105737 055656 5$:
11479 055570 001416 055656 10$: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
11480 055574 005737 001340 BEQ 12$ ;; IF NOT: BR
11481 055576 001413 001340 TST $ENV ;; RUNNING UNDER APT?
11482 055602 001413 001320 BEQ 12$ ;; IF NOT: BR
11483 055604 005737 001320 11$: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
11484 055610 001375 000004 BNE 11$ ;; IF NOT: WAIT
11485 055612 017637 000004 001322 MOV @4(SP), $FATAL ;; GET ERROR #
11486 055620 062766 000002 000004 ADD #2, 4(SP) ;; BUMP RETURN ADDR.
11487 055626 005237 001320 INC $MSGTYPE ;; TELL APT TO TAKE ERROR
11488 055632 105737 055656 12$: CLRB $FFLG ;; CLEAR FATAL FLAG
11489 055636 105037 055655 CLRB $LFLG ;; CLEAR LOG FLAG
11490 055642 105037 055654 CLRB $MFLG ;; CLEAR MESSAGE FLAG
11491 055646 012601 MOV (SP)+, R1 ;; POP STACK INTO R1
11492 055650 012600 MOV (SP)+, R0 ;; POP STACK INTO R0
11493 055652 000207 RTS PC ;; RETURN
11494 055654 000 $MFLG: .BYTE 0 ;; MESSG. FLAG
11495 055655 000 $LFLG: .BYTE 0 ;; LOG FLAG
11496 055656 000 $FFLG: .BYTE 0 ;; FATAL FLAG
11497 055660 .EVEN
11498 000200 APTSIZE=200
11499 000001 APTENV=001
11500 000100 APTSPool=100
11501 000040 APTCSUP=040
11502 .SBTTL ROUTINE TO SIZE MEMORY
11503 *****
11504 *CALL:
11505 * JSR PC, $SIZE
11506 * RETURN
11507 * $LSTAD WILL CONTAIN:
11508 * WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11509 * WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11510 * $LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11511 * $KT11 IS THE MEMORY MANAGEMENT KEY
11512 * $BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11513 * MUST BE SETUP BEFORE THE CALL
11514 * $BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11515 * DETERMINED BY ROUTINE
11516
11517 $SIZE: MOV R0, -(SP) ;; SAVE R0 ON THE STACK
11518 055660 010046 MOV R1, -(SP) ;; SAVE R1 ON THE STACK
11519 055662 010146 MOV R2, -(SP) ;; SAVE R2 ON THE STACK
11520 055664 010246 MOV R3, -(SP) ;; SAVE R3 ON THE STACK
11521 055666 010346 MOV @#ERRVEC, -(SP) ;; SAVE PRESENT ERROR VECTOR PS & PC
11522 055670 013746 000004 MOV @#ERRVEC+2, -(SP)
11523 055674 013746 000006 MOV SP, R0 ;; SAVE THE STACK POINTER
11524 055700 010600 ;; SET THE ERRVEC PS TO THE PRESENT PS
11525 TRAP ;; PUSH OLD PSW AND PC ON STACK
11526 055702 104400 MOV (SP)+, @#ERRVEC+2 ;; SAVE THE PSW IN @#ERRVEC+2
11527 055704 012637 000006 MOV #3776, R1 ;; SETUP ADDRESS
11528 055710 012701 003776 TSTB (PC)+ ;; USE MEMORY MANAGEMENT?
11529 055714 105727 $KT11: .WORD 200 ;; SET TO USE MEMORY MANAGEMENT
11530 055716 000200 BPL $CORE ;; BR IF NO
11531 055720 100062 MOV # $KT11, @#ERRVEC ;; SET FOR TIMEOUT
11532 055722 012737 056060 000004 TST @#SRO ;; KT11 ARE YOU THERE?
11533 055730 005737 177572
```

MO1

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 220
ROUTINE TO SIZE MEMORY

SEQ 0498
SEQ 0219

```

11534 055734 052737 100000 055716 BIS #100000,$KT11 ;;YES--SET KT11 KEY
11535 055742 005046 CLR -(SP) ;;INITIALIZE FOR "PAR" LOADING
11536 055744 012702 172340 MOV #KIPAR0,R2 ;;ADDRESS OF FIRST "PAR"
11537 055750 012703 000010 MOV #108,R3 ;;LOAD EIGHT "PAR."S" AND EIGHT "PDR."S"
11538 055754 012762 077406 177740 1$: MOV #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
11539 055762 011622 MOV (SP),(R2)+ ;;LOAD "PAR"
11540 055764 062716 000200 ADD #200,(SP) ;;UPDATE FOR NEXT "PAR"
11541 055770 077307 SOB R3,1$ ;;LOOP UNTIL ALL EIGHT ARE LOADED
11542 055772 012742 177600 MOV #177600,-(R2) ;;SETUP KIPAR7 FOR I/O
11543 055776 005042 CLR -(R2) ;;SETUP KIPAR6 FOR TESTING
11544 056000 012737 056016 000004 MOV #2$,2$ERRVEC ;;CATCH TIMEOUT IF NO SR3
11545 056006 012737 000020 172516 MOV #20,2$SR3 ;;ENABLE 22 BIT MODE
11546 056014 000401 BR 3$ ;;THIS PDP-11 HAS A SR3 REGISTER
11547 056016 022626 2$: CMP (SP)+,(SP)+ ;;CLEAN OFF THE STACK--NO SR3
11548 056020 005237 177572 3$: INC 2$SR0 ;;TURN ON MEMORY MANAGEMENT
11549 056024 012737 056050 000004 MOV #SKTOUT,2$ERRVEC ;;SET FOR TIME OUT
11550 056032 005737 143776 4$: TST 2$143776 ;;TRAP ON NON-EX-MEM
11551 056036 062712 000040 ADD #40,(R2) ;;MAKE A 1K STEP
11552 056042 023712 172356 CMP 2$KIPAR7,(R2) ;;LAST ONE?
11553 056046 101371 BHI 4$ ;;NO--TRY IT
11554 056050 011202 SKTOUT: MOV (R2),R2 ;;GET LAST BANK+1
11555 056052 005037 177572 CLR 2$SR0 ;;TURN OFF MEMORY MANAGEMENT
11556 056056 000421 BR $SIZEX
11557 056060 042737 100000 055716 SKTNEX: BIC #100000,$KT11 ;;KT11 NON-EXISTENT
11558 056066 012737 056116 000004 SCORE: MOV #SCORE,2$ERRVEC ;;SET FOR TIMEOUT
11559 056074 005002 CLR R2 ;;SET UP BANK
11560 056076 062701 004000 1$: ADD #4000,R1 ;;INCREMENT BY 1K
11561 056102 062702 000040 ADD #40,R2 ;;1K STEP
11562 056106 005711 TST (R1) ;;TRAP ON TIME OUT
11563 056110 022701 177776 CMP #177776,R1 ;;LAST ONE
11564 056114 001370 BNE 1$ ;;NO--TRY AGAIN
11565 056116 162701 004000 $RCOUT: SUB #4000,R1
11566 056122 162702 000040 $SIZEX: SUB #40,R2 ;;DROP BACK
11567 056126 010006 MOV R0,SP ;;RESTORE THE STACK
11568 056130 012637 000006 MOV (SP)+,2$ERRVEC+2 ;;RESTORE ERROR VECTOR
11569 056134 012637 000004 MOV (SP)+,2$ERRVEC
11570 056140 010137 056162 MOV R1,$LASTAD ;;LAST ADDRESS
11571 056144 010237 056164 MOV R2,$LASTBK ;;LAST BANK
11572 056150 012603 MOV (SP)+,R3 ;;RESTORE R3
11573 056152 012602 MOV (SP)+,R2 ;;RESTORE R2
11574 056154 012601 MOV (SP)+,R1 ;;RESTORE R1
11575 056156 012600 MOV (SP)+,R0 ;;RESTORE R0
11576 056160 000207 RTS PC
11577 056162 000000 $LASTAD: .WORD 0 ;;CONTAINS THE LAST ADDRESS
11578 056164 000000 $LASTBK: .WORD 0 ;;CONTAINS THE LAST BANK
11579 .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
11580
11581 *****
11582 *SAVE R0-R5
11583 *CALL:
11584 * SAVREG
11585 *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11586 *
11587 *TOP---(+16)
11588 * +2---(+18)
11589 * +4---R5

```

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 221
SAVE AND RESTORE RO-R5 ROUTINES

NO1

SEQ 0499
SEQ 0220

```
11590      ; * +6---R4
11591      ; * +8---R3
11592      ; * +10---R2
11593      ; * +12---R1
11594      ; * +14---R0
11595
11596      056166      $SAVREG:
11597      056166      010046      MOV      RO, -(SP)      ;; PUSH RO ON STACK
11598      056170      010146      MOV      R1, -(SP)      ;; PUSH R1 ON STACK
11599      056172      010246      MOV      R2, -(SP)      ;; PUSH R2 ON STACK
11600      056174      010346      MOV      R3, -(SP)      ;; PUSH R3 ON STACK
11601      056176      010446      MOV      R4, -(SP)      ;; PUSH R4 ON STACK
11602      056200      010546      MOV      R5, -(SP)      ;; PUSH R5 ON STACK
11603      056202      016646      000022      MOV      22(SP), -(SP)      ;; SAVE PS OF MAIN FLOW
11604      056206      016646      000022      MOV      22(SP), -(SP)      ;; SAVE PC OF MAIN FLOW
11605      056212      016646      000022      MOV      22(SP), -(SP)      ;; SAVE PS OF CALL
11606      056216      016646      000022      MOV      22(SP), -(SP)      ;; SAVE PC OF CALL
11607      056222      000002      RTI
11608
11609      ; *RESTORE RO-R5
11610      ; *CALL:
11611      ; *      RESREG
11612      056224      $RESREG:
11613      056224      012666      000022      MOV      (SP)+, 22(SP)      ;; RESTORE PC OF CALL
11614      056230      012666      000022      MOV      (SP)+, 22(SP)      ;; RESTORE PS OF CALL
11615      056234      012666      000022      MOV      (SP)+, 22(SP)      ;; RESTORE PC OF MAIN FLOW
11616      056240      012666      000022      MOV      (SP)+, 22(SP)      ;; RESTORE PS OF MAIN FLOW
11617      056244      012605      MOV      (SP)+, R5      ;; POP STACK INTO R5
11618      056246      012604      MOV      (SP)+, R4      ;; POP STACK INTO R4
11619      056250      012603      MOV      (SP)+, R3      ;; POP STACK INTO R3
11620      056252      012602      MOV      (SP)+, R2      ;; POP STACK INTO R2
11621      056254      012601      MOV      (SP)+, R1      ;; POP STACK INTO R1
11622      056256      012600      MOV      (SP)+, R0      ;; POP STACK INTO R0
11623      056260      000002      RTI
11624      .SBTTL  TRAP DECODER
11625
11626      ; *****
11627      ; *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11628      ; *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11629      ; *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11630      ; *GO TO THAT ROUTINE.
11631
11632      056262      010046      $TRAP:  MOV      RO, -(SP)      ;; SAVE RO
11633      056264      016600      000002      MOV      2(SP), RO      ;; GET TRAP ADDRESS
11634      056270      005740      TST      -(RO)      ;; BACKUP BY 2
11635      056272      111000      MOV8     (RO), RO      ;; GET RIGHT BYTE OF TRAP
11636      056274      006300      ASL      RO      ;; POSITION FOR INDEXING
11637      056276      016000      056316      MOV      $TRPAD(RO), RO      ;; INDEX TO TABLE
11638      056302      000200      RTS      RO      ;; GO TO ROUTINE
11639
11640
11641      ; THIS IS USE TO HANDLE THE "GETPRI" MACRO
11642
11643      056304      01164b      $TRAP2: MOV      (SP), -(SP)      ;; MOVE THE PC DOWN
11644      056306      016666      000004      MOV      4(SP), 2(SP)      ;; MOVE THE PSW DOWN
11645      056314      000002      RTI      ;; RESTORE THE PSW
```

11646
11647
11648
11649
11650
11651
11652
11653
11654
11655
11656
11657
11658
11659
11660
11661
11662
11663
11664
11665
11666
11667
11668
11669
11670
11671
11672
11673
11674
11675
11676
11677
11678
11679
11680
11681
11682
11683
11684
11685
11686
11687
11688
11689
11690
11691
11692
11693
11694
11695
11696
11697
11698
11699
11700
11701056316 056304
056320 052722
056322 054022
056324 053776
056326 054036
056330 054224

056332 054644
056334 056166
056336 056224
056340 055130

056342 020040 020040 042524
056350 052123 000040
056354 025052 020040 000
056361 125 044516 052502
056366 020123 040520 020122
056374 051105 000122
056400 047516 026516 054105
056406 051511 020124 042515
056414 000115
056416 047516 026516 054105
056424 051511 020124 051104
056432 000126
056434 047125 052111 043040
056442 042511 042114 042440
056450 051122 000
056453 123 041125 054523
056460 020123 044524 042515
056466 052517 000124
056472 020104 047524 041440
056500 050040 051101 042440
056506 051122 000
056511 103 052040 020117
056516 020104 040520 020122
056524 051105 000122
056530 041501 046040 053517
056536 000
056537 123 042520 042105
056544 046040 051517 000123
056552 046111 020114 052506
056560 041516 000124
056564 043520 020115 051105
056572 000122
056574 047516 026516 054105
056602 051511 020124 052506
056610 041516 000124

.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)
	\$TYPOS	::CALL=TYPOS	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)
	\$RDCMR	::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=SCOPER	TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
TSTMSG:	.ASCIZ	/ TEST /	
AS2SP2:	.ASCIZ	/** /	
EM1:	.ASCIZ	/UNIBUS PAR ERR/	
EM2:	.ASCIZ	/NON-EXIST MEM/	
EM3:	.ASCIZ	/NON-EXIST DRV/	
EM4:	.ASCIZ	/UNIT FIELD ERR/	
EM5:	.ASCIZ	/SUBSYS TIMEOUT/	
EM6:	.ASCIZ	/D TO C PAR ERR/	
EM7:	.ASCIZ	/C TO D PAR ERR/	
EM10:	.ASCIZ	/AC LOW/	
EM11:	.ASCIZ	/SPEED LOSS/	
EM12:	.ASCIZ	/ILL FUNCT/	
EM13:	.ASCIZ	/PGM ERR/	
EM14:	.ASCIZ	/NON-EXIST FUNCT/	

C02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 223
TRAP TABLESEQ 0501
SEQ 0222

11702	056614	051104	020126	054524	EM15:	.ASCIZ	/DRV TYP ERR/
11703	056622	020120	051105	000122			
11704	056630	046506	020124	051105	EM16:	.ASCIZ	/FMT ERP/
11705	056636	000122					
11706	056640	051127	020124	047514	EM17:	.ASCIZ	/WRT LOCK ERR/
11707	056646	045503	042440	051122			
11708	056654	000					
11709	056655	104	053122	052440	EM20:	.ASCIZ	/DRV UNSAFE/
11710	056662	051516	043101	000105			
11711	056670	042523	045505	044440	EM21:	.ASCIZ	/SEEK INC/
11712	056676	041516	000				
11713	056701	103	046131	047440	EM22:	.ASCIZ	/CYL OVRFL0/
11714	056706	051126	046106	000117			
11715	056714	046111	020114	054503	EM23:	.ASCIZ	/ILL CYL ADDR/
11716	056722	020114	042101	051104			
11717	056730	000					
11718	056731	104	053122	047440	EM24:	.ASCIZ	/DRV OFF TRK/
11719	056736	043106	052040	045522			
11720	056744	000					
11721	056745	104	053122	052040	EM25:	.ASCIZ	/DRV TIMING ERR/
11722	056752	046511	047111	020107			
11723	056760	051105	000122				
11724	056764	040504	040524	046040	EM26:	.ASCIZ	/DATA LATE/
11725	056772	052101	000105				
11726	056776	047503	052116	052040	EM27:	.ASCIZ	/CONT TIMEOUT/
11727	057004	046511	047505	052125			
11728	057012	000					
11729	057013	117	020120	047111	EM30:	.ASCIZ	/OP INCOMP/
11730	057020	047503	050115	000			
11731	057025	110	051104	053040	EM31:	.ASCIZ	/HDR VRC ERR/
11732	057032	041522	042440	051122			
11733	057040	000					
11734	057041	104	052101	020101	EM32:	.ASCIZ	/DATA CHK ERR/
11735	057046	044103	020113	051105			
11736	057054	000122					
11737	057056	051127	020124	044103	EM33:	.ASCIZ	/WRT CHK ERR/
11738	057064	020113	051105	000122			
11739	057072	040504	040524	046440	EM34:	.ASCIZ	/DATA MISCOMPARE/
11740	057100	051511	047503	050115			
11741	057106	051101	000105				
11742	057112	047516	042040	053122	EM35:	.ASCIZ	/NO DRV RESPONSE-UFE & NXD/
11743	057120	051040	051505	047520			
11744	057126	051516	026505	043125			
11745	057134	020105	020046	054116			
11746	057142	000104					
11747	057144	051104	020126	051105	EM36:	.ASCIZ	/DRV ERR WILL NOT CLEAR/
11748	057152	020122	044527	046114			
11749	057160	047040	052117	041440			
11750	057166	042514	051101	000			
11751	057173	104	053122	051440	EM37:	.ASCIZ	/DRV STATUS CHANGE WILL NOT CLEAR/
11752	057200	040524	052524	020123			
11753	057206	044103	047101	042507			
11754	057214	053440	046111	020114			
11755	057222	047516	020124	046103			
11756	057230	040505	000122				
11757	057234	052101	023524	020116	EM40:	.ASCIZ	/ATT'N BUT NO STATUS CHANGE OR FAULT/

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 224
TRAP TABLE

SEQ 0502
SEQ 0223

11758	057242	052502	020124	047516	
11759	057250	051440	040524	052524	
11760	057256	020123	044103	047101	
11761	057264	042507	047440	020122	
11762	057272	040506	046125	000124	
11763	057300	052101	023524	020116	EM41: .ASCIZ /ATT'N BUT DRV NOT AVAIL/
11764	057306	052502	020124	051104	
11765	057314	020126	047516	020124	
11766	057322	053101	044501	000114	
11767	057330	052101	023524	020116	EM42: .ASCIZ /ATT'N WHEN NOT EXPECTED/
11768	057336	044127	047105	047040	
11769	057344	052117	042440	050130	
11770	057352	041505	042524	000104	
11771	057360	051105	020122	040507	EM43: .ASCIZ /ERR GATHERING DRV STATUS/
11772	057366	044124	051105	047111	
11773	057374	020107	051104	020126	
11774	057402	052123	052101	051525	
11775	057410	000			
11776	057411	115	046125	020124	EM52: .ASCIZ /MULT DRV SEL/
11777	057416	051104	020126	042523	
11778	057424	000114			
11779	057426	042110	020122	047503	EM53: .ASCIZ /HDR COMP ERR/
11780	057434	050115	042440	051122	
11781	057442	000			
11782	057443	123	041125	054523	EM56: .ASCIZ /SUBSYS TIMEOUT/
11783	057450	020123	044524	042515	
11784	057456	052517	000124		
11785	057462	051105	020122	047111	EM60: .ASCIZ /ERR IN RECAL FOR RECOVERY/
11786	057470	051040	041505	046101	
11787	057476	043040	051117	051040	
11788	057504	041505	053117	051105	
11789	057512	000131			
11790	057514	051104	050117	044520	EM61: .ASCIZ /DROPPING DRV, FATAL ERR/
11791	057522	043516	042040	053122	
11792	057530	020054	040506	040524	
11793	057536	020114	051105	000122	
11794	057544	054503	020114	044515	EM62: .ASCIZ /CYL MISCOMP/
11795	057552	041523	046517	000120	
11796	057560	046103	040505	020122	EM63: .ASCIZ /CLEAR CONTR DID NOT CLEAR ERR/
11797	057566	047503	052116	020122	
11798	057574	044504	020104	047516	
11799	057602	020124	046103	040505	
11800	057610	020122	051105	000122	
11801	057616	047516	040440	052124	EM64: .ASCIZ /NO ATT'N IN ATT'N REG/
11802	057624	047047	044440	020116	
11803	057632	052101	023524	020116	
11804	057640	042522	000107		
11805	057644	047125	047523	044514	EM65: .ASCIZ /UNSOLICITED ATT'N/
11806	057652	044503	042524	020104	
11807	057660	052101	023524	000116	
11808	057666	047125	054105	020120	EM66: .ASCIZ /UNEXP DATA ERR/
11809	057674	040504	040524	042440	
11810	057702	051122	000		
11811	057705	101	052124	047047	EM67: .ASCIZ /ATT'N DID NOT RESET WITH CLEAR/
11812	057712	042040	042111	047040	
11813	057720	052117	051040	051505	

E02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 225
TRAP TABLESEQ 0503
SEQ 0224

11814	057726	052105	053440	052111	
11815	057734	020110	046103	040505	
11816	057742	000122			
11817	057744	052523	051502	051531	EM70: .ASCIZ /SUBSYS CLEAR DID NOT CLEAR DRV ATT'N/
11818	057752	041440	042514	051101	
11819	057760	042040	042111	047040	
11820	057766	052117	041440	042514	
11821	057774	051101	042040	053122	
11822	060002	040440	052124	047047	
11823	060010	000			
11824	060011	104	052101	020101	EM71: .ASCIZ /DATA LATE WHEN READING HDR/
11825	060016	040514	042524	053440	
11826	060024	042510	020116	042522	
11827	060032	042101	047111	020107	
11828	060040	042110	000122		
11829	060044	047503	052116	020122	EM72: .ASCIZ /CONTR ERR WHEN DRIVER SERV/
11830	060052	051105	020122	044127	
11831	060060	047105	042040	044522	
11832	060066	042526	020122	042523	
11833	060074	053122	000		
11834	060077	104	053122	042040	EM73: .ASCIZ /DRV DET PAR ERR/
11835	060104	052105	050040	051101	
11836	060112	042440	051122	000	
11837	060117	125	042116	043105	EM74: .ASCIZ /UNDEF ERR/
11838	060124	042440	051122	000	
11839	060131	115	051101	044513	EM75: .ASCIZ /MARKING THIS SECTOR BAD/
11840	060136	043516	052040	044510	
11841	060144	020123	042523	052103	
11842	060152	051117	041040	042101	
11843	060160	000			
11844	060161	102	042101	042040	EM76: .ASCIZ /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
11845	060166	052101	020101	042526	
11846	060174	044522	023506	020116	
11847	060202	044527	044124	051040	
11848	060210	040505	027104	042440	
11849	060216	041503	047440	020106	
11850	060224	040514	052123	051040	
11851	060232	052105	054522	044440	
11852	060240	035123	000		
11853	060243	122	052105	054522	EM77: .ASCIZ /RETRY SUCCESSFUL/
11854	060250	051440	041525	042503	
11855	060256	051523	052506	000114	
11856	060264	042522	051124	020131	EM100: .ASCIZ /RETRY UNSUCCESSFUL/
11857	060272	047125	052523	041503	
11858	060300	051505	043123	046125	
11859	060306	000			
11860	060307	103	047101	047516	EM101: .ASCIZ /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
11861	060314	020124	044506	042116	
11862	060322	040440	053040	046101	
11863	060330	042111	044040	040505	
11864	060336	042504	020122	047111	
11865	060344	052040	040522	045503	
11866	060352	045040	051525	020124	
11867	060360	042522	042101	000	
11868	060365	102	042101	051440	EM102: .ASCIZ /BAD SECT ERR ON SECT NOT LISTED BAD/
11869	060372	041505	020124	051105	

F02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 226
TRAP TABLESEQ 0504
SEQ 0225

11870	060400	020122	047117	051440	
11871	060406	041505	020124	047516	
11872	060414	020124	044514	052123	
11873	060422	042105	041040	042101	
11874	060430	000			
11875	060431	124	046511	042105	EM103: .ASCIZ /TIMED-OUT ON READ HDR/
11876	060436	047455	052125	047440	
11877	060444	020116	042522	042101	
11878	060452	044040	051104	000	
11879	060457	124	046511	042105	EM104: .ASCIZ /TIMED-OUT ON SEEK/
11880	060464	047455	052125	047440	
11881	060472	020116	042523	045505	
11882	060500	000			
11883	060501	104	053122	051440	EM105: .ASCIZ /DRV SIEZED BY OTHER PORT/
11884	060506	042511	042532	020104	
11885	060514	054502	047440	044124	
11886	060522	051105	050040	051117	
11887	060530	000124			
11888	060532	040504	040524	046440	EM106: .ASCIZ /DATA MISCMPR WHILE BAI SET/
11889	060540	051511	046503	051120	
11890	060546	053440	044510	042514	
11891	060554	041040	044501	051440	
11892	060562	052105	000		
11893	060565	116	020117	042516	EM107: .ASCIZ /NO NEM ERR WHEN REF'ING LOC 760000/
11894	060572	020115	051105	020122	
11895	060600	044127	047105	051040	
11896	060606	043105	044447	043516	
11897	060614	046040	041517	033440	
11898	060622	030066	030060	000060	
11899	060630	047111	051124	052120	EM110: .ASCIZ /INTRPT WHEN CNTRLR NOT RDY/
11900	060636	053440	042510	020116	
11901	060644	047103	051124	051114	
11902	060652	047040	052117	051040	
11903	060660	054504	000		
11904	060663	116	020117	052101	EM111: .ASCIZ /NO ATT'N ON SEEK/
11905	060670	023524	020116	047117	
11906	060676	051440	042505	000113	
11907	060704	051104	020126	054503	EM112: .ASCIZ /DRV CYL INCORRECT/
11908	060712	020114	047111	047503	
11909	060720	051122	041505	000124	
11910	060726	041101	051117	026524	EM113: .ASCIZ /ABORT- CAN'T READ BSF/
11911	060734	041440	047101	052047	
11912	060742	051040	040505	020104	
11913	060750	051502	000106		
11914	060754	052113	030461	043040	EM114: .ASCIZ /KT11 FAILURE/
11915	060762	044501	052514	042522	
11916	060770	000			
11917	060771	115	046505	050040	EM115: .ASCIZ /MEM PAR ERR/
11918	060776	051101	042440	051122	
11919	061004	000			
11920	061005	103	051125	042522	DH100: .ASCIZ /CURRENT CMD :/
11921	061012	052116	041440	042115	
11922	061020	035040	000		
11923	061023	120	042522	020126	DH105: .ASCIZ /PREV CMD :/
11924	061030	046503	020104	000072	
11925	061036	045522	030466	020061	DH200: .ASCIZ /RK611 REGS :/

G02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 227
TRAP TABLESEQ 0505
SEQ 0226

11926	061044	042522	051507	035040	
11927	061052	000			
11928	061053	122	046505	044501	DH500: .ASCIZ /REMAINING REGS NOT VALID/
11929	061060	044516	043516	051040	
11930	061066	043505	020123	047516	
11931	061074	020124	040526	044514	
11932	061102	000104			
11933	061104	044124	020105	047506	DH501: .ASCIZ /THE FOLLOWING REG DATA MAY BE INCORRECT :/
11934	061112	046114	053517	047111	
11935	061120	020107	042522	020107	
11936	061126	040504	040524	046440	
11937	061134	054501	041040	020105	
11938	061142	047111	047503	051122	
11939	061150	041505	020124	000072	
11940	061156	051105	020122	041520	DH101: .ASCIZ /ERR PC DRIVE CMND CYLNDR TRACK SECTOR WD CNT/
11941	061164	020040	051104	053111	
11942	061172	020105	020040	046503	
11943	061200	042116	020040	020040	
11944	061206	054503	047114	051104	
11945	061214	020040	051124	041501	
11946	061222	020113	020040	042523	
11947	061230	052103	051117	020040	
11948	061236	042127	041440	052116	
11949	061244	000			
11950	061245	110	020111	040502	DH102: .ASCIZ /HI BA LO BA/
11951	061252	020040	046040	020117	
11952	061260	040502	000		
11953	061263	120	042522	027126	DH103: .ASCIZ /PREV. UNIBUS MAP :/
11954	061270	052440	044516	052502	
11955	061276	020123	040515	020120	
11956	061304	000072			
11957	061306	044510	051040	043505	DH104: .ASCIZ /HI REGD LO REGD/
11958	061314	020060	047514	051040	
11959	061322	043505	000060		
11960	061326	052503	027122	052440	DH106: .ASCIZ /CUR. UNIBUS MAP :/
11961	061334	044516	052502	020123	
11962	061342	040515	020120	000072	
11963	061350	045522	051503	020061	DH201: .ASCIZ /RKCS1 RKCS2 RKDC RKDA RKWCR RKBA RKASOF/
11964	061356	020040	045522	051503	
11965	061364	020062	020040	045522	
11966	061372	041504	020040	020040	
11967	061400	045522	040504	020040	
11968	061406	020040	045522	041527	
11969	061414	020122	020040	045522	
11970	061422	040502	020040	020040	
11971	061430	045522	051501	043117	
11972	061436	000			
11973	061437	122	042113	020123	DH202: .ASCIZ /RKDS RKER/
11974	061444	020040	051040	042513	
11975	061452	000122			
11976	061454	030101	020060	020040	DH203: .ASCIZ /A00 B00 A01 B01 A02 B02 A03 B03/
11977	061462	020040	030102	020060	
11978	061470	020040	020040	030101	
11979	061476	020061	020040	020040	
11980	061504	030102	020061	020040	
11981	061512	020040	030101	020062	

H02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 228
TRAP TABLESEQ 0506
SEQ 0227

11982	061520	020040	020040	030102	
11983	061526	020062	020040	020040	
11984	061534	C 0101	020063	020040	
11985	061542	020040	030102	000063	
11986	061550	042110	020122	044123	DH601: .ASCIZ /HDR SHOULD BE: /
11987	061556	052517	042114	041040	
11988	061564	035105	000		
11989	061567	110	051104	047440	DH602: .ASCIZ /HDR OF SECTOR 0 THIS CYL IS: /
11990	061574	020106	042523	052103	
11991	061602	051117	030040	052040	
11992	061610	044510	020123	054503	
11993	061616	020114	051511	000072	
11994	061624	040520	045503	040440	DH604: .ASCIZ /PACK ADDR OF ERR(S) : /
11995	061632	042104	020122	043117	
11996	061640	042440	051122	051450	
11997	061646	020051	000072		
11998	061652	054503	047114	051104	DH6041: .ASCIZ /CYLNR TRACK SECTOR/
11999	061660	020040	051124	041501	
12000	061666	020113	020040	042523	
12001	061674	052103	051117	000	
12002	061701	103	046131	042116	DH6042: .ASCIZ /CYLNR TRACK/
12003	061706	020122	052040	040522	
12004	061714	045503	000		
12005	061717	105	041503	042040	DH605: .ASCIZ /ECC DATA IS: /
12006	061724	052101	020101	051511	
12007	061732	000072			
12008	061734	047520	020123	020040	DH6051: .ASCIZ /POS PAT CORRECTABLE? /
12009	061742	020040	040520	020124	
12010	061750	020040	020040	047503	
12011	061756	051122	041505	040524	
12012	061764	046102	037505	000	
12013	061771	127	051117	030504	DH606: .ASCIZ /WORD1 WORD2 WORD3/
12014	061776	020040	053440	051117	
12015	062004	031104	020040	053440	
12016	062012	051117	031504	000	
12017	062017	102	042101	044040	DH607: .ASCIZ /BAD HEADER IS : /
12018	062024	040505	042504	020122	
12019	062032	051511	031504	000	
12020	062037	127	021504	020043	DH701: .ASCIZ /WD # GOOD BAD HI PHY LO PHY VRT AD KIPAR6/
12021	062044	020040	043440	047517	
12022	062052	020104	020040	041040	
12023	062060	042101	020040	020040	
12024	062066	044040	020111	044120	
12025	062074	020131	046040	020117	
12026	062102	044120	020131	053040	
12027	062110	052122	040440	020104	
12028	062116	045440	050111	051101	
12029	062124	000066			
12030	062126	047507	042117	020040	DH702: .ASCIZ /GOOD BAD/
12031	062134	020040	040502	000104	
12032	062142	040506	046111	047111	DH703: .ASCIZ /FAILING DATA WORD : /
12033	062150	020107	040504	040524	
12034	062156	053440	051117	020104	
12035	062164	000072			
12036	062166	051123	004460	051123	DH704: .ASCIZ /SRO SR1 SR2 SR3/
12037	062174	004461	051123	004462	

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 229
TRAP TABLE

SEQ 0507
SEQ 0228

12038	062202	051123	000063			
12039	062206	052113	030461	051040	DH705:	.ASCIZ /KT11 REGS :/
12040	062214	043505	020123	000072		
12041	062222	042515	020115	042522	DH706:	.ASCIZ /MEM REGS :/
12042	062230	051507	035040	000		
12043	062235	120	004503	047514	DH707:	.ASCIZ /PC LOERAD HIERAD MEMSYS/
12044	062242	051105	042101	044011		
12045	062250	042511	040522	004504		
12046	062256	042515	051515	051531		
12047	062264	000				
12048	062265	120	004503	051503	DH710:	.ASCIZ /PC CSR AD CSR/
12049	062272	020122	042101	041411		
12050	062300	051123	000			
12051	062303	103	046131	047111	DH711:	.ASCIZ /CYLINDERS :/
12052	062310	042504	051522	035040		
12053	062316	000				
12054	062317	122	042113	020123	DH204:	.ASCIZ /RKDS RKER RKMR2 RKMR3/
12055	062324	020040	051040	042513		
12056	062332	020122	020040	051040		
12057	062340	046513	031122	020040		
12058	062346	051040	046513	031522		
12059	062354	000				
12060	062355	105	051122	052040	DH502:	.ASCIZ /ERR TRYING TO GET FAILURE INFO/
12061	062362	054522	047111	020107		
12062	062370	047524	043440	052105		
12063	062376	043040	044501	052514		
12064	062404	042522	044440	043116		
12065	062412	000117				
12066	062414	023462	042116	052040	DH503:	.ASCIZ /2'ND TIMEOUT OCCURRED GETTING DRV STATUS/
12067	062422	046511	047505	052125		
12068	062430	047440	041503	051125		
12069	062436	042522	020104	042507		
12070	062444	052124	047111	020107		
12071	062452	051104	020126	052123		
12072	062460	052101	051525	000		
12073	062465	043	047440	020106	DH800:	.ASCIZ /# OF RETRIES:/
12074	062472	042522	051124	042511		
12075	062500	035123	000			
12076		062504				
12077	062504	001116	005510	005506	DT100:	.EVEN \$ERRPC,DRIVE,ERRCOM,\$REG0,\$REG1,\$REG2,\$REG3
12078	062512	001162	001164	001166		
12079	062520	001170				
12080	062522	001256	001260		DT102:	.WORD \$REG36,\$REG37
12081	062526	001174	001176	001200	DT201:	.WORD \$REG5,\$REG6,\$REG7,\$REG10,\$REG11,\$REG12,\$REG13
12082	062534	001202	001204	001206		
12083	062542	001210				
12084	062544	001212	001214		DT202:	.WORD \$REG14,\$REG15
12085	062550	001216	001220	001222	DT203:	.WORD \$REG16,\$REG17,\$REG20,\$REG21,\$REG22,\$REG23,\$REG24,\$REG25
12086	062556	001224	001226	001230		
12087	062564	001232	001234			
12088	062570	001174	001176	001200	DT601:	.WORD \$REG5,\$REG6,\$REG7
12089	062576	001202	001204	001206	DT602:	.WORD \$REG10,\$REG11,\$REG12,\$REG13,\$REG14,\$REG15,\$REG16
12090	062604	001210	001212	001214		
12091	062612	001216				
12092	062614	005264	005262		DT103:	.WORD PRMPHO,PRMPLO
12093						

J02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 230
TRAP TABLESEQ 0508
SEQ 0229

12094 062620 000006
12095 062622 000
12096 062623 000
12097 062624 061156
12098 062626 007 000
12099 062630 061245
12100 062632 002 000
12101 062634 061036
12102 062636 000 000
12103 062640 061350
12104 062642 007 000
12105 062644 061053
12106 062646 000 000
12107
12108 062650 000007
12109 062652 000 000
12110 062654 061156
12111 062656 007 000
12112 062660 061245
12113 062662 002 000
12114 062664 061036
12115 062666 000 000
12116 062670 061350
12117 062672 007 000
12118 062674 061437
12119 062676 002 000
12120 062700 061454
12121 062702 010 000
12122
12123 062704 000010
12124 062706 000 000
12125 062710 061156
12126 062712 007 000
12127 062714 061245
12128 062716 002 000
12129 062720 061036
12130 062722 000 000
12131 062724 061350
12132 062726 007 000
12133 062730 061104
12134 062732 000 000
12135 062734 061437
12136 062736 002 000
12137 062740 061454
12138 062742 010 000
12139
12140 062744 000003
12141 062746 000 000
12142 062750 061156
12143 062752 007 000
12144 062754 061245
12145 062756 002 000
12146
12147 062760 000007
12148 062762 000 000
12149 062764 061156

DF01: .WORD 6 ; NUMBER OF HEADER LINES
.BYTE 0 ; NUMBER OF COL FOR FIRST HDR
.BYTE 0 ; ALL CHARACTERS OCTAL
.WORD DH101 ; SECOND HDR LINE
.BYTE 7,0 ; NUM OF COL-ALL OCTAL
.WORD DH102
.BYTE 2,0
.WORD DH200
.BYTE 0,0
.WORD DH201
.BYTE 7,0
.WORD DH500
.BYTE 0,0
DF02: .WORD 7
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
.WORD DH200
.BYTE 0,0
.WORD DH201
.BYTE 7,0
.WORD DH202
.BYTE 2,0
.WORD DH203
.BYTE 10,0
DF03: .WORD 10
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
.WORD DH200
.BYTE 0,0
.WORD DH201
.BYTE 7,0
.WORD DH501
.BYTE 0,0
.WORD DH202
.BYTE 2,0
.WORD DH203
.BYTE 10,0
DF04: .WORD 3
.BYTE 0,0
.WORD DH101
.BYTE 7,0
.WORD DH102
.BYTE 2,0
DF05: .WORD 7 ; OPI, HVRC
.BYTE 0,0
.WORD DH101

;"THE FOLLOWING REG DATA MB INCORRECT"

K02

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 231
TRAP TABLESEQ 0509
SEQ 0230

12150	062766	007	000		.BYTE	7,0
12151	062770	061245			.WORD	0H102
12152	062772	002	000		.BYTE	2,0
12153	062774	061550			.WORD	0H601
12154	062776	000	000		.BYTE	0,0
12155	063000	061771			.WORD	0H606
12156	063002	003	000		.BYTE	3,0
12157	063004	061567			.WORD	0H602
12158	063006	000	000		.BYTE	0,0
12159	063010	061771			.WORD	0H606
12160	063012	003	000		.BYTE	3,0
12161						
12162	063014	000007		DF07:	.WORD	7
12163	063016	000	000		.BYTE	0,0
12164	063020	061156			.WORD	0H101
12165	063022	007	000		.BYTE	7,0
12166	063024	061245			.WORD	0H102
12167	063026	002	000		.BYTE	2,0
12168	063030	061624			.WORD	0H604
12169	063032	000	000		.BYTE	0,0
12170	063034	061652			.WORD	0H6041
12171	063036	003	000		.BYTE	3,0
12172	063040	061717			.WORD	0H605
12173	063042	000	000		.BYTE	0,0
12174	063044	061734			.WORD	0H6051
12175	063046	003	000		.BYTE	3,0
12176						
12177	063050	000005		DF10:	.WORD	5
12178	063052	000	000		.BYTE	0,0
12179	063054	061156			.WORD	0H101
12180	063056	007	000		.BYTE	7,0
12181	063060	061245			.WORD	0H102
12182	063062	002	000		.BYTE	2,0
12183	063064	061624			.WORD	0H604
12184	063066	000	000		.BYTE	0,0
12185	063070	061652			.WORD	0H6041
12186	063072	003	000		.BYTE	3,0
12187						
12188	063074	000004		DF11:	.WORD	4
12189	063076	000	000		.BYTE	0,0
12190	063100	061624			.WORD	0H604
12191	063102	000	000		.BYTE	0,0
12192	063104	061652			.WORD	0H6041
12193	063106	003	000		.BYTE	3,0
12194	063110	062037			.WORD	0H701
12195	063112	000	000		.BYTE	0,0
12196						
12197	063114	000006		DF12:	.WORD	6
12198	063116	000	000		.BYTE	0,0
12199	063120	061156			.WORD	0H101
12200	063122	007	000		.BYTE	7,0
12201	063124	061245			.WORD	0H102
12202	063126	002	000		.BYTE	2,0
12203	063130	061036			.WORD	0H200
12204	063132	000	000		.BYTE	0,0
12205	063134	061350			.WORD	0H201

L02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 232
TRAP TABLE

SEQ 0510
SEQ 0231

12206	063136	007	000	.BYTE	7,0	
12207	063140	062317		.WORD	DH204	
12208	063142	004	000	.BYTE	4,0	
12209						
12210	063144	000006		DF13:	.WORD	6
12211	063146	000	000		.BYTE	0,0
12212	063150	061156			.WORD	DH101
12213	063152	007	000		.BYTE	7,0
12214	063154	061245			.WORD	DH102
12215	063156	002	000		.BYTE	2,0
12216	063160	061550			.WORD	DH601
12217	063162	000	000		.BYTE	0,0
12218	063164	061771			.WORD	DH606
12219	063166	003	000		.BYTE	3,0
12220	063170	062355			.WORD	DH502
12221	063172	000	000		.BYTE	0,0
12222						
12223	063174	000006		DF14:	.WORD	6
12224	063176	000	000		.BYTE	0,0
12225	063200	061156			.WORD	DH101
12226	063202	007	000		.BYTE	7,0
12227	063204	061245			.WORD	DH102
12228	063206	002	000		.BYTE	2,0
12229	063210	061550			.WORD	DH601
12230	063212	000	000		.BYTE	0,0
12231	063214	061771			.WORD	DH606
12232	063216	003	000		.BYTE	3,0
12233	063220	062355			.WORD	DH502
12234	063222	000	000		.BYTE	0,0
12235						
12236	063224	000011		DF15:	.WORD	11
12237	063226	000	000		.BYTE	0,0
12238	063230	061156			.WORD	DH101
12239	063232	007	000		.BYTE	7,0
12240	063234	061145			.WORD	DH102
12241	063236	002	000		.BYTE	2,0
12242	063240	061036			.WORD	DH200
12243	063242	000	000		.BYTE	0,0
12244	063244	061350			.WORD	DH201
12245	063246	007	000		.BYTE	7,0
12246	063250	061437			.WORD	DH202
12247	063252	002	000		.BYTE	2,0
12248	063254	062414			.WORD	DH503
12249	063256	000	000		.BYTE	0,0
12250	063260	061104			.WORD	DH501
12251	063262	000	000		.BYTE	0,0
12252	063264	061454			.WORD	DH203
12253	063266	010	000		.BYTE	10,0
12254						
12255	063270	000011		DF16:	.WORD	11
12256	063272	000	000		.BYTE	0,0
12257	063274	061156			.WORD	DH101
12258	063276	007	000		.BYTE	7,0
12259	063300	061245			.WORD	DH102
12260	063302	002	000		.BYTE	2,0
12261	063304	061036			.WORD	DH200

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

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

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 233
TRAP TABLE

SEQ 0511
SEQ 0232

12262	063306	000	000	.BYTE	0,0
12263	063310	061350		.WORD	04201
12264	063312	007	000	.BYTE	7,0
12265	063314	061437		.WORD	04202
12266	063316	002	000	.BYTE	2,0
12267	063320	062355		.WORD	04502
12268	063322	000	000	.BYTE	0,0
12269	063324	061104		.WORD	04501
12270	063326	000	000	.BYTE	0,0
12271	063330	061454		.WORD	04203
12272	063332	010	000	.BYTE	10,0
12273					
12274	063334	000005		DF17: .WORD	5
12275	063336	000	000	.BYTE	0,0
12276	063340	061156		.WORD	04101
12277	063342	007	000	.BYTE	7,0
12278	063344	061245		.WORD	04102
12279	063346	002	000	.BYTE	2,0
12280	063350	062303		.WORD	04711
12281	063352	000	000	.BYTE	0,0
12282	063354	062126		.WORD	04702
12283	063356	002	000	.BYTE	2,0
12284					
12285	063360	000002		DF20: .WORD	2
12286	063362	000	000	.BYTE	0,0
12287	063364	061306		.WORD	04104
12288	063366	002	000	.BYTE	2,0
12289					
12290	063370	000002		DF21: .WORD	2
12291	063372	000	000	.BYTE	0,0
12292	063374	061734		.WORD	046051
12293	063376	003	000	.BYTE	3,0
12294					
12295	063400	000006		DF22: .WORD	6
12296	063402	000	000	.BYTE	0,0
12297	063404	061005		.WORD	04100
12298	063406	000	000	.BYTE	0,0
12299	063410	061156		.WORD	04101
12300	063412	007	000	.BYTE	7,0
12301	063414	061245		.WORD	04102
12302	063416	002	000	.BYTE	2,0
12303	063420	061326		.WORD	04106
12304	063422	000	000	.BYTE	0,0
12305	063424	061306		.WORD	04104
12306	063426	002	000	.BYTE	2,0
12307					
12308	063430	000002		DF23: .WORD	2
12309	063432	000	000	.BYTE	0,0
12310	063434	062465		.WORD	04800
12311	063436	001	000	.BYTE	1,0
12312					
12313	063440	000001		DF24: .WORD	1
12314	063442	002	000	.BYTE	2,0
12315					
12316	063444	000000		DF25: .WORD	0
12317	063446	007	000	.BYTE	7,0

N02

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 234
TRAP TABLE

SEQ 0512
SEQ 0233

12318						
12319	063450	000002			DF26:	.WORD 2
12320	063452	002	000			.BYTE 2,0
12321	063454	061454				.WORD 04203
12322	063456	010	000			.BYTE 10,0
12323						
12324	063460	000005			DF27:	.WORD 5
12325	063462	000	000			.BYTE 0,0
12326	063464	061156				.WORD 04101
12327	063466	007	000			.BYTE 7,0
12328	063470	061245				.WORD 04102
12329	063472	002	000			.BYTE 2,0
12330	063474	062142				.WORD 04703
12331	063476	000	000			.BYTE 0,0
12332	063500	062126				.WORD 04702
12333	063502	002	000			.BYTE 2,0
12334						
12335	063504	000005			DF30:	.WORD 5
12336	063506	000	000			.BYTE 0,0
12337	063510	061156				.WORD 04101
12338	063512	007	000			.BYTE 7,0
12339	063514	061245				.WORD 04102
12340	063516	002	000			.BYTE 2,0
12341	063520	062206				.WORD 04705
12342	063522	000	000			.BYTE 0,0
12343	063524	062166				.WORD 04704
12344	063526	004	000			.BYTE 4,0
12345						
12346	063530	000005			DF31:	.WORD 5
12347	063532	000	000			.BYTE 0,0
12348	063534	061156				.WORD 04101
12349	063536	007	000			.BYTE 7,0
12350	063540	061245				.WORD 04102
12351	063542	002	000			.BYTE 2,0
12352	063544	062222				.WORD 04706
12353	063546	000	000			.BYTE 0,0
12354	063550	062265				.WORD 04710
12355	063552	003	000			.BYTE 3,0
12356						
12357	063554				RWBUF:	;READ/WRITE DATA BUF <AT LEAST 1536(DEC) WORDS>
12358						
12359	063554	005015	020040	020040	NOTMSG:	.ASCII <15><12>/ *** NOTE ***/<15><12><12>
12360	063562	020040	020040	020040		
12361	063570	025052	020052	047516		
12362	063576	042524	025040	025052		
12363	063604	005015	012			
12364	063607	101	046114	042040		.ASCII /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12365	063614	044522	042526	020123		
12366	063622	047524	041040	020105		
12367	063630	042524	052123	042105		
12368	063636	046440	051525	020124		
12369	063644	040510	042526	035040		
12370	063652	005015	012			
12371	063655	061	020056	020101		.ASCII /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12372	063662	030062	047440	020122		
12373	063670	031062	051440	041505		

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 235
TRAP TABLE

SEQ 0513
SEQ 0234

12374	063676	047524	020122	047506	
12375	063704	046522	052101	042524	
12376	063712	020104	040503	052122	
12377	063720	044522	043504	006505	
12378	063726	012			
12379	063727	062	020056	042510	.ASCII /2. HEADS MANUALLY LOADED/<15><12>
12380	063734	042101	020123	040515	
12381	063742	052516	046101	054514	
12382	063750	046040	040517	042504	
12383	063756	006504	012		
12384	063761	063	020056	042504	.ASCII /3. DESIRED PORT SELECTED/<15><12>
12385	063766	044523	042522	020104	
12386	063774	047520	052122	051440	
12387	064002	046105	041505	042524	
12388	064010	006504	012		
12389	064013	064	020056	051127	.ASCII /4. WRITE LOCK DISABLED/<15><12>
12390	064020	052111	020105	047514	
12391	064026	045503	042040	051511	
12392	064034	041101	042514	006504	
12393	064042	012			
12394	064043	065	020056	051104	.ASCII /5. DRIVE READY LIGHT ON/<15><12><12>
12395	064050	053111	020105	042522	
12396	064056	042101	020131	044514	
12397	064064	044107	020124	047117	
12398	064072	005015	012		
12399	064075	104	044522	042526	.ASCII /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12400	064102	020123	047516	020124	
12401	064110	047524	041040	020105	
12402	064116	042524	052123	042105	
12403	064124	046440	051525	020124	
12404	064132	040510	042526	041040	
12405	064140	052117	006510	012	
12406	064145	120	051117	051524	.ASCII /PORTS DE-SELECTED./<15><12><12>
12407	064152	042040	026505	042523	
12408	064160	042514	052103	042105	
12409	064166	006456	005012		
12410	064172	046120	040505	042523	.ASCII &PLEASE NOTE THAT THE DEFAULT VERSION OF READ/WRITE&<15><12>
12411	064200	047040	052117	020105	
12412	064206	044124	052101	052040	
12413	064214	042510	042040	043105	
12414	064222	052501	052114	053040	
12415	064230	051105	044523	047117	
12416	064236	047440	020106	042522	
12417	064244	042101	053457	044522	
12418	064252	042524	005015		
12419	064256	040504	040524	052040	.ASCII /DATA TEST 21 TAKES ABOUT 10 MIN, (20 FOR RK07),/<15><12>
12420	064264	051505	020124	030462	
12421	064272	052040	045501	051505	
12422	064300	040440	047502	052125	
12423	064306	030440	020060	044515	
12424	064314	026116	024040	030062	
12425	064322	043040	051117	051040	
12426	064330	030113	024467	006454	
12427	064336	012			
12428	064337	122	043505	051101	.ASCII /REGARDLESS OF THE PASS NUMBER./<15><12><12>
12429	064344	046104	051505	020123	

C03

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 308(1052) 10-JAN-78 12:16 PAGE 236
TRAP TABLE

SEQ 0514
SEQ 0235

12430	064352	043117	052040	042510
12431	064360	050040	051501	020123
12432	064366	052516	041115	051105
12433	064374	006456	005012	
12434	064400	047506	020122	030065
12435	064406	044040	020132	050103
12436	064414	020125	050117	051105
12437	064422	052101	047511	026116
12438	064430	050040	052101	044103
12439	064436	046040	041517	020056
12440	064444	033061	020066	020075
12441	064452	027061	005015	000012
12442				
12443				
12444				
12445	000001			

.ASCIZ /FOR 50 HZ CPU OPERATION, PATCH LOC. 166 = 1./<15><12><12>

.END

E03

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 239
CROSS REFERENCE TABLE -- USER SYMBOLSSEQ 0516
SEQ 0237

APTCU= 000040	10841	11501*															
APTENV= 000001	6420	10834	11280	11457	11499*												
APTPAS 025570	6415	6420*															
APTSI2= 000200	4478	11498*															
APTSP0= 000100	10836	11459	11500*														
ASKMDE 015344	4900*	4905	4906	4911	4926	4931	4936	4953	4975	4999	5013	5024	5033				
	5049	5068	5077	5103	5126	5198	7351										
	4782*	4785	4786	4791	4815	4823	4829	4845	4856	4865	4890						
ASKTMD 014600	4698	4781*															
ASKTST 014574	2104	2117															
ASWREG= 000000	8990	11669*															
AS2SP2 056354	2104	2108															
ATESTN= 000000	2104	2111															
AUNIT = 000000	2104	2118															
AUSWR = 000000	2104	2104	2143	4554													
AVECT1= 120210	2011*	2104															
AVECT2= 000000	2104	2144															
AVGEQ 011774	4345*	7987															
AVGSEK 011677	4322*	5978															
A.ABNL 00307	3340*	6882*	6969*	6973*	8438*	8527*	8544*	8553*	9204*	9460*	9473*	9475*	10402				
A.CONT 00307	3341*	10412															
A.NORM 003044	3339*	6883*	9205*	9461*	10406												
BADDRV 007650	4132*	6976*	6977*	6978	6990*	6991*	6992	6998*	6999*	7000	7025*	7026*	7027				
BA0SEC= 001000	4073*	8245	8816	8824	9406	9509											
BA0632 012362	4392*	7022															
BAI = 000020	3018*	10571															
BA16 = 000400	2994*																
BA17 = 001000	2995*																
BDSCHD 037140	8527	8544	8558*														
BDSRCT 011450	4300*	8576															
BDSRCK 040212	8243	8806*	9405														
BELOW 012007	4348*	5830															
BGNRTY 044152	9128	9471	9477	9502*													
BIT0 = 000001	1903*	2991	3008	3034	3056	3204	3348	3441	4639	4737	6151	7390	8119				
	8136	8151	8190	8207	8261	8303	8860	8865	9087	10686							
BIT00 = 000001	1893*	1903															
BIT01 = 000002	1892*	1902	5710														
BIT02 = 000004	1891*	1901															
BIT03 = 000010	1890*	1900	5823	5926	6080												
BIT04 = 000020	1889*	1899															
BIT05 = 000040	1888*	1898															
BIT06 = 000100	1887*	1897															
BIT07 = 000200	1886*	1896	5455	7518	7705												
BIT08 = 000400	1885*	1895	7546														
BIT09 = 001000	1884*	1894	5696	11290	11421												
BIT1 = 000002	1902*	3009	3036	3205	3349	3442	4065	6319	9091								
BIT10 = 002000	1883*	2996	3025	3047	3082	3096	3109	3124	3138	3217	3358	4074	9120				
	11268																
BIT11 = 004000	1882*	2997	3026	3048	3067	3083	3097	3110	3125	3139	3218	4075	11428				
BIT12 = 010000	1881*	2999	3027	3049	3084	3098	3111	3112	3126	3140	3219	4076	8863				
	9002	9030															
BIT13 = 020000	1880*	3000	3028	3050	3068	3085	3099	3113	3127	3141	3220	6862	6864				
	8142	8145	8216	8219	11275												
BIT14 = 040000	1879*	3001	3029	3051	3069	3086	3100	3114	3128	3142	3221	3359	11403				
BIT15 = 100000	1878*	3002	3003	3030	3052	3070	3087	3147	3222	3360	4077	5010	7281				
	9123																
BIT2 = 000004	1901*	3010	3037	3038	3058	3206	3350	3443	4066	6325	9095						

SEQ 0517
SEQ 0238

BIT3	=	000010	1900#	3016	3017	3039	3059	3207	3351	3444	4067	6321	9102		
BIT4	=	000020	1899#	3011	3018	3040	3060	3061	3076	3118	3132	3208	3352	3445	4068
			5308#	6313	9106										
BIT5	=	000040	1898#	3019	3020	3041	3062	3077	3091	3104	3119	3133	3209	3353	4069
			6262#	6309	6462	8967	9110	10483							
BIT6	=	000100	1897#	2992	3021	3042	3063	3078	3092	3105	3120	3134	3210	3355	4070
			4822#	5012	5032	6159	6391	6467	6500	8288					
BIT7	=	000200	1896#	2993	3022	3043	3064	3065	3079	3093	3106	3121	3135	3212	4071
			4552#	4734	5284	5287	6980	8861	8888	9606	9609				
BIT8	=	000400	1895#	2994	3023	3044	3045	3066	3080	3094	3107	3122	3136	3213	3356
			4072#	9114											
BIT9	=	001000	1894#	2995	3024	3046	3081	3095	3108	3123	3137	3214	3357	4073	5293
			6646#	9117	11380										
BKSLSH		012463	4404#	7147	7155										
BLDEXH		045012	9357#	9382	9653#										
BLWMIN		003170	3457#	5803*	5828	7720*									
BPTVEC	=	000014	1910#												
BSE	=	000200	3043#	8558	9397	9836	10002	10238							
BSERR	=	000002	4065#	6141	8738	9399	9409	9502							
BSFACT		004232	3475#	8526	8574	8807									
BSSOFT		003232	3474#	8538	8607	8812	8814								
BUFF0		005272	3484#	4598	4629	4632	4681	4702	4706	4708	4713	4724	4728	4789	4792
			4795#	4798	4833	4835	4866	4869	4871*	4876	4894	4909	4912	4915	4918
			4921#	4957	4970	4973	5053	5055	5087	5104	5130	5137	5147	5149	5152
			5157#	5166	5204	5207	6591	6595	7150	7164*	7169*	7170	7174*	7291	7293
			7313#	7316	7318*	7323	7339								
BUFPRT		005502	3487#												
B.BA16	=	000001	3008#	10565											
B.BA17	=	000002	3009#	10565											
B.CDT	=	000004	3010#	5256	6966	10219	10565								
B.CFMT	=	000020	3011#	5295	5741	5749	5769	7017	7485	8516	9611	10219	10565	10649	
CALBRT		032712	5734#	5875	5957	6029	7632#								
CART		011401	4292#	7495											
CCLR	=	100000	3003#	10172											
CDT	=	002000	2996#	6951	9978	10528	10585	10636	10659						
CERR	=	100000	3007#	9366	9391	9874	9933	9992	10174	10225	10540	10596	10684		
CFMT	=	010000	2994#	5750	5770	9632	9665	9978	10528	10585	10636	10659			
CHGTST		015016	4797#	4826#	4830	4837									
CHKITR		030530	5342#	5366	5392	5440	5479	5527	5583	5619	5643	5676	5726	5867	5949
			6021#	6098	6181	6256	7063#								
CHKLIM		035762	6274#	6301	8322#										
CHKLST		014522	4605#	4687	4730	4747#									
CHKPRM		031630	5093#	5172	7372#										
CHKXDP		027000	4955#	6737#											
CLEAR	=	000105	2962#												
CLKFAL		011556	4316#	7644											
CLKOF		032670	7609#	7686											
CLKON		032614	7588#	7639											
CLOCK		032664	4525#	7603#											
CLR	=	000040	3019#												
CLAPRM		027622	6887#	6905#											
CMDTO	=	000100	3210#	9251	9566	9735									
CMPBUF		035054	6327#	8177#											
CMPTIM		033624	5892#	5912	6047	5065	7840#								
CMSTRT		012606	4429#	4433	4438#	9493									
CNTRDY		010071	4160#	4683	8451										

SEQ 0518
SEQ 0239

CNTRLG	012426	4398#	6466	7118															
CNTRLG	012452	4402#	6480	7136															
CNTRLR	012440	4400#	6476																
CNTRLU	012445	4401#	7130																
CNTRLZ	012433	4399#	6472	7124															
CNVTIM	034136	5776	7763	7940#															
COE =	001000	3046#	9323	9836	10002	10238													
COMMA	012465	4405#	4644																
COMSTR	005246	3478#	8908	8923	9521														
CONCLR=	000176	2980#	8877	10667															
CONERR	041460	3341	9083#																
CR =	000015	1818#	4081	4084	4367	4377	10880	10890											
CRLF =	000200	1819#	10851	10890															
CRMPHO	005270	3483#	8251	8918	8920*														
CRMPLD	005266	3482#	8252	8919	8921*														
CRTSER	032254	5307	7484#																
CR2LF	012457	4403#	8989	9001	9025														
CS	005770	3607#	5308	5823	5926	6080	8967												
CTLOUT	036332	5781	5889	5909	6044	6062	8422#	8852											
CTO =	004000	2997#	9286	10234															
CYLNR	005514	3492#	5452*	5455	5458	5460*	5625	5681*	5688	5695*	5696	5699*	5706	5710					
		7574#	7575*	7576	7578*														
C.INIT	050570	7759	8893	10445#															
C.RTRN	052442	10549	10577	10605	10612	10615	10622	10626	10643	10665	10670	10680	10689	10702					
		10706	10711#																
C.SPEC	051462	10485	10581#																
DASTRT	012556	1975	4431#																
DCK =	100000	3052#	8558	9411	9836	10002	10238												
DCKERR=	000020	4068#	9413	9502															
DCLO =	000020	3061#																	
DDISP =	177570	1825#	2042	4466															
DDT =	000400	3066#																	
DERCNT	003121	3417#	9350	9375	9420	9431													
DESL =	000010	3016#																	
DFLTST	005670	3567#	3587	4820	4839	4842	4863	4888											
DFQUES	011034	4249#	5046																
DFSTRT	012522	1969	4423#																
DFTEST	010374	4195#	4826																
DF01	062620	2169	2175	2181	2187	2229	2337	2607	2643	12094#									
DF02	062650	2193	2211	2217	2223	2235	2241	2247	2253	2259	2265	2271	2277	2283					
		2289	2295	2301	2355	2361	2367	2475	2481	2487	2493	2499	2505	2511					
		2517	2523	2529	2571	2589	2613	12108#											
		2199	2205	2577	2583	12123#													
DF03	062704	2331	12140#																
DF04	062744	2307	2313	12147#															
DF05	062760	2319	12162#																
DF07	063014	2325	12177#																
DF10	063050	2625	12188#																
DF11	063074	2373	2379	2385	2391	2397	2403	2409	2415	12197#									
DF12	063114	2421	2433	12210#															
DF13	063144	2427	12223#																
DF14	063174	2439	12236#																
DF15	063224	2445	12255#																
DF16	063270	2463	12274#																
DF17	063334	2631	12285#																
DF20	063360	2541	12290#																
DF21	063370																		

SEQ 0519
SEQ 0240

[illegible]

J03

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 244
CROSS REFERENCE TABLE -- USER SYMBOLSSEQ 0521
SEQ 0242

DZR6M	007226	4079#	4488			
ECCNC =	000040	4069#	9416			
ECCW =	020000	3085#				
ECH =	000100	3042#	9414	9836	10002	10238
EC08AD	026060	4816	5000	5025	6514#	
EMTVEC=	070030	1913#	4449*	4450*		
EM1	056361	2166	11670#			
EM10	056530	2208	11691#			
EM100	060264	2556	11856#			
EM101	060307	2562	11860#			
EM102	060365	2568	11868#			
EM103	060431	2574	11875#			
EM104	060457	2580	11879#			
EM105	060501	2586	11883#			
EM106	060532	2592	11888#			
EM107	060565	2598	11893#			
EM11	056537	2214	11693#			
EM110	060630	2604	11899#			
EM111	060663	2610	11904#			
EM112	060704	2616	11907#			
EM113	060726	2640	11910#			
EM114	060754	2646	11914#			
EM115	060771	2652	11917#			
EM12	056552	2220	11695#			
EM13	056564	2226	11697#			
EM14	056574	2232	11699#			
EM15	056614	2238	11702#			
EM16	056630	2244	11704#			
EM17	056640	2250	11706#			
EM2	056400	2172	11673#			
EM20	056655	2256	11709#			
EM21	056670	2262	11711#			
EM22	056701	2268	11713#			
EM23	056714	2274	11715#			
EM24	056731	2280	11718#			
EM25	056745	2286	11721#			
EM26	056764	2292	11724#			
EM27	056776	2298	11726#			
EM3	056416	2178	2658	11676#		
EM30	057013	2304	2424	11729#		
EM31	057025	2310	2430	11731#		
EM32	057041	2316	11734#			
EM33	057056	2322	11737#			
EM34	057072	2328	2628	11739#		
EM35	057112	2334	11742#			
EM36	057144	2340	11747#			
EM37	057173	2346	11751#			
EM4	056434	2184	11679#			
EM40	057234	2352	11757#			
EM41	057300	2358	11763#			
EM42	057330	2364	11767#			
EM43	057360	2370	11771#			
EM5	056453	2190	11682#			
EM52	057411	2412	11776#			
EM53	057426	2418	11779#			
EM56	057443	2436	2442	11782#		

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 245
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0522
SEQ 0243

[illegible]

SEQ 0523
SEQ 0244

[illegible]

SEQ 0524
SEQ 0245

[illegible]

N03

CZR6M00 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 248
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0525
SEQ 0246

LODSEC	034304	6120	6131	6195	6210	8001*	8023								
LOERAD=	177740	1956*													
LOWOCT	005476	3485*	5090*	5170*	7220*	7225	7382	7384	7387	7390	7395	7400	7403		
LS	005520	3494*	5291*	5297*	6350										
LSTCYL	017760	5238*	5261*	5319*	5900	5923	5976	6038	7490	7576	8328	8357	8380	8518	
LSTMEM	007413	4101*	6560												
LSTTRK=	000002	4063*	7014	7491											
LSTTST	014700	4794	4803*												
LT	005742	3596*	4932	6355											
MA	005762	3604*	4944	4946	6278	6279	6282	6283	6746	6757	6759	6764	6772	7427	
		7429	8502*	8503*	8504*	8711	8712								
MAHILM	006126	3663*	4947	4950	6532*	6533*	6547*	6548*	6561	6713	6717	6719	6784	6787	
		7425	7426												
MAPH00=	170202	1954*	8920												
MAPL00=	170200	1953*	6695	8921											
MAWRDC	010710	4232*	7407												
MAXEQ	011765	4343*	7974												
MAXHI1=	000000	4050*	5804												
MAXHI2	007220	4054*	5251*	5275*	5894	5914									
MAXHI3=	000000	4056*													
MAXHI4	007224	4060*	5253*	5277*	6049	6067									
MAXIH1	003220	3469*	5790	5795*	5991	7730*	7828*								
MAXIH2	003230	3473*	5993	7734*	7825*										
MAXIL1	003216	3468*	5793	5796*	5832	5990	7729*	7900							
MAXIL2	003226	3472*	5992	7733*	7827*	7915									
MAXL01=	062031	4049*	5807												
MAXL02	007216	4053*	5250*	5274*	5897	5917									
MAXL03=	112160	4055*													
MAXL04	007222	4059*	5252*	5276*	6052	6070									
MAXSEK	011675	4331*	6082												
MCLK =	000400	3080*													
MDFLAG	003116	3414*	4423*	4432*	4435*	4548	4666	4844	5064	6941	7067	8426			
MDFY15	011167	4266*	7284												
MDS =	001000	3024*	6955	9800	10227	10602									
MEMABT	003145	3437*													
MEMCSR=	172100	1955*													
MEMSYS=	177744	1958*													
MEMVEC=	000114	3454*													
MERD =	001000	3081*													
MESMSK=	000017	3074*													
MEWD =	002000	3082*													
MICROS	012003	4347*	7965	7978											

[illegible]

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 250
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0527
SEQ 0248

```
OREQSW= 000020
OVLORD 007443
O.WAIT 003054
PACK = 000103
PARMDE 010626
PARMO 002630
PARM1 002714
PART = 000000
```

[illegible]

PART1		007403
PART2		007407
PAT	=	000020
PATRN		005544
PAT00		006216
PAT01		006256
PAT02		006316
PAT03		006356
PAT04		006416
PAT05		006456
PAT06		006516
PAT07		006556
PAT08		006616
PAT09		006656
PAT10		006716
PAT11		006756
PAT12		007016
PAT13		007056
PAT14		007116
PAT15		007156
PBLKT		003112
PBSVAL	=	001000
PCA	=	004000
PCD	=	010000
PCLKF		003132
PCVEC		003162
PFIFTN		011116
PGE	=	002000
PIP	=	020000
PIRQ	=	177772
PIRQVE	=	000240
PKRB		003156
PKS		003152
PKSB		003154
PLOFST		005554
PMA		005602
PROFLT		005776
PREPAR		026604
PREPKB		026040
PRMBUF		012476
PRMINP		007500

[illegible]

D04

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 251
CROSS REFERENCE TABLE -- USER SYMBOLSSEQ 0528
SEQ 0249

PRMLIM	006042	3636*	5236*	5240*	5241*	5259*	5263*	5264*	7382	7384	7392	7395	7397	7400
PRMLST	005732	3591*	5007	5060	5100	5141	7220	7224	7387*	7403*	7404*			
PRMNEH	006152	3677*	5002	5081	5137	7218	7222	7278	7375					
PRMPHO	005264	3481*	8918*	12092										
PRMPLO	005262	3480*	8919*	12092										
PRMPSP	012517	4416*	5122											
PROMPT	012514	4415*	4688	4852	4902	5073	7301							
PRVCMO	005232	3476*	4507	8876*	8878*	8907	9170							
PRO	= 000000	1840*	4506	5218	7048	9529	9601							
PR1	= 000040	1841*												
PR2	= 000100	1842*												
PR3	= 000140	1843*												
PR4	= 000200	1844*	4499											
PR5	= 000240	1845*	3338											
PR6	= 000300	1846*												
PR7	= 000340	1847*	4438	4504	4526	6660	11360	11361						
PS	= 177776	1820*	1821	4438*	4506*	5218*	7048*	9722	9728*	9742*	10451	10452*	10711*	
PSTART	012574	1973	4435*											
PSW	= 177776	1821*	9529*	9601*										
PT	005760	3603*	5010	6265	6266	6292	6333	6744	7281					
PWRMSG	055106	11363	11367*											
PWRVEC	= 000024	1912*	4453*	4454*	11351*	11359*	11360*							
P.ASOF	= 000032	3235*	9157	10064*	10065	10098*	10329*	10517*	10518					
P.A00	= 000040	3238*	6988	6996	9160	9569	10055*	10056	10072*	10073	10386*	10594*		
P.A01	= 000044	3240*	9162	10365*										
P.A10	= 000050	3242*	9164	10372*										
P.A11	= 000054	3244*	9166	10379*										
P.BAHI	= 000007	3197*	6281*	6703*	8689*	9146								
P.BALO	= 000010	3198*	6110*	6190*	6269*	6278*	6702*	7015*	7492*	8476*	8526*	8538*	8686*	9149
		10551	10648											
P.BAR	= 000024	3232*	9156	10095*	10325*									
P.B00	= 000042	3239*	9161	10050*	10051	10387*	10595*							
P.B01	= 000046	3241*	9163	10366*										
P.B10	= 000052	3243*	9165	9591	10373*									
P.B11	= 000056	3245*	9167	9971*	10380*									
P.CMND	= 000001	3191*	5281*	5348*	5742*	5879*	5961*	6033*	6103*	6132*	6135*	6186*	6211*	6311*
		6315*	6323*	6330*	6888*	6957*	6968*	7004*	7006*	7008*	7046*	7451*	7489*	7517*
		7520*	7754*	7771*	8024*	8440*	8479*	8482*	8517*	8877*	9137	9472*	9519*	9564*
		9600*	9603*	9866	10483	10487	10499	10509	10515	10521	10524	10530	10555	10562
		10581	10628	10645	10667	10676								
P.CS1	= 000016	3229*	5750*	5770*	5771	6908	6958*	8879*	9150	9268	9286	9632	9665	9887*
		9939*	9977*	9978*	9980*	9981	10090*	10092*	10319*	10322*	10473	10527*	10528*	10530*
		10534*	10535	10559*	10562*	10564*	10565*	10575*	10576	10584*	10585*	10587*	10588	10590*
		10591	10596	10635*	10636*	10638*	10641*	10642	10658*	10659*	10661*	10664	10683*	10684
		10693*	10694											
P.CS1H	= 000007	3196*	5741*	5749*	5769*	6892*	6893*	7017*	7018*	7485*	8516*	9611*	9612*	9977
		10218	10527	10564	10584	10635	10649	10658						
P.CS2	= 000020	3230*	6433	9151	9225	9230	9235	9237	9243	9245	9293	9343	9427	9893
		10093*	10323*	10481*	10506	10571*	10572	10583	10602	10606	10632*	10633	10657	
P.CYLN	= 000002	3192*	5351*	5372*	5375*	5398*	5401*	5403*	5404	5407*	5409	5446*	5449*	5453*
		5457*	5458*	5462*	5486*	5507*	5534*	5536*	5557*	5559*	5590*	5592*	5625*	5648*
		5650*	5652*	5686*	5688*	5704*	5706*	5880*	5883*	5900	5903*	5920	5962*	5968*
		5972*	6034*	6038*	6056*	6104*	6187*	6270*	6290*	6344*	6345	6360*	6361	6364*
		6365	7011*	7013*	7490*	7523	7526	8091	8323	8328	8357	8369	8380	8518*
		8620	8674*	9139	9599*	9975	10511	10553	10655					
P.DCYL	= 000030	3234*	8624	8670	9152	9400	9622	9654	10097*	10331*				

E04

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 252
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0529
SEQ 0250

P.DRVN= 000000	3190*	6890*	6936*	9863	10040	10216	10455							
P.OS = 000036	3237*	9158	9278	9282	9333	10100*	10327*	10608						
P.DTS = 000026	3233*	8626	8628	8671	8672	9153	9401	9403	9590	9625	9626	9655	9661	
	10096*	10326*												
P.EPAT= 000062	3247*	9424	9889*	10475										
P.EPOS= 000060	3246*	9423	9890*											
P.ER = 000034	3236*	8558	9159	9224	10059*	10328*								
P.OFST= 000006	3195*	10517												
P.PRST= 000014	3200*	9251	9264	9436	9441	9446	9451	9566	9735*	9848*	9882*	9910*	9917	
	9921*	10018*	10041	10045*	10058*	10076*	10086*	10112	10116*	10126*	10138*	10243*	10395*	
	10470*	10507*	10531*	10532	10557	10563*	10569	10573	10600	10610*	10618	10639	10662	
	10671	10705												
P.SECT= 000004	3193*	6109*	6118	6122*	6123	6127*	6129	6140	6164*	6165	6189*	6273*	6298*	
	6336*	6337	6349*	6350	7488*	8018	8029*	8068	8347	8523*	8532*	8533	8539*	
	8549*	8550	8622	8676*	9142	9598*	9976	10512	10554	10656				
P.TRCK= 000005	3194*	6105*	6188*	6193	6197*	6198	6207*	6208	6214*	6215	6271*	6297*	6340*	
	6341	6352	6354*	7014*	7491*	8019*	8021	8026*	8027	8082	8330	8360	8519*	
	8621	8675*	9140											
P.WC = 000012	3199*	6111*	6191*	6272*	6299*	6307*	7016*	7493*	8388	8401*	8477*	8494	8525*	
	8679*	8705	8742	9144	10552									
	3231*	9154	10094*	10324*										
P.WCR = 000022	2950													
QNEWSW= 000000	3221*													
Q.INIT= 040000	3444*													
RCDASW= 000010	4075*	9271	9276	9321	9326	9331	9336	9470						
RCLREQ= 004000	2979*	9866	10645											
RDALHO= 000164	7113	11662*												
RDCHR = 104406	4623	4675	4690	4783	4827	4854	4903	4961	5047	5075	5124	5196	6582	
RDCHRS 030624	7107*	7285	7303											
	2968*	6135	6323	7008	7489	8517								
RDDATA= 000121	3087*													
RDGATE= 100000	9358	9383	9588*											
RDHDO 044522	2970*	5281	5742	7520	9603	9980	10661							
RDHEAD= 000125	2978*	6968	7451	9564	10581									
RDSTAT= 000141	2993*	9466	10538	10591										
RDY = 000200	2965*	5348	7006	8440	9472	10524								
RECAL = 000113	3488*	6141	6962*	7020	7043*	8245	8735	8738	8816*	8824*	8875*	8943	8953*	
RECODE 005504	9200*	9201	9228*	9233*	9254	9256	9271*	9276*	9316*	9321*	9326*	9331*	9336*	
	9341*	9349*	9353	9359	9374*	9378	9384	9399*	9406	9409*	9413*	9416*	9429*	
	9430*	9439*	9444*	9449*	9454*	9462	94 9*	9470	9476	9478*	9502	9504*	9509	
	9516*	9536*	9568*											
REDBSF 036674	5300	8514*												
REISSU 003140	3432*	6308*	9099*											
RELEAS= 000140	2977*	6330	10628											
REPLPK 007753	4146*	4673												
REPSUP 041704	5757	6655	6960	7525	7776	8182	8535	8881	9134*	9206				
RESREG= 104410	5761	5810	6566	6647	6704	6749	6830	7416	7474	7687	7809	7829	7879	
	8006	8030	8047	8094	8163	8244	8305	8403	8407	8484	8505	8610	8635	
	8693	8721	8817	8898	8930	9070	9185	9531	9580	9614	9673	11008	11060	
	11664*													
RESVEC= 000010	1907*													
RETANL 044324	9204	9473	9535*											
RETNML 044342	6448	8562	9205	9538*										
REVRSE 011740	4338*	6003	7911											
RHEDHO 022422	5747	5847*												
RKASOF= 000016	2946*	8884	9557	9876	9946	10026	10281	10329	10518*	10543				

F04

CZR6MDD RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 253
CROSS REFERENCE TABLE -- USER SYMBOLSSEQ 0530
SEQ 0251

RKBA = 000004	2941#	9556	10277	10325	10551*									
RKBADR 007526	4115#	4570												
RKBAS 003036	3336#	4551*	4555*	4569	4572*	6884	7774	9361	9386	9465	9518	9549	9729	
	9798	10462												
RKCS1 = 000000	2939#	4684*	5771*	5848	6984*	7757	7775*	9362*	9366	9387*	9391	9551	9821	
	9873	9931	9981*	9990	10172*	10173	10182*	10221*	10222	10246*	10274	10535*	10537	
	10576*	10588*	10590	10620*	10642*	10664*	10673*	10683	10693	10708*				
RKCS2 = 000010	2943#	6946*	6955	8859*	8863	8886*	9552	9799	9959	10217*	10275	10323	10506*	
	10572*	10583*	10633*	10657*	10682*									
RKDA = 000006	2942#	9554	9976*	10278	10326	10512*	10554*	10656*						
RKDB = 000024	2943#	5292	5293	7522	9363	9364	9365	9388	9389	9390	9955	9956	9957	
RKDC = 000020	2947#	9553												
RKDCYL = 000020	2948#	9975*	10282	10331	10511*	10553*	10655*							
RKDS = 000012	2944#	8865	9466	9558	9824	10279	10327							
RKECPS = 000030	2954#	9890	10286											
RKECPT = 000032	2956#	9889	10287											
RKER = 000014	2945#	6964	9559	9833	9999	10280	10328							
RKMR1 = 000026	2950#	9967*	10046*	10283	10360*	10367*	10374*	10381*	10482*					
RKMR2 = 000034	2951#	10284	10594											
RKMR3 = 000036	2952#	10285	10595											
RKPAT = 000032	2955#													
RKPOS = 000030	2953#													
RKPRI 003042	3338#	4502	4560*	4579	4595*	9728	10452							
RKPRTY 007574	4123#	4584												
RKVADR 007551	4119#	4575												
RKVEC 003040	3337#	4500	4556*	4557*	4574	4577*	5747*	5841*	7753*	7768*				
RKWC = 000002	2940#	9555	10276	10324	10552*									
RLS = 000010	3017#	10632												
RNDADR 032554	5624	7573#												
RNDLNG = 016514	4062#													
RNDSH = 002734	4061#													
ROTMS 011577	4319#	5825												
RUNTST 016720	4957	5193#												
RVTERM 033560	5975	7818#												
RWBUR 063554	3604	3626	3661	6110	6137	6190	6269	7015	7023	7492	7497	8002	8004	
	8116	8187	8470	8476	8707	12357#								
RWDST 024516	5315	6220#												
RWTINX = 000040	6221#	6742												
R.ABNL 050522	9741	9851	9926	9943	10121	10141	10401#	10611						
R.CONT 050546	9803	9841	9905	9914	9985	10007	10030	10036	10069	10109	10133	10178	10230	
	10253	10391	10409#	10625	10679	10688	10701							
R.NORM 050534	9879	9972	10081	10405#	10548	10621	10709							
SAVPAR 003206	3464#	6691*	8135	8206										
SAVPRS 005560	3509#	8704	8715											
SAVREG = 104407	5775	6527	6633	6677	6737	6809	6812	7372	7449	7632	7798	7818	7862	
	8001	8017	8039	8062	8110	8177	8239	8325	8468	8492	8572	8619	8668	
	8702	8714	8806	8850	8906	8965	9083	9135	9197	9548	9589	9653	10993	
	11034	11663#												
SAVWRD 003210	3465#													
SAVXDP 027254	6776	6809#												
SCLR = 000040	3020#	10682												
SCNDRV 027650	4751	6934#												
SCOPE = 104411	6332	11665#												
SCOPE1 055130	11378#	11665												
SCRACH 005542	3503#	4645*	4646*	4647	5684*	5690*	5702*	5708*	5767*	5812	5965*	5967*	6036*	
	6073#	6074	6139*	6145	6150*	6153	6952*	6953*	7148*	7150*	7151	7161*	7162*	

CZR6MDO RK611/06 SS VERIF 1 MACY11 30A(1052) 10-JAN-78 12:16 PAGE 254
CZR6MD.P11 10-JAN-78 11:48 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0531
SEQ 0252

[illegible]

H04

CZR6MDO RK611.06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48MACY11 30A(1052) 10-JAN-78 12:16 PAGE 255
CROSS REFERENCE TABLE -- USER SYMBOLSSEQ 0532
SEQ 0253

SWO	=	000001	1875#				
SW00	=	000001	1865#	1875			
SW01	=	000002	1864#	1874			
SW02	=	000004	1863#	1873			
SW03	=	000010	1862#	1872			
SW04	=	000020	1861#	1871			
SW05	=	000040	1860#	1870			
SW06	=	000100	1859#	1869			
SW07	=	000200	1858#	1868			
SW08	=	000400	1857#	1867			
SW09	=	001000	1856#	1866			
SW1	=	000002	1874#				
SW10	=	002000	1855#				
SW11	=	004000	1854#				
SW12	=	010000	1853#				
SW13	=	020000	1852#	8975			
SW14	=	040000	1851#				
SW15	=	100000	1850#				
SW2	=	000004	1873#				
SW3	=	000010	1872#				
SW4	=	000020	1871#				
SW5	=	000040	1870#				
SW6	=	000100	1869#				
SW7	=	000200	1868#				
SW8	=	000400	1867#				
SW9	=	001000	1866#				
S.ACLO	=	000100	3105#				
S.BRHM	=	000100	3120#				
S.BRKE	=	040000	3142#				
S.CART	=	000400	3122#				
S.DCLO	=	010000	3112#				
S.DIB	=	002000	3138#				
S.DOOR	=	000200	3121#				
S.DRA	=	000040	3091#				
S.DROT	=	020000	3113#				
S.DRY	=	000200	3093#	6988			
S.DSC	=	040000	3100#	9908	10056	10073	10136
S.FLT	=	000200	3106#	10051			
S.FORM	=	001000	3095#				
S.FWD	=	002000	3124#				
S.HOFL	=	000200	3135#				
S.HOHM	=	000040	3119#				
S.ICYL	=	000040	3104#				
S.ILF	=	000400	3107#				
S.LIMD	=	020000	3141#				
S.LOAD	=	010000	3126#				
S.MHO	=	000400	3136#				
S.NMOV	=	010000	3140#				
S.OFF	=	002000	3096#				
S.PAR	=	001000	3108#	9911	10388		
S.PIP	=	020000	3099#	9919	10114		
S.PLO	=	004000	3139#				
S.REV	=	004000	3125#				
S.RTZ	=	020000	3127#				
S.SECT	=	000020	3132#				
S.SKI	=	002000	3109#				

[illegible]

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 257
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0534
SEQ 0255

```
TWOTOS= 002000
TYPAVG  034260
TYPBSF  037164
TYPDS   = 104405
TYPE    = 104401
```

4074#	9254	9568										
5839	6000	6005	7908	7923	7987#							
5310	8572#											
5829	5835	6404	7903	7918	11659#							
4488	4489	4490	4493	4497	4535	4545	4566	4570	4575	4584	4596	4598
4599	4622	4632	4633	4636	4644	4647	4655	4656	4665	4673	4683	4688
4702	4703	4781	4782	4769	4790	4803	4807	4826	4835	4836	4848	4851
4852	4894	4895	4899	4901	4902	4909	4910	4925	4929	4930	4934	4935
4952	4960	4967	4968	4991	5015	5028	5046	5055	5056	5072	5073	5104
5105	5122	5130	5131	5195	5204	5205	5303	5825	5830	5831	5836	5837
5840	5928	5978	5998	6001	6002	6006	6082	6402	6405	6466	6472	6476
6480	6482	6483	6514	6515	656	6564	6565	6581	6591	6592	6616	6943
6978	6979	6992	6993	7000	7001	7022	7027	7028	7118	7124	7130	7136
7147	7151	7155	7163	7168	7170	7171	7173	7192	7199	7219	7244	7250
7284	7293	7294	7297	7300	7301	7339	7340	7407	7413	7453	7454	7473
7495	7496	7500	7644	7645	7896	7899	7905	7909	7911	7914	7920	7924
7961	7965	7974	7978	7987	7991	8450	8451	8575	8576	8577	8578	8584
8585	8589	8595	8599	8608	8973	8989	8990	8991	8995	8996	9001	9004
9005	9006	9007	9012	9015	9016	9017	9018	9019	9022	9025	9026	9028
9042	9047	9051	9057	9062	9063	9065	10854	11101	11166	11241	11270	11278
11363	11655#											
8965#	11277											
3420#	4755	5233*	5256*	6893	6934*	6947	6966*	7009	7018			
4914	4987#											
5833	7901	7916	7974#									
5827	7898	7913	7961#									
8588	8594	8598	9011	9021	9024	9044	11656#					
11658#												
6578	7196	7201	7247	7252	8998	11657#						
5027	7243#	7299										
4990	5071	7218#										
5930	6084	7894#										
4806	4850	7192#										
3323#	9876*	9877	9901	9946*	9947	10026*	10027	10033	10064	10098	10106	10130
10281*	10543*	10544										
3320#	10095	10277*										
3315#	9821*	9873*	9874	9887	9898*	9931*	9933	9939	9968*	9990*	9992	10022
10047*	10061*	10091*	10092	10103*	10127*	10173*	10174	10214	10219*	10219*	10221	10225
10234	10274*	10321*	10322	10362*	10369*	10376*	10383*	10537*	10538	10540		
3317#	9799*	9800	9815	9818	9830	9859	9959*	9960	9996	10010	10093	10215
10216*	10217	10227	10236	10275*								

```

TYPEERR      040776
TYPFMT      003124
TYPLST      015726
TYPMAX      034234
TYPMIN      034210
TYPOC      = 104402
TYPOC      = 104404
TYPOS      = 104403
TYPPAT      031252
TYPPRM      031172
TYPTMS      033770
TYPTST      031136
T.ASOF      003014

```

T.BA 003006
T.CS1 003000

T.CS2 003002

```
T.DA      003010
T.DB      003034
T.DC      003012
T.DS      003020
T.ER      003016
T.MR1     003022
T.MR2     003024
T.MR3     003026
T.PAT     003032
T.POS     003030
T.WCR     003004
UBMPRS    003144
UEXATT=   000010
UFE      = 000400
UNLOAD=   000107
```

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 258
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0535
SEQ 0256

[illegible]

SEQ 0536
SEQ 0257

SINTHL 001135
SITEMP 001114

[illegible]

SEQ 0537
SEQ 0258

SEQ 0538
SEQ 0259

\$REG10	001202	2061#	6155*	6661*	8235*	8236*	8237	8275*	8278	9363*	9388*	9423*	12081	12089
\$REG11	001204	2062#	6156*	8276*	9364*	9389*	9424*	12081	12089					
\$REG12	001206	2063#	6157*	8277*	9365*	9390*	9417*	9419*	12081	12089				
\$REG13	001210	2064#	8279*	8281*	8283*	8284*	12081	12089						
\$REG14	001212	2065#	8278*	8280*	8282*	12084	12089							
\$REG15	001214	2066#	8285*	8286*	12084	12089								
\$REG16	001216	2067#	8214*	12085	12089									
\$REG17	001220	2068#	12085											
\$REG2	001166	2055#	12077											
\$REG20	001222	2069#	12085											
\$REG21	001224	2070#	12085											
\$REG22	001226	2071#	12085											
\$REG23	001230	2072#	12085											
\$REG24	001232	2073#	12085											
\$REG25	001234	2074#	12085											
\$REG26	001236	2075#	9009	9169										
\$REG27	001240	2076#												
\$REG3	001170	2056#	12077											
\$REG30	001242	2077#												
\$REG31	001244	2078#												
\$REG32	001246	2079#												
\$REG33	001250	2080#												
\$REG34	001252	2081#												
\$REG35	001254	2082#												
\$REG36	001256	2083#	9148*	12080										
\$REG37	001260	2084#	9149*	12080										
\$REG4	001172	2057#												
\$REG5	001174	2058#	6106*	6656*	7526*	8251*	8257*	8265	8271*	8949*	8950*	9145	9513*	9514*
		9550	9622*	9642*	9654*	9668	9671	12081	12088					
\$REG6	001176	2059#	6107*	6657*	7527*	8252*	8258*	8267	8272*	9623*	9626*	9637	9639*	9641*
		9661#	9662*	9663*	9664*	9667*	9669	9670	12081	12088				
\$REG7	001200	2060#	6108*	6140*	6658*	8259*	8269	8273*	9624*	9625*	9627	9630*	9634*	9636*
		9668#	9670*	9672*	12081	12088								
\$RESRE	056224	11612#	11664											
\$RTNAD	025546	6415#												
\$R2A =	***** U	11665												
\$SAVRE	056166	11596#	11663											
\$SCOPE	055154	4447	11400#											
\$SETUP=	000037	4419#	4446	4447	4449	4451	4453	4455	4456	4457	4459	6392	11265	11290
		11297	11307	11345	11401									
\$SIZE	055660	6529	11518#											
\$SIZEEX	056122	11556	11566#											
\$STUP =	177777	4419#												
\$SUPRS	053736	6563	7227	7412	7472	7499	7964	7977	7990	11093#				
\$SVLAD	055340	11412	11435#											
\$SVPC =	000230	1982#	1937											
\$SWR =	167000	1760#	1780	1784	1785	1786	1787	1788	1789	1790	1791	2094	2095	2096
		4456	4457	4459	4460	5340	5364	5390	5438	5477	5525	5581	5617	5641
		5674	5724	5865	5947	6019	6096	6179	6254	6383	6393	6408	6414	6416
		11256	11257	11258	11259	11260	11268	11275	11287	11290	11302	11393	11394	11395
		11396	11397	11403	11415	11417	11418	11419	11426	11427	11428	11438	11441	11444
		2117#	4480											
\$SWREG	001342	11397												
\$SWRMK=	000000	2108#	5340*	5364*	5390*	5438*	5477*	5525*	5581*	5617*	5641*	5674*	5724*	5865*
\$TESTN	001324	5947*	6019*	6096*	6179*	6254*	11436*							
		2094#	4456*	5230*	6393*	7071*	7076*	8436*	11401	11426*	11431	11434*	11444	
\$TIMES	001304													

B05

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 262
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0539
SEQ 0260

\$TKB	001146	2044#	6456	6498*	11305	11322	11328									
\$TKS	001144	2043#	4822*	5012*	5032*	6391*	6467*	6500*	8858*	11305	11320	11326				
\$TMP0	001262	2085#														
\$TMP1	001264	2086#	8771*	8772*	8780											
\$TMP10	001302	2093#	8784*	8785*	8786											
\$TMP2	001266	2087#	8767*	8795												
\$TMP3	001270	2088#	8768*	8796												
\$TMP4	001272	2089#	8769*	8797												
\$TMP5	001274	2090#														
\$TMP6	001276	2091#	5682*	5685	5693*	5701*	5703									
\$TMP7	001300	2092#	5694*	5700*												
\$TN =	000022	1769#	1780	5332	5340#	5343	5356	5364#	5367	5380	5390#	5393	5414	5438#		
		5441	5467	5477#	5480	5513	5525#	5528	5565	5581#	5584	5609	5617#	5620		
		5630	5641#	5644	5657	5674#	5677	5716	5724#	5727	5736	5855	5865#	5868		
		5877	5885	5905	5934	5947#	5950	5959	5970	5974	6010	6019#	6022	6031		
		6040	6058	6088	6096#	6099	6170	6179#	6182	6221	6222	6254#	6257			
\$TNPWR	053652	11041	11042	11062#												
\$TPB	001152	2046#	10879*	10890												
\$TPFLG	001157	2050#	10828	10890												
\$TPS	001150	2045#	10877	10890												
\$TRAP	056262	4451	11632#													
\$TRAP2	056304	11643#	11654													
\$TRP =	000012	11647#	11656#	11657#	11658#	11659#	11660#	11662	11663#	11664#	11665#	11666#				
\$TRPAD	056316	11637	11654#													
\$TSTM	001004	2007#														
\$TSTNM	001102	2023#	5229*	5314*	6392*	6424*	7073	8442*	8997	11267	11302	11392	11435*	11436		
		11441	11445													
\$STYPBN=	***** U	11660														
\$STYDPS	054224	11195#	11659													
\$STYPE	052722	10828#	11476	11647	11655	10878										
\$STYPEC	053134	10858	10865	10872	10877#											
\$STYPEX	053202	10883	10885	10888#												
\$STYPOC	054022	11136#	11656													
\$STYPON	054036	11135	11138#	11658												
\$STYPOS	053776	11131#	11657													
\$UNIT	001332	2111#	5278*													
\$UNITM	001010	2009#														
\$USWR	001344	2118#														

C05

CZR6MD0 RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 308(1052) 10-JAN-78 12:16 PAGE 264
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0540
SEQ 0261

B4EOPS	6378#	6387													
B4SCOP	11387#	11401													
CKEXIT	2667#	5781	5889	5909	6044	6062	8852								
COMMEN	1918#														
ENDCOM	1918#														
ERROR	1812#	5759	6149	6158	6666	6961	6982	7528	7778	8253	8254	8256	8274	8287	8536
	8882	8951	9089	9093	9100	9104	9108	9112	9116	9119	9122	9125	9126	9227	9232
	9239	9241	9247	9249	9258	9260	9262	9266	9270	9275	9280	9284	9288	9291	9295
	9299	9303	9307	9311	9315	9320	9325	9330	9335	9340	9345	9355	9368	9370	9380
	9393	9395	9408	9425	9434	9438	9443	9448	9453	9479	9481	9515			
ESCAPE	1918#														
GETPRI	1918#	11526													
GETSWR	1918#														
MSG	5332#	5334	5356#	5358	5380#	5382	5414#	5416	5467#	5469	5513#	5515	5565#	5567	5609#
	5611	5630#	5632	5657#	5659	5716#	5718	5855#	5857	5934#	5936	6010#	6012	6088#	6090
	6170#	6172	6220#	6224											
MULT	1918#														
NEWTST	1918#	5332	5356	5380	5414	5467	5513	5565	5609	5630	5657	5716	5855	5934	6010
	6088	6170	6222												
NEXTAD	2666#	5735	5876	5885	5905	5958	5970	5974	6030	6040	6058				
NXTAD	2665#	5736	5877	5885	5905	5959	5970	5974	6031	6040	6058				
NXTST	2663#	5343	5367	5393	5441	5480	5528	5584	5620	5644	5677	5727	5868	5950	6022
	6099	6182	6257												
PARMBK	1#	3248	3280												
PARMOF	1#	3154													
POP	1918#	10805	11236	11491	11492	11617									
PUSH	1918#	10786	11195	11452	11454	11475	11597								
P. DRVR	1#	3312													
REGDEF	1#	2936													
REPORT	1918#														
R. DRIV	1#	9676													
R. QDRV	1#														
SCOPE	1813#	5339	5363	5389	5437	5476	5524	5580	5616	5640	5673	5723	5864	5946	6018
	6095	6178	6253	6387											
SETPRI	1918#														
SETTRA	11647#	11656	11657	11658	11659	11662	11663	11664	11665						
SETUP	1918#	4439													
SKIP	1918#														
SLASH	1918#														
SPACE	1918#														
STARS	1761	1767	1918#	1980	1991	1993	2000	2016	2100	2103	4609	4617	4764	4778	5332
	5338	5356	5362	5380	5388	5414	5436	5467	5475	5513	5523	5565	5579	5609	5615
	5630	5639	5657	5672	5716	5722	5844	5846	5855	5863	5934	5945	6010	6017	6088
	6094	6170	6177	6222	6252	6380	6451	6453	6488	6496	6505	6512	6520	6526	6571
	6576	6606	6612	6623	6632	6652	6654	6671	6676	6709	6712	6726	6736	6798	6808
	6872	6880	6898	6903	6918	6933	7036	7040	7053	7084	7105	7182	7190	7207	7216
	7232	7241	7259	7271	7356	7370	7421	7424	7444	7448	7479	7483	7505	7514	7536
	7542	7566	7572	7585	7587	7600	7602	7606	7608	7618	7630	7692	7701	7717	7719
	7739	7748	7782	7784	7794	7797	7814	7817	7835	7839	7858	7861	7884	7892	7929
	7939	7957	7960	7970	7973	7983	7986	7997	8000	8011	8016	8035	8038	8052	8061
	8099	8108	8168	8176	8310	8321	8412	8421	8462	8467	8489	8491	8510	8513	8566
	8571	8615	8618	8640	8644	8662	8667	8698	8701	8726	8730	8756	8766	8799	8805
	8832	8848	8903	8905	8935	8941	8955	8964	9072	9082	9129	9187	9195	9542	9546
	9585	9587	9619	9621	9647	9652	9685	9716	9750	9790	10153	10170	10188	10212	10259
	10272	10291	10317	10336	10358	10416	10443	10721	10734	10777	10813	10893	10903	10921	10932
	10984	11023	11085	11108	11185	11252	11304	11310	11374	11377	11389	11447	11504	11581	11626

005

CZR6MDO RK611/06 SS VERIF 1
CZR6MD.P11 10-JAN-78 11:48

MACY11 30A(1052) 10-JAN-78 12:16 PAGE 265
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0541
SEQ 0262

SWRSU	1760#	1918#	4461#												
S.DRVE	1#	3088													
TEST	2664#	5332	5356	5380	5414	5467	5513	5565	5609	5630	5657	5716	5855	5934	6010
	6088	6170	6222												
TRMTRP	11647#														
TYPBIN	1918#														
TYPDEC	1918#	6403													
TYPNAM	1918#														
TYPNUM	1918#														
TYPOCS	1918#														
TYPOCT	1918#														
TYPTXT	1918#														
SDECBN	1#														
SOCTBN	1#	10719													
SSCMRE	2014#	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066
	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081
	2082	2083	2084												
SSCMTM	2014#	2085	2086	2087	2088	2089	2090	2091	2092	2093					
SSESCA	1918#														
SSNEW	1918#	5332	5356	5380	5414	5467	5513	5565	5609	5630	5657	5716	5855	5934	6010
	6088	6170	6222												
SSSET	11647#	11656	11657	11658	11659	11662	11663	11664	11665						
SSSETM	4477#														
SSSKIP	1918#														
.EQUAT	1760#	1808													
.HEADE	1760#	1770													
.KT11	1760#	1918													
.SETUP	1760#	4419													
.SWRHI	1760#	1780													
.SWRLO	1760#	1792#	1793	1794	1795	1796	1797	1798	1799						
.SACT1	1760#	1978													
.SAPT8	2101#														
.SAPTH	1760#	1989													
.SAPTY	1760#	11445													
.SCATC	1760#	1959													
.SCMTA	1760#	2014													
.SDB2D	1760#	11021													
.SDB20	1760#	10982													
.SEOP	1760#	6378													
.SERRO	1760#	11250													
.SPOWE	1760#														
.SRAND	1760#	10775													
.SREAD	1760#	11302													
.SSAVE	1760#	11579													
.SSCOP	1760#	11387													
.SSIZE	1760#	11502													
.SSUPR	1760#	11083													
.STRAP	1760#	11624													
.STYPD	1760#	11183													
.STYPE	1760#	10811													
.STYPO	1760#	11106													

. ABS. 064460 000

E05

CZR6MDD RK611/06 SS VERIF 1 MACY11 30A(1052) 10-JAN-78 12:16 PAGE 266
CZR6MD.P11 10-JAN-78 11:48 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0542
SEQ 0263

ERRORS DETECTED: 0

DSKZ:CZR6MD,DSKZ:CZR6MD.SEG/SOL/CRF/NL:TOC/EQ:QNEWSW/DOC=DSKM:DRIV10.P11,DSKM:CZR6MD.P11
RUN-TIME: 45 40 3 SECONDS
RUN-TIME RATIO: 658/89=7.3
CORE USED: 49K (97 PAGES)

DOCUMENT PAGES: 263

F05