

RK611

DISKLESS CONTROL PART 4
CZR6DB0

AH-9110B-MC

COPYRIGHT © 76-78

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA

B01

EOF1CZP6CCSEQ 00010000 780223 PDP10 411 3H0R1CZP6DBSEQ 00010000 780223
PDP10 411SEQ 0001 CZP6DB0 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 1
CZP6DB.P11 02-DEC-77 10:00

.REM % IDENTIFICATION

PRODUCT CODE: AC-5108B-MC
PRODUCT NAME: CZP6DB0 RK611 DSKLS CTRL PRT 4
DATE: FEB 1978
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: ROY SPITZER

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 ERROR 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 OF RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976, 1978 BY DIGITAL EQUIPMENT CORPORATION

UNCLASSIFIED
DATE 10-10-2001 BY 60322 UCBAW/BJS/STP

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PROGRAMS
- 3.0 OPERATING PROGRAMS
 - 3.1 LOADING PROCEDURE
 - 3.2 STARTING PROCEDURE
 - 3.3 OPTIONAL SWITCH SETTING
 - 3.4 RUN TIME
- 4.0 OPERATING PROCEDURES
 - 4.1 "SOFTWARE" SWITCH REGISTER
 - 4.2 CONTROL C (↑C) OPERATION
 - 4.3 CONTROL S (↑S) OPERATION
 - 4.4 CONTROL Q (↑Q) OPERATION
- 5.0 PROGRAM DESCRIPTION
- 6.0 ERROR REPORTING

C2P6DB0 PK611 DSKLS CTRL PRT4
C2P6DB.P11 02-DEC-77 10:00

1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 4

- A. TESTS THE LOADING OF THE DRIVE BUS MESSAGE SHIFT REGISTERS FOR CLASS C COMMANDS.
- B. TESTS HEADER GENERATION FOR SEARCH OPERATIONS.
- C. TESTS WRITE DATA NPR TRANSFERS TO SILO.
- D. TESTS HEADER RECOGNITION.
- E. TESTS CYLINDER, TRACK, AND SECTOR INCREMENT AFTER SUCCESSFUL HEADER SEARCH.
- F. TESTS DETECTION OF ALL HEADER TYPE ERRORS.
- G. TESTS ECC GENERATION AND WRITING.
- H. TESTS PARTIAL SECTOR WRITE (ZERO FILL).
- I. TESTS 18 BIT FORMAT ECC GENERATION AND DATA WRITE.

NO RK06 DRIVE IS REQUIRED FOR PROGRAM EXECUTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 SYSTEM (16K CORE MEMORY)
CONSOLE TERMINAL
DECTAPE, PAPER TAPE READER, OR DECDISK
RK611 CONTROLLER

2.2 PRELIMINARY PROGRAMS

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 1 (CZR6A)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (CZR6B)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3 (CZR6C)

3.0 OPERATING PROCEDURES

3.1 LOADING PROCEDURE

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

3.2 STARTING PROCEDURE

LOCATION 200 - START PROGRAM
LOCATION 204 - RESTART PROGRAM
LOCATION 214 - REQUEST BUS ADDRESS, VECTOR ADDRESS, AND PRIORITY MODIFICATION

3.3 OPTIONAL SWITCH SETTINGS

SW15 - HALT PROGRAM
SW14 - LOOP ON TEST
SW13 - INHIBIT ERROR TYPE OUT

SW12 - ABORT AFTER 20 ERRORS
SW11 - INHIBIT ITERATION COUNT
SW10 - BELL ON ERROR
SW9 - LOOP ON ERROR
SW8 - LOOP ON TEST IN SWITCHES 0-7

3.4 RUN TIME

FIRST PASS :25 MINUTES
SUBSEQUENT PASSES 3:15 MINUTES

4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPRIATE ADDRESS.

4.1 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/04 OR 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RK06 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

SWR = NNNNNN NEW =

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROES ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

4.2 CONTROL C (↑C) OPERATION

IF ↑C IS TYPED AT ANY TIME DURING THE PROGRAM EXECUTION THE PROGRAM IS HALTED IMMEDIATELY. IF A MONITOR IS PRESENT (XXDP CHAIN, ACT, APT) THE PROGRAM RETURNS CONTROL TO THE MONITOR. IF NO MONITOR IS PRESENT, THE CPU IS HALTED. DEPRESSING THE CONTINUE KEY WILL DO A PROGRAM RESTART.

4.3 CONTROL S (↑S) OPERATION

IF ↑S IS TYPED AT ANY TIME THE PROGRAM WILL GO INTO A STALL LOOP UNTIL A CONTROL Q (↑Q) IS TYPED.

4.4 CONTROL Q (↑Q) OPERATION

IF A $\uparrow$ S HAS BEEN TYPED, TYPING THE $\uparrow$ Q CANCELS THE STALL
INITIATED BY THE $\uparrow$ S.

5.0 PROGRAM DESCRIPTION

**TYPE C INSTRUCTION'S DRIVE MESSAGES

TEST 1 READ DATA SEEK MESSAGE

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND
MAKE SURE IT IS GENERATED PROPERLY.

TEST 2 WRITE DATA SEEK MESSAGE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777,
TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE. CHECK IF
PROPER MESSAGE IS ASSEMBLED.

TEST 3 SEEK MESSAGE FOR WRITE CHECK

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK
OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
IN SEEK COMMAND. MAKE SURE CORRECT MESSAGE
IS ASSEMBLED.

TEST 4 READ DATA CLEAR MESSAGE

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE
CONTROLLER ON DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND THE
CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT

TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777,
TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE AND DRIVE CLEAR
CHECK IF PROPER DRIVE CLEAR MESSAGE IS ASSEMBLED.

TEST 6 DRIVE CLEAR MESSAGE FOR WRITE CHECK

168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR.
MAKE SURE CORRECT MESSAGE IS ASSEMBLED.

**HEADER GENERATION

TEST 7 HEADER GENERATION (PART 1)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE
CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 1
WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0
SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR CYLINDER
1-1777.

TEST 10 HEADER GENERATION (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE
CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR HEADS 1-

TEST 11 HEADER GENERATION (PART 3)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HE
0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE
CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
TO MAKE SURE HEADER IS CORRECT. REPEAT FOR SECTORS 1-25

TEST 12 HEADER GENERATION (PART 4)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
ONE WORD FOR AND RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
MESSAGES, CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE
REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR 2
SECTOR FORMAT.

22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79

**NPR TRANSFER FOR WRITE DATA

TEST 13 WRITE DATA NPR TRANSFER

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 67 WORDS, TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, TRACK 0, DRIVE 0. CLOCK IN SEEK AND DRIVE CLEAR MESSAGE GIVE ENOUGH CLOCK PULSE FOR 68 SILO WORDS. MAKE SURE DATA DOES NOT OCCUR. READ BACK 66 WORDS AND VERIFY THEY ARE CORRECT.

**HEADER RECOGNITION TESTS

TEST 14 WRITE DATA HEADER RECOGNITION

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
140000
140000

MAKE SURE WRITE GATE SETS SHOWING CORRECT HEADER RECOGNITION

TEST 15 SECTOR PULSE DETECTION FOR WRITE DATA

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE AN INDEX PULSE AND A HEADER WITH THE FOLLOWING DATA:

000000
140000
140000

MAKE SURE WRITE GATE DOES NOT SET.

TEST 16 SECTOR INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0 TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,

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
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335

SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
THAT WRITE GATE SETS,

REPEAT FOR SECTOR 1-24.

TEST 17 TRACK INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0
TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
THAT WRITE GATE SETS.

TEST 20 CYLINDER INCREMENT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF 0
TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2,
SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
THAT WRITE GATE SETS.

REPEAT FOR CYLINDER = 1-632.

TEST 21 BAD SECTOR ERROR (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR
MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH
THE FOLLOWING DATA:

000000
040000
040000

MAKE SURE BAD SECTOR ERROR SETS. CHECK THAN DISK ADDRES
IS NOT INCREMENTED.

TEST 22 BAD SECTOR ERROR (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR
MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH

336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391

THE FOLLOWING DATA:

000000
100000
100000

MAKE SURE BAD SECTOR ERROR SETS. CHECK THAT DISK ADDRESS IS NOT INCREMENTED.

TEST 23 OPERATION INCOMPLETE

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253, HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH 1 BIT DIFFERENT IN 30 BITS OF OPI DETERMINATION. ALL SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS THAT SET.

TEST 24 HEADER VRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1253 HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR

MESSAGES SIMULATE A SECTOR PULSE AND HEADER WITH BIT 0 OF THE VRC INCORRECT. MAKE SURE ONLY HEADER VRC AND CONTROLLER ERROR ARE THE ONLY ERRORS SET. REPEAT FOR BITS 1-15 OF VRC.

TEST 25 BAD SECTOR ERROR AND HEADER VRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300. HEAD 1, SECTOR 17. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE THE FOLLOWING HEADER:

000300
040057
040356

MAKE SURE ONLY HEADER VRC ERROR SETS.

TEST 26 GOOD HEADER AND PREVIOUS BSE

CLEAR RK611 CONTROLLER WITH CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF

392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447

K01

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 10

SEQ 0010

ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 100,
HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE
FOLLOWING 3 WORDS WITH A BAD SECTOR INDICATION:

000100
040000
040100

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOW
3 WORDS:

000100
140001
140101

MAKE SURE WRITE GATE SETS INDICATING THAT HEADER HAS
BEEN RECOGNIZED.

TEST 27 GOOD HEADER AND PREVIOUS HVRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF
ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 200,
HEAD 0, SECTOR 1. CLOCK THROUGH SEEK AND DRIVE CLEAR
MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER OF THE

FOLLOWING 3 WORDS WITH A BAD HEADER VRC:

000200
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE
FOLLOWING 3 WORDS:

000200
140001
140201

MAKE SURE WRITE GATE SEES INDICATING THAT HEADER HAS
BEEN RECOGNIZED.

TEST 30 BAD SECTOR ERROR AND PREVIOUS HVRC

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A
WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR
FORMAT, CYLINDER 400, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER OF THE FOLLOWING 3 WORDS WITH A

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

BAD HEADER VRC:

000400
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING 3 WORDS:

000400
040001
040401

MAKE SURE BAD SECTOR ERROR SETS AND HEADER VRC
ERROR DOES NOT SET.

TEST 31 HEADER VRC AND PREVIOUS BSE

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A
WRITE DATA OR ONE WORD TO AN RK06 IN 26 SECTOR
FORMAT, CYLINDER 140, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER OF THE FOLLOWING 3 WORDS WITH A HEADER
VRC ERROR:

000140
040000
040140

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. STIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000140
140001
140101

MAKE SURE HEADER VRC ERROR SETS AND BAD SECTOR
ERROR DOES NOT SET.

TEST 32 OPI AND HVRC ON LAST HEADER

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 240, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:

000240

560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615

140000
140240

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000240
140000
140040

MAKE SURE HEADER VRC AND OPI ERROR SET.

TEST 33 OPI AND PREVIOUS HEADER VRC (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A
SECTOR PULSE AND 30 HEADERS CONSISTING OF THE
FOLLOWING 3 WORDS:

000300
140000
140300

THEN SIMULATE A 3 WORD HEADER CONSISTING ON
THE FOLLOWING DATA:

000300
140000
140200

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND A
HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

000300
140000
140300

MAKE SURE HEADER VRC AND OPI ERRORS SET.

TEST 34 OPI AND PREVIOUS HEADER VRC (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 40, HEAD 0, SECTOR 1. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
PULSE AND A HEADER CONSISTING OF THE FOLLOWING

616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671

THREE WORDS HAVING A BAD HEADER VRC:

000040
140000
140000

MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
DOES NOT SET. SIMULATE A SECTOR PULSE AND 31
HEADERS CONSISTING OF THE FOLLOWING THREE WORDS:

000040
140000
140040

MAKE SURE HEADER VRC AND OPI ERRORS SET.

TEST 35 BSE AND CONTROLLER ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A BSE ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

TEST 36 HVRC AND CONTROLLER ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

TEST 37 READ DATA AND HVRC ERROR

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A HEADER WITH A HVRC ERROR. MAKE SURE CONTROLLER
ERROR AND HVRC SET. CLEAR CONTROLLER AND MAKE SURE
CONTROLLER ERROR RESETS.

TEST 40 WRITE CHECK AND HVRC

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT
CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
A HEADER WITH A HEADER VRC ERROR. MAKE SURE
HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
AND MAKE SURE CONTROLLER ERROR RESETS.

**ECC GENERATION TESTS

TEST 41 ECC INITIALIZATION

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 10 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY FO
DATA WORDS OF ZEROES. MAKE SURE THE ECC PATTERN
REGISTER REMAINS ZERO.

TEST 42 ECC GENERATION (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY THE
FOLLOWING SIX WORDS OF DATA:

005001
040040
020004
000064
000000
000000

CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

TEST 43 ECC GENERATION (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE

AND A GOOD HEADER. CLOCK THROUGH ONLY THE FOLLOWING
SIX WORDS OF DATA:

177777
177777
177777
177777
177777
177777

CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

TEST 44 ECC WRITING

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 400 WORDS. TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGE. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS
AND THE TWO ECC WORDS. MAKE SURE THE TWO ECC WORDS
ARE CORRECT AND WRITTEN PROPERLY. CHECK BUS ADDRESS,
WORD COUNT, CYLINDER, TRACK, AND SECTOR.

**PARTIAL WRITE DATA

TEST 45 ZERO FILL ON WRITE DATA

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS AND
THE TWO ECC WORDS. CHECK THE SECTOR FOR ZERO FILL
AND MAKE SURE THE TWO ECC WORDS ARE WRITTEN PROPERLY.
CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECT

**18 BIT FORMAT WRITES

TEST 46 18 BIT WRITE DATA (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK

728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783

AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777.
VERIFY THAT TWO ZERO BITS ARE PRESENT FOR EACH WORD.

TEST 47 18 BIT WRITE DATA (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777
WITH BAD PARITY SET. VERIFY WRITING OF TWO PARITY
BITS ON SIMULATED DISK.

NOTE: THIS TEST IS ONLY EXECUTED IF MEMORY
PARITY ENABLE IS PRESENT FOR BUFFER
LOCATION.

TEST 50 CLEAR BAD PARITY BUFFER

THE SOLE PURPOSE OF THIS TEST IS TO CLEAR BADPAR
BUFFER AND REMOVE THE BAD PARITY.

TEST 51 18 BIT WRITE DATA (PART 3)

CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH 400 18 BIT WORDS
AND THE 32 BIT ECC. VERIFY THAT THE ECC IS
WRITTEN CORRECTLY.

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION

TEST	ERROR	
NUM	PC	
XXXXXX	YYYYYY	
EXPECT	ACTUAL	OTHER PERTENANT
REG	REG	INFORMATION
ZZZZZZ	WWWWWW	AAAAAA

NOTE: MORE THAN ONE SET OF EXPECT/ACTUAL REGISTERS MAY BE
PRINTED OUT. OTHER PERTENANT INFORMATION MAY CONSIST
OF MORE THAN ONE WORD.

OTHER PERTINENT INFORMATION MAY CONTAIN A WORD LABELED "BIT COUNT". THIS COUNT IS REPORTED WHEN THE OPERATION BEING PERFORMED INVOLVES CLOCKING THE CONTROLLER THROUGH ONE OR MORE OF THE VARIOUS FIELDS THAT MAKE UP THE PHYSICAL SECTOR FORMAT.

THESE FIELDS (ALL VALUES GIVEN IN OCTAL) ARE:

FIELD	BITS	WORDS
HEADER PREAMBLE	400	20
HEADER	60	3
GAP	100	4
DATA PREAMBLE	400	20
DATA		
(22(10) SECTOR/TRACK)	10000	400
(20(10) SECTOR/TRACK)	11000	400
ECC	40	2
POSTAMBLE	20	1
GAP		
(22(10) SECTOR/TRACK)	160	7
(20(10) SECTOR/TRACK)	140	6

REFER TO THE RK06 UNIBUS DISK SUBSYSTEM SPECIFICATION FOR MORE DETAILED DESCRIPTION.

THE "BIT COUNT" REPORTED IS INITIALIZED AT THE START OF EACH FIELD AND IS INCREMENTED FOR EACH BIT PROCESSED.

WHEN THE OPERATION BEING PERFORMED INVOLVES WRITING, OTHER PERTINENT INFORMATION MAY CONTAIN WORDS LABELED PRESENT BIT, PRESENT BIT -1, PRESENT BIT -2, AND PRESENT BIT +1. THESE BITS ARE PRESENTED IN THIS WAY TO SHOW THE WRITE DATA STREAM IN THE SAME MANNER THAT THE RK611 LOOKS AT THE DATA STREAM TO COMPUTE PRECOMPENSATION ADVANCE AND PRECOMPENSATION DELAY IN THE WRITE OPERATION.

WHEN THE OPERATION BEING PERFORMED INVOLVES READING, OTHER PERTINENT INFORMATION MAY CONTAIN PREVIOUS BIT AND PRESENT BIT WORDS. THESE BITS RELATE TO RK611 LOGIC THAT DETERMINES WHEN THE BIT IS VALID FROM THE DECODER.

%

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 18

SEQ 0018

```

880      : *** REV 003 ***
881      .NLIST CND,MD,MC
882      .LIST ME
883      .ENABL ABS,AMA
884      $SWR= 167400
885      $TN= 1
886      .TITLE CZR6DB0 RK611 DSKLS CTRL PRT4
887      ;*COPYRIGHT (C) 1976,1977
888      ;*DIGITAL EQUIPMENT CORP.
889      ;*MAYNARD, MASS. 01754
890      ;*
891      ;*PROGRAM BY ROY SPITZER
892      ;*
893      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
894      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
895      ;*
896      .SBTTL OPERATIONAL SWITCH SETTINGS
897      ;*
898      ;*      SWITCH      USE
899      ;*      -----
900      ;*      15      HALT ON ERROR
901      ;*      14      LOOP ON TEST
902      ;*      13      INHIBIT ERROR TYPEOUTS
903      ;*      12      ABORT PROGRAM AFTER 20 ERRORS
904      ;*      11      INHIBIT ITERATIONS
905      ;*      10      BELL ON ERROR
906      ;*      9       LOOP ON ERROR
907      ;*      8       LOOP ON TEST IN SWR<7:0>
908      .SBTTL BASIC DEFINITIONS
909      ;*
910      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
911      STACK= 1100
912      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
913      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
914      ;*
915      ;*MISCELLANEOUS DEFINITIONS
916      HT= 11      ;;CODE FOR HORIZONTAL TAB
917      LF= 12      ;;CODE FOR LINE FEED
918      CR= 15      ;;CODE FOR CARRIAGE RETURN
919      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
920      PS= 177776   ;;PROCESSOR STATUS WORD
921      .EQUIV PS,PSW
922      STKLMT= 177774 ;;STACK LIMIT REGISTER
923      PIRQ= 177772  ;;PROGRAM INTERRUPT REQUEST REGISTER
924      DSWR= 177570  ;;HARDWARE SWITCH REGISTER
925      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
926      ;*
927      ;*GENERAL PURPOSE REGISTER DEFINITIONS
928      R0= %0      ;;GENERAL REGISTER
929      R1= %1      ;;GENERAL REGISTER
930      R2= %2      ;;GENERAL REGISTER
931      R3= %3      ;;GENERAL REGISTER
932      R4= %4      ;;GENERAL REGISTER
933      R5= %5      ;;GENERAL REGISTER
934      R6= %6      ;;GENERAL REGISTER
935      R7= %7      ;;GENERAL REGISTER

```

G02

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 19
BASIC DEFINITIONS

SEQ 0019

```

936      000006      SP=      %6      ;;STACK POINTER
937      000007      PC=      %7      ;;PROGRAM COUNTER
938
939      ;*PRIORITY LEVEL DEFINITIONS
940      000000      PR0=      0      ;;PRIORITY LEVEL 0
941      000040      PR1=      40     ;;PRIORITY LEVEL 1
942      000100      PR2=      100    ;;PRIORITY LEVEL 2
943      000140      PR3=      140    ;;PRIORITY LEVEL 3
944      000200      PR4=      200    ;;PRIORITY LEVEL 4
945      000240      PR5=      240    ;;PRIORITY LEVEL 5
946      000300      PR6=      300    ;;PRIORITY LEVEL 6
947      000340      PR7=      340    ;;PRIORITY LEVEL 7
948
949      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
950      100000      SW15=      100000
951      040000      SW14=      40000
952      020000      SW13=      20000
953      010000      SW12=      10000
954      004000      SW11=      4000
955      002000      SW10=      2000
956      001000      SW09=      1000
957      000400      SW08=      400
958      000200      SW07=      200
959      000100      SW06=      100
960      000040      SW05=      40
961      000020      SW04=      20
962      000010      SW03=      10
963      000004      SW02=      4
964      000002      SW01=      2
965      000001      SW00=      1
966      .EQUIV      SW09,SW9
967      .EQUIV      SW08,SW8
968      .EQUIV      SW07,SW7
969      .EQUIV      SW06,SW6
970      .EQUIV      SW05,SW5
971      .EQUIV      SW04,SW4
972      .EQUIV      SW03,SW3
973      .EQUIV      SW02,SW2
974      .EQUIV      SW01,SW1
975      .EQUIV      SW00,SW0
976
977      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
978      100000      BIT15=      100000
979      040000      BIT14=      40000
980      020000      BIT13=      20000
981      010000      BIT12=      10000
982      004000      BIT11=      4000
983      002000      BIT10=      2000
984      001000      BIT09=      1000
985      000400      BIT08=      400
986      000200      BIT07=      200
987      000100      BIT06=      100
988      000040      BIT05=      40
989      000020      BIT04=      20
990      000010      BIT03=      10
991      000004      BIT02=      4

```

BASIC DEFINITIONS

```

992          000002      BIT01= 2
993          000001      BIT00= 1
994          .EQUIV      BIT09,BIT9
995          .EQUIV      BIT08,BIT8
996          .EQUIV      BIT07,BIT7
997          .EQUIV      BIT06,BIT6
998          .EQUIV      BIT05,BIT5
999          .EQUIV      BIT04,BIT4
1000         .EQUIV      BIT03,BIT3
1001         .EQUIV      BIT02,BIT2
1002         .EQUIV      BIT01,BIT1
1003         .EQUIV      BIT00,BIT0
1004
1005         ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1006         ERRVEC= 4      ; TIME OUT AND OTHER ERRORS
1007         RESVEC= 10     ; RESERVED AND ILLEGAL INSTRUCTIONS
1008         TBITVEC= 14    ; "T" BIT
1009         TRTVEC= 14     ; TRACE TRAP
1010         BPTVEC= 14     ; BREAKPOINT TRAP (BPT)
1011         IOTVEC= 20     ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
1012         PWRVEC= 24     ; POWER FAIL
1013         EMTVEC= 30     ; EMULATOR TRAP (EMT) **ERROR**
1014         TRAPVEC= 34    ; "TRAP" TRAP
1015         TKVEC= 60      ; TTY KEYBOARD VECTOR
1016         TPVEC= 64     ; TTY PRINTER VECTOR
1017         PIRQVEC= 240   ; PROGRAM INTERRUPT REQUEST VECTOR
1018         MEMVEC= 114    ; MEMORY PARITY VECTOR
1019         MEMBAS= 172100 ; MEMORY PARITY BUS ADDRESS
1020         PAR.EN= 1      ; ENABLE MEMORY PARITY
1021         WR.PAR= 5      ; WRITE BAD PARITY
1022         AVECT1= 210    ; DEFINE RK611 VECTOR ADDRESS
1023         APRIOR= PR5    ; DEFINE RK611 PRIORITY
1024         ABASE= 177440 ; DEFINE BASE OF RK611 REGISTERS
1025
1026         .SBTTL RK611 CONTROLLER REGISTER DEFINITION
1027
1028         RKCS1= 0        ; CONTROL AND STATUS REGISTER 1
1029         RKWC= 2         ; WORD COUNT REGISTER
1030         RKBA= 4         ; BUS ADDRESS REGISTER
1031         RKDA= 6         ; DESIRED TRACK SECTOR REGISTER
1032         RKCS2= 10       ; CONTROL AND STATUS REGISTER 2
1033         RKDS= 12        ; DRIVE STATUS REGISTER
1034         RKER= 14        ; ERROR REGISTER
1035         RKASOF= 16       ; ATTENTION SUMMARY AND OFFSET REGISTER
1036         RKDCYL= 20      ; DESIRED CYLINDER REGISTER
1037         RKDB= 24        ; DATA BUFFER
1038         RKMR1= 26       ; MAINTENANCE REGISTER 1
1039         RKMR2= 34       ; MAINTENANCE REGISTER 2
1040         RKMR3= 36       ; MAINTENANCE REGISTER 3
1041         RKECPS= 30      ; ECC POSITION INFORMATION
1042         RKECPT= 32      ; ECC PATTERN INFORMATION
1043         RKSPAR= 22      ; SPARE REGISTER
1044
1045         .SBTTL DRIVE COMMANDS
1046
1047         SELDRV= 01      ; SELECT DRIVE

```

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 21
DRIVE COMMANDS

SEQ 0021

1048	000003	PACK= 03	; PACK ACKNOWLEDGE
1049	000005	CLEAR= 05	; DRIVE CLEAR
1050	000007	UNLOAD= 07	; UNLOAD
1051	000011	SRTSPL= 11	; START SPINDLE
1052	000013	RECAL= 13	; RECALIBRATE
1053	000015	OFFSET= 15	; OFFSET
1054	000017	SEEK= 17	; SEEK
1055	000021	RDDATA= 21	; READ DATA
1056	000023	WRDATA= 23	; WRITE DATA
1057	000025	RDHEAD= 25	; READ HEADER
1058	000027	WRHEAD= 27	; WRITE HEADER AND DATA
1059	000031	WATCHK= 31	; WRITE CHECK
1060	000300	INTR= 300	; GENERATE INTERRUPT TO CPU
1061			
1062		.SBTTL CONTROL AND STATUS REGISTER 1 BITS	
1063			
1064	000001	GO= BIT0	; GO BIT
1065	000100	IE= BIT6	; INTERRUPT ENABLE
1066	000200	RDY= BIT7	; CONTROLLER READY
1067	000400	BA16= BIT8	; BUS ADDRESS BIT 16
1068	001000	BA17= BIT9	; BUS ADDRESS BIT 17
1069	002000	CDT= BIT10	; CONTROLLER DRIVE TYPE (0=RK06)
1070	004000	CTO= BIT11	; CONTROLLER TIMED OUT WAITING FOR DRIVE RESPONSE
1071			
1072	010000	CFMT= BIT12	; CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1073	020000	SPAR= BIT13	; DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1074	040000	DI= BIT14	; DRIVE INTERRUPT
1075	100000	CERR= BIT15	; CONTROLLER ERROR
1076	100000	CCLR= BIT15	; CONTROLLER CLEAR
1077			
1078		.SBTTL CONTROL AND STATUS REGISTER 2 BITS	
1079			
1080	000007	DRVMSK= 7	; MASK FOR DRIVE SELECTION CODE
1081	000010	RLS= BIT3	; DESELECT OR RELEASE DRIVE IN BITS 0-2
1082	000020	BAI= BIT4	; BUS ADDRESS INCREMENT INHIBIT
1083	000040	SCLR= BIT5	; CLEAR CONTROLLER AND ALL DRIVES
1084	000100	IR= BIT6	; INPUT READY
1085	000200	OR= BIT7	; OUTPUT READY
1086	000400	UFE= BIT8	; UNIT FIELD ERROR
1087	001000	MDS= BIT9	; MULTIPLE DRIVE SELECT
1088	002000	PGE= BIT10	; PROGRAMMING ERROR
1089	004000	NEM= BIT11	; NON-EXISTENT MEMORY
1090	010000	NED= BIT12	; NON-EXISTENT DRIVE
1091	020000	UPE= BIT13	; UNIBUS PARITY ERROR
1092	040000	WCE= BIT14	; WRITE CHECK ERROR
1093	100000	DLT= BIT15	; DATA LATE ERROR
1094			
1095		.SBTTL ERROR REGISTER BIT DEFINITION	
1096			
1097	000001	ILF= BIT0	; ILLEGAL FUNCTION CODE
1098	000002	SKI= BIT1	; SEEK INCOMPLETE
1099	000004	NXF= BIT2	; NON-EXECUTABLE DRIVE FUNCTION
1100	000010	DRPAR= BIT3	; DRIVE DETECTED DRIVE BUS PARITY ERROR
1101	000020	FMTE= BIT4	; FORMAT ERROR
1102	000040	DTYE= BIT5	; DRIVE TYPE ERROR
1103	000100	ECH= BIT6	; ECC HARD

1104	000200	BSE=	BIT7	;BAD SECTOR ERROR
1105	000400	HVRC=	BIT8	;HEADER VRC ERROR
1106	001000	COE=	BIT9	;CYLINDER ADDRESS OVERFLOW ERROR
1107	002000	IDAE=	BIT10	;INVALID DISK ADDRESS ERROR
1108	004000	WLE=	BIT11	;WRITE LOCK ERROR
1109	010000	DTE=	BIT12	;DRIVE TIMING ERROR
1110	020000	OPI=	BIT13	;OPERATION (SEARCH) INCOMPLETE
1111	040000	UNS=	BIT14	;DRIVE UNSAFE
1112	100000	DCK=	BIT15	;DATA CHECK
1113				
1114		.SBTTL	STATUS REGISTER BIT DEFINITION	
1115				
1116	000001	DRA=	BIT0	;DRIVE AVAILABLE (CONTROLLER IS SET IF
1117				; THIS BIT IS RESET)
1118	000004	OFST=	BIT2	;DRIVE OFFSET
1119	000010	ACLO=	BIT3	;AC LOW
1120	000020	SPOLSS=	BIT4	;SPEED LOSS
1121	000040	DROT=	BIT5	;DRIVE OFF TRACK
1122	000100	VV=	BIT6	;VOLUME VALID
1123	000200	DRDY=	BIT7	;DRIVE READY
1124	000400	DDT=	BIT8	;DRIVE TYPE (0=RK06)
1125	004000	WRL=	BIT11	;WRITE LOCK
1126	020000	PIP=	BIT13	;POSITIONING IN PROGRESS
1127	040000	DSC=	BIT14	;DRIVE STATUS CHANGE
1128	100000	SVAL=	BIT15	;STATUS VALID
1129				
1130		.SBTTL	MAINTENANCE REGISTER 1 BIT DEFINITION	
1131				
1132	000017	MESMSK=	17	;MESSAGE MASK
1133				
1134	000020	PAT=	BIT4	;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1135	000040	DMD=	BIT5	;DIAGNOSTIC MODE
1136	000100	MSP=	BIT6	;MAINTENANCE SECTOR PULSE
1137	000200	MIND=	BIT7	;MAINTENANCE INDEX
1138	000400	MCLK=	BIT8	;MAINTENANCE CLOCK
1139	001000	MERD=	BIT9	;MAINTENANCE ENCODED READ DATA
1140	002000	MEWD=	BIT10	;MAINTENANCE ENCODED WRITE DATA
1141	004000	PCA=	BIT11	;PRECOMPENSATION ADVANCE
1142	010000	PCD=	BIT12	;PRECOMPENSATION DELAY
1143	020000	ECCW=	BIT13	;ECC WORD IS BEING READ OR WRITTEN
1144	040000	WRTGAT=	BIT14	;WRITE GATE
1145	100000	RDGATE=	BIT15	;READ GATE
1146				
1147		.SBTTL	TRANSMITTED MESSAGE A	
1148				
1149	000020	S. SEEK=	BIT4	;SEEK COMMAND
1150	000040	S. RECL=	BIT5	;RECALIBRATE COMMAND
1151	000100	S. STSP=	BIT6	;START SPINDLE COMMAND
1152	000200	S. RTC=	BIT7	;DRIVE RETURN TO CENTERLINE COMMAND
1153	000400	S. CLR=	BIT8	;CLEAR ERROR AND DSC
1154	001000	S. FMT=	BIT9	;FORMAT
1155	002000	S. UNLD=	BIT10	;UNLOAD
1156	004000	S. PACK=	BIT11	;SET VOLUME VALID (PACK ACKNOWLEDGE)
1157		.SBTTL	TRAP CATCHER	
1158				
1159	000000	.	=0	


```

1160 ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1161 ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1162 ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1163 ;.=174
1164 000174 000000 DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
1165 000176 000000 SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
1166 .SBTTL STARTING ADDRESS(ES)
1167 000200 000137 003334 JMP 2#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1168 000204 000137 003324 JMP RESTRT ;;JUMP TO RESTART ROUTINE
1169 000214 000214 .=214
1170 000214 000137 003314 JMP PARM ;JUMP TO OPERATOR ASSIGNED PARAMETERS
1171 .SBTTL ACT11 HOOKS
1172 ;*****
1173 ;HOOKS REQUIRED BY ACT11
1174 $SVPC= ;SAVE PC
1175 000220 .=46 ;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1176 000046 000046 $ENDAD
1177 000052 000052 .=52 ;2)SET LOC.52 TO ZERO
1178 000052 000000 .WORD 0 ;; RESTORE PC
1179 000052 000000 .=$SVPC
1180 000220 000114 .=MEMVEC
1181 000114 034526 MEMERR ;MEMPARITY ERROR HANDLER
1182 000116 000340 PR7
1183 000116 001000 .=1000
1184 .SBTTL APT PARAMETER BLOCK
1185 ;*****
1186 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1187 ;*****
1188 ;*****
1189 ;*****
1190 001000 .SX= ;SAVE CURRENT LOCATION
1191 000024 .=24 ;SET POWER FAIL TO POINT TO START OF PROGRAM
1192 000024 000200 200 ;FOR APT START UP
1193 000044 .=44 ;POINT TO APT INDIRECT ADDRESS PNTR.
1194 000044 001000 $APTHDR ;POINT TO APT HEADER BLOCK
1195 001000 .=$X ;RESET LOCATION COUNTER
1196 ;*****
1197 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1198 ;INTERFACE SPEC.
1199
1200 $APTHD:
1201 001000 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1202 001002 001214 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1203 001004 000000 $TSTM: .WORD ;;RUN TIM OF LONGEST TEST
1204 001006 000000 $PASTM: .WORD ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1205 001010 000000 $UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1206 001012 000032 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1207 000100 MSP= BIT6 ;MAINTENANCE SECTOR PULSE
1208 000200 MIND= BIT7 ;MAINTENANCE INDEX
1209 000400 MCLK= BIT8 ;MAINTENANCE CLOCK
1210 001000 MERD= BIT9 ;MAINTENANCE ENCODED READ DATA
1211 002000 MEWD= BIT10 ;MAINTENACNE ENCODED WRITE DATA
1212 004000 PCA= BIT11 ;PRECOMPENSATION ADVANCE
1213 010000 PCD= BIT12 ;PRECOMPENSATION DELAY
1214 020000 ECCW= BIT13 ;ECC WORD IS BEING READ OR WRITTEN
1215 040000 WRTGAT= BIT14 ;WRITE GATE

```



```

1216      100000      RDGATE= BIT15      ;READ GATE
1217
1218      .SBTTL  TRANSMITTED MESSAGE A
1219
1220      000020      S.SEEK= BIT4      ;SEEK COMMAND
1221      000040      S.RECL= BIT5      ;RECALIBRATE COMMAND
1222      000100      S.STSP= BIT6      ;START SPINDLE COMMAND
1223      000200      S.RTC= BIT7      ;DRIVE RETURN TO CENTERLINE COMMAND
1224      000400      S.CLR= BIT8      ;CLEAR ERROR AND DSC
1225      001000      S.FMT= BIT9      ;FORMAT
1226      002000      S.UNLD= BIT10     ;UNLOAD
1227      004000      S.PACK= BIT11    ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1228      .SBTTL  TRAP CATCHER
1229
1230      000000      .=0
1231      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1232      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1233      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1234      .=174
1235      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1236      000176      000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
1237      .SBTTL  STARTING ADDRESS(ES)
1238      000200      000137      003334      JMP 2*START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1239      000204      000137      003324      JMP RESTART ;;JUMP TO RESTART ROUTINE
1240      .=214
1241      000214      000137      003314      JMP PARM      ;JUMP TO OPERATOR ASSIGNED PARMETERS
1242      .SBTTL  ACT11 HOOKS
1243
1244      ;*****
1245      ;HOOKS REQUIRED BY ACT11
1246      .SSVPC=.      ;SAVE PC
1247      .=46
1248      000046      034062      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1249      000052      000052      .=52
1250      000052      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
1251      000220      000114      .=$SVPC      ;; RESTORE PC
1252      000114      034526      .MEMVEC
1253      000116      000340      MEMERR      ;MEMPARITY ERROR HANDLER
1254      000116      000340      PR7
1255      .=1000
1256      .SBTTL  APT PARAMETER BLOCK
1257
1258      ;*****
1259      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1260      ;*****
1261      001000      .SX=.      ;SAVE CURRENT LOCATION
1262      000024      .=24      ;SET POWER FAIL TO POINT TO START OF PROGRAM
1263      000024      000200      200      ;FOR APT START UP
1264      000044      .=44      ;POINT TO APT INDIRECT ADDRESS PNTR.
1265      000044      001000      $APTHDR ;POINT TO APT HEADER BLOCK
1266      001000      .=$X      ;RESET LOCATION COUNTER
1267      ;*****
1268      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1269      ;INTERFACE SPEC.
1270
1271      001000      $APTHD:

```

M02

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 25
APT PARAMETER BLOCK

SEQ 0025

1272 001000 000000
1273 001002 001214
1274 001004 000000
1275 001006 000000
1276 001010 000000
1277 001012 000032

\$HIBTS: .WORD 0 ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MBADR: .WORD \$MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
\$TSTM: .WORD ;; RUN TIME OF LONGEST TEST
\$PASTM: .WORD ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
SETEND-\$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

1278
1279
1280
1281
1282
1283
1284
1285 001100
1286 001100 000000
1287 001102 000
1288 001103 000
1289 001104 000000
1290 001106 000000
1291 001110 000000
1292 001112 000000
1293 001114 000
1294 001115 001
1295 001116 000000
1296 001120 000000
1297 001122 000000
1298 001124 000000
1299 001126 000000
1300 001130 000000
1301 001132 000000
1302 001134 000
1303 001135 000
1304 001136 000000
1305 001140 177570
1306 001142 177570
1307 001144 177560
1308 001146 177562
1309 001150 177564
1310 001152 177566
1311 001154 000
1312 001155 002
1313 001156 012
1314 001157 000
1315 001160 000000
1316 001162 000000
1317 001164 000000
1318 001166 000000
1319 001170 000000
1320 001172 000000
1321 001174 000000
1322 001176 000000
1323 001200 000000
1324 001202 000000
1325 001204 177607 000377
1326 001210 077
1327 001211 015
1328 001212 000012
1329
1330
1331
1332
1333

.SBTTL COMMON TAGS

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

SCMTAG: . =1100

;; START OF COMMON TAGS

STSTNM:	.WORD	0	;; CONTAINS THE TEST NUMBER
SERFLG:	.BYTE	00	;; CONTAINS ERROR FLAG
SICNT:	.WORD	00	;; CONTAINS SUBTEST ITERATION COUNT
SLPADR:	.WORD	00	;; CONTAINS SCOPE LOOP ADDRESS
SLPERR:	.WORD	00	;; CONTAINS SCOPE RETURN FOR ERRORS
SERTTL:	.WORD	00	;; CONTAINS TOTAL ERRORS DETECTED
SITEMB:	.BYTE	0	;; CONTAINS ITEM CONTROL BYTE
SERMAX:	.BYTE	1	;; CONTAINS MAX. ERRORS PER TEST
SERRPC:	.WORD	0	;; CONTAINS PC OF LAST ERROR INSTRUCTION
SGDADR:	.WORD	00	;; CONTAINS ADDRESS OF 'GOOD' DATA
SBDADR:	.WORD	00	;; CONTAINS ADDRESS OF 'BAD' DATA
SGDDAT:	.WORD	00	;; CONTAINS 'GOOD' DATA
SBDAT:	.WORD	00	;; CONTAINS 'BAD' DATA
	.WORD	00	;; RESERVED--NOT TO BE USED
	.WORD	00	
SAUTOB:	.BYTE	00	;; AUTOMATIC MODE INDICATOR
SINTAG:	.BYTE	00	;; INTERRUPT MODE INDICATOR
	.WORD	0	
SWR:	.WORD	DSWR	;; ADDRESS OF SWITCH REGISTER
DISPLAY:	.WORD	DDISP	;; ADDRESS OF DISPLAY REGISTER
STKS:	177560		;; TTY KBD STATUS
STKB:	177562		;; TTY KBD BUFFER
STPS:	177564		;; TTY PRINTER STATUS REG. ADDRESS
STPB:	177566		;; TTY PRINTER BUFFER REG. ADDRESS
SNULL:	.BYTE	0	;; CONTAINS NULL CHARACTER FOR FILLS
SFILLS:	.BYTE	2	;; CONTAINS # OF FILLER CHARACTERS REQUIRED
SFILLC:	.BYTE	12	;; INSERT FILL CHARS. AFTER A "LINE FEED"
STPFLG:	.BYTE	0	;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
STMP0:	.WORD	0	;; USER DEFINED
STMP1:	.WORD	00	;; USER DEFINED
STMP2:	.WORD	00	;; USER DEFINED
STMP3:	.WORD	C	;; USER DEFINED
STMP4:	.WORD	00	;; USER DEFINED
STMP5:	.WORD	00	;; USER DEFINED
STMP6:	.WORD	00	;; USER DEFINED
STMP7:	.WORD	0	;; USER DEFINED
STIMES:	0		;; MAX. NUMBER OF ITERATIONS
SESCAPE:	0		;; ESCAPE ON ERROR ADDRESS
SBELL:	.ASCIZ	<207><377><377>	;; CODE FOR BELL
SQUES:	.ASCII	/ ? /	;; QUESTION MARK
SCRLF:	.ASCII	<15>	;; CARRIAGE RETURN
SLF:	.ASCIZ	<12>	;; LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN

1334	001214		\$MAIL:		;; APT MAILBOX
1335	001214	000000	\$MSGTY:	.WORD	;; MESSAGE TYPE CODE
1336	001216	000000	\$FATAL:	.WORD	;; FATAL ERROR NUMBER
1337	001220	000000	\$TESTN:	.WORD	;; TEST NUMBER
1338	001222	000000	\$PASS:	.WORD	;; PASS COUNT
1339	001224	000000	\$DEVCT:	.WORD	;; DEVICE COUNT
1340	001226	000000	\$UNIT:	.WORD	;; I/O UNIT NUMBER
1341	001230	000000	\$MSGAD:	.WORD	;; MESSAGE ADDRESS
1342	001232	000000	\$MSGLG:	.WORD	;; MESSAGE LENGTH
1343	001234		\$ETABLE:		;; APT ENVIRONMENT TABLE
1344	001234	000	\$ENV:	.BYTE	;; ENVIRONMENT BYTE
1345	001235	000	\$ENVM:	.BYTE	;; ENVIRONMENT MODE BITS
1346	001236	000000	\$SWREG:	.WORD	;; APT SWITCH REGISTER
1347	001240	000000	\$USWR:	.WORD	;; USER SWITCHES
1348	001242	000000	\$CPUOP:	.WORD	;; CPU TYPE, OPTIONS
1349			;;		BITS 15-11=CPU TYPE
1350			;;		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1351			;;		11/70=06, PDQ=07, Q=10
1352			;;		BIT 10=REAL TIME CLOCK
1353			;;		BIT 9=FLOATING POINT PROCESSOR
1354			;;		BIT 8=MEMORY MANAGEMENT
1355	001244	000	\$MAMS1:	.BYTE	;; HIGH ADDRESS, M.S. BYTE
1356	001245	000	\$MTYP1:	.BYTE	;; MEM. TYPE, BLK#1
1357			;;		MEM. TYPE BYTE -- (HIGH BYTE)
1358			;;		900 NSEC CORE=001
1359			;;		300 NSEC BIPOLAR=002
1360			;;		500 NSEC MOS=003
1361	001246	000000	\$MADR1:	.WORD	;; HIGH ADDRESS, BLK#1
1362			;;		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1363	001250	000	\$MAMS2:	.BYTE	;; HIGH ADDRESS, M.S. BYTE
1364	001251	000	\$MTYP2:	.BYTE	;; MEM. TYPE, BLK#2
1365	001252	000000	\$MADR2:	.WORD	;; MEM. LAST ADDRESS, BLK#2
1366	001254	000	\$MAMS3:	.BYTE	;; HIGH ADDRESS, M.S. BYTE
1367	001255	000	\$MTYP3:	.BYTE	;; MEM. TYPE, BLK#3
1368	001256	000000	\$MADR3:	.WORD	;; MEM. LAST ADDRESS, BLK#3
1369	001260	000	\$MAMS4:	.BYTE	;; HIGH ADDRESS, M.S. BYTE
1370	001261	000	\$MTYP4:	.BYTE	;; MEM. TYPE, BLK#4
1371	001262	000000	\$MADR4:	.WORD	;; MEM. LAST ADDRESS, BLK#4
1372	001264	000210	\$VECT1:	.WORD	;; INTERRUPT VECTOR#1, BUS PRIORITY#1
1373	001266	000000	\$VECT2:	.WORD	;; INTERRUPT VECTOR#2, BUS PRIORITY#2
1374	001270	177440	\$BASE:	.WORD	;; BASE ADDRESS OF EQUIPMENT UNDER TEST
1375	001272	000000	\$DEVN:	.WORD	;; DEVICE MAP
1376	001274	000000	\$CDW1:	.WORD	;; CONTROLLER DESCRIPTION WORD#1
1377	001276	000000	\$CDW2:	.WORD	;; CONTROLLER DESCRIPTION WORD#2
1378	001300		\$EEND:		
1379			.MEXIT		

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

; ERROR 1: UNEXPECTED MEMORY PARTIY ENAB.E TRAP
 ; EM000
 ; DH000C
 ; DT000
 ; DF000

.EMW: ERROR 2: WRITE BIT ERROR

; 0
 ; 0
 ; DT002
 ; DF002

; ERROR 3: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
 ; CS1 INCORRECT
 ; EM300
 ; EM4000
 ; DT003
 ; DF003

; ERROR 4: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
 ; MESS A INCORRECT
 ; EM300
 ; EM4001
 ; DT003
 ; DF003

; ERROR 5: ATTEMPTING TO CHECK SEEK MESSAGE FOR READ DATA
 ; MESS B INCORRECT
 ; EM300
 ; EM4002
 ; DT003
 ; DF003

; ERROR 6: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA
 ; CS1 INCORRECT
 ; EM301
 ; EM4000
 ; DT003
 ; DF003

; ERROR 7: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA
 ; MESS A INCORRECT
 ; EM301
 ; EM4001
 ; DT003
 ; DF003

; ERROR 10: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE DATA

1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388
 1389
 1390
 1391
 1392
 1393
 1394 001300
 1395
 1396 001300 046111
 1397 001302 044116
 1398 001304 042456
 1399 001306 043012
 1400
 1401 001310 000000
 1402 001312 000000
 1403 001314 042462
 1404 001316 043016
 1405
 1406
 1407 001320 046156
 1408 001322 052052
 1409 001324 042504
 1410 001326 043042
 1411
 1412
 1413 001330 046156
 1414 001332 052070
 1415 001334 042504
 1416 001336 043042
 1417
 1418
 1419 001340 046156
 1420 001342 052111
 1421 001344 042504
 1422 001346 043042
 1423
 1424
 1425 001350 046233
 1426 001352 052052
 1427 001354 042504
 1428 001356 043042
 1429
 1430
 1431 001360 046233
 1432 001362 052070
 1433 001364 042504
 1434 001366 043042
 1435

1436			:	MESS B INCORRECT
1437	001370	046233	:	EM301
1438	001372	052111	:	EM4002
1439	001374	042504	:	DT003
1440	001376	043042	:	DF003
1441			:	ERROR 11: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1442			:	CS1 INCORRECT
1443	001400	046311	:	EM302
1444	001402	052052	:	EM4000
1445	001404	042504	:	DT003
1446	001406	043042	:	DF003
1447			:	ERROR 12: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1448			:	MESS A INCORRECT
1449	001410	046311	:	EM302
1450	001412	052070	:	EM4001
1451	001414	042504	:	DT003
1452	001416	043042	:	DF003
1453			:	ERROR 13: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1454			:	MESS B INCORRECT
1455	001420	046311	:	EM302
1456	001422	052111	:	EM4002
1457	001424	042504	:	DT003
1458	001426	043042	:	DF003
1459			:	ERROR 14: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1460			:	CS1 INCORRECT
1461	001430	046370	:	EM303
1462	001432	052052	:	EM4000
1463	001434	042504	:	DT003
1464	001436	043042	:	DF003
1465			:	ERROR 15: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1466			:	MESS A INCORRECT
1467	001440	046370	:	EM303
1468	001442	052070	:	EM4001
1469	001444	042504	:	DT003
1470	001446	043042	:	DF003
1471			:	ERROR 16: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1472			:	MESS B INCORRECT
1473	001450	046370	:	EM303
1474	001452	052111	:	EM4002
1475	001454	042504	:	DT003
1476	001456	043042	:	DF003
1477			:	ERROR 17: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1478			:	CS1 INCORRECT
1479	001460	046446	:	EM304
1480	001462	052052	:	EM4000
1481	001464	042504	:	DT003
1482	001466	043042	:	DF003
1483			:	ERROR 20: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1484			:	MESS A INCORRECT
1485	001470	046446	:	EM304
1486	001472	052070	:	EM4001
1487	001474	042504	:	DT003
1488	001476	043042	:	DF003
1489			:	ERROR 21: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1490			:	MESS B INCORRECT
1491	001500	046446	:	EM304

CZR6080 RK611 DSKLS CTRL PRTH
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 30
ERROR POINTER TABLE

SEQ 0030

1492	001502	052111	EM4002
1493	001504	042504	DT003
1494	001506	043042	DF003
1495			ERROR 22: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1496			CS1 INCORRECT
1497	001510	046525	EM305
1498	001512	052052	EM4000
1499	001514	042504	DT003
1500	001516	043042	DF003
1501			ERROR 23: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1502			MESS A INCORRECT
1503	001520	046525	EM305
1504	001522	052070	EM4001
1505	001524	042504	DT003
1506	001526	043042	DF003
1507			ERROR 24: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1508			MESS B INCORRECT
1509	001530	046525	EM305
1510	001532	052111	EM4002
1511	001534	042504	DT003
1512	001536	043042	DF003
1513			ERROR 25: ATTEMPTING TO CHECK HEADER GENERATION
1514			WITH VARIOUS CYLINDER VALUES
1515			CS1 INCORRECT
1516	001540	046605	EM306
1517	001542	052052	EM4000
1518	001544	042504	DT003
1519	001546	043066	DF025
1520			ERROR 26: ATTEMPTING TO CHECK HEADER GENERATION
1521			WITH VARIOUS CYLINDER VALUES
1522			FIRST WORD OF HEADER INCORRECT
1523	001550	046605	EM306
1524	001552	052132	EM4003
1525	001554	042504	DT003
1526	001556	043066	DF025
1527			ERROR 27: ATTEMPTING TO CHECK HEADER GENERATION
1528			WITH VARIOUS CYLINDER VALUES
1529			SECOND WORD OF HEADER INCORRECT
1530	001560	046605	EM306
1531	001562	052162	EM4004
1532	001564	042504	DT003
1533	001566	043066	DF025
1534			ERROR 30: ATTEMPTING TO CHECK HEADER GENERATION
1535			WITH VARIOUS TRACK VALUES
1536			CS1 INCORRECT
1537	001570	046707	EM307
1538	001572	052052	EM4000
1539	001574	042504	DT003
1540	001576	043066	DF025
1541			ERROR 31: ATTEMPTING TO CHECK HEADER GENERATION
1542			WITH VARIOUS TRACK VALUES
1543			FIRST WORD OF HEADER INCORRECT
1544	001600	046707	EM307
1545	001602	052132	EM4003
1546	001604	042504	DT003
1547	001606	043066	DF025

1548	:	ERROR 32: ATTEMPTING TO CHECK HEADER GENERATION
1549	:	WITH VARIOUS TRACK VALUES
1550	:	SECOND WORD OF HEADER INCORRECT
1551	001610	EM307
1552	001612	EM4004
1553	001614	DT003
1554	001616	DF025
1555	:	ERROR 33: ATTEMPTING TO CHECK HEADER GENERATION
1556	:	WITH VARIOUS SECTOR VALUES
1557	:	CS1 INCORRECT
1558	001620	EM308
1559	001622	EM4000
1560	001624	DT003
1561	001626	DF025
1562	:	ERROR 34: ATTEMPTING TO CHECK HEADER GENERATION
1563	:	WITH VARIOUS SECTOR VALUES
1564	:	FIRST WORD OF HEADER INCORRECT
1565	001630	EM308
1566	001632	EM4003
1567	001634	DT003
1568	001636	DF025
1569	:	ERROR 35: ATTEMPTING TO CHECK HEADER GENERATION
1570	:	WITH VARIOUS SECTOR VALUES
1571	:	SECOND WORD OF HEADER INCORRECT
1572	001640	EM308
1573	001642	EM4004
1574	001644	DT003
1575	001646	DF025
1576	:	ERROR 36: ATTEMPTING TO CHECK HEADER GENERATION
1577	:	WITH VARIOUS FORMAT VALUES
1578	:	CS1 INCORRECT
1579	001650	EM309
1580	001652	EM4000
1581	001654	DT003
1582	001656	DF025
1583	:	ERROR 37: ATTEMPTING TO CHECK HEADER GENERATION
1584	:	WITH VARIOUS FORMAT VALUES
1585	:	FIRST WORD OF HEADER INCORRECT
1586	001660	EM309
1587	001662	EM4003
1588	001664	DT003
1589	001666	DF025
1590	:	ERROR 40: ATTEMPTING TO CHECK HEADER GENERATION
1591	:	WITH VARIOUS FORMAT VALUES
1592	:	SECOND WORD OF HEADER INCORRECT
1593	001670	EM309
1594	001672	EM4004
1595	001674	DT003
1596	001676	DF025
1597	:	ERROR 41: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1598	:	CS1 INCORRECT
1599	001700	EM310
1600	001702	EM4000
1601	001704	DT041
1602	001706	DF041
1603	:	ERROR 42: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA

1604			:	CS2 INCORRECT
1605	001710	047206		EM310
1606	001712	052212		EM4005
1607	001714	042524		DT041
1608	001716	043112		DF041
1609			:	ERROR 43: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1610			:	ERROR REGISTER INCORRECT
1611	001720	047206		EM310
1612	001722	052230		EM4006
1613	001724	042524		DT041
1614	001726	043112		DF041
1615			:	ERROR 44: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1616			:	BUS ADD INCORRECT
1617	001730	047206		EM310
1618	001732	052254		EM4007
1619	001734	042524		DT041
1620	001736	043112		DF041
1621			:	ERROR 45: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1622			:	WORD COUNT INCORRECT
1623	001740	047206		EM310
1624	001742	052302		EM4008
1625	001744	042524		DT041
1626	001746	043112		DF041
1627			:	ERROR 46: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1628			:	CS1 INCORRECT AFTER READING DATA BUFFER
1629	001750	047206		EM310
1630	001752	052327		EM4009
1631	001754	042554		DT046
1632	001756	043146		DF046
1633			:	ERROR 47: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1634			:	CS2 INCORRECT AFTER READING DATA BUFFER
1635	001760	047206		EM310
1636	001762	052377		EM4010
1637	001764	042554		DT046
1638	001766	043146		DF046
1639			:	ERROR 50: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1640			:	ERROR REG INCORRECT AFTER READING DATA BUFFER
1641	001770	047206		EM310
1642	001772	052447		EM4011
1643	001774	042554		DT046
1644	001776	043146		DF046
1645			:	ERROR 51: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1646			:	DATA READ FROM MEMORY INCORRECT
1647	002000	047206		EM310
1648	002002	052525		EM4012
1649	002004	042574		DT051
1650	002006	043172		DF051
1651			:	ERROR 52: ATTEMPTING TO CHECK HEADER RECOGNITION
1652			:	CS1 INCORRECT
1653	002010	047273		EM311
1654	002012	052052		EM4000
1655	002014	042606		DT052
1656	002016	043216		DF052
1657			:	ERROR 53: ATTEMPTING TO CHECK HEADER RECOGNITION
1658			:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1659	002020	047273		EM311

H03

1660	002022	052565	EM4013
1661	002024	042622	DT053
1662	002026	043242	DF053
1663	:	:	ERROR 54: ATTEMPTING TO CHECK SECTOR INCREMENT
1664	:	:	DISK ADDRESS REG INCORRECT
1665	002030	047342	EM312
1666	002032	052633	EM4014
1667	002034	042632	DT054
1668	002036	043266	DF054
1669	:	:	ERROR 55: ATTEMPTING TO CHECK SECTOR INCREMENT
1670	:	:	CYLINDER ADDRESS INCORRECT
1671	002040	047342	EM312
1672	002042	052667	EM4015
1673	002044	042632	DT054
1674	002046	043266	DF054
1675	:	:	ERROR 56: ATTEMPTING TO CHECK TRACK INCREMENT
1676	:	:	DISK ADDRESS REG INCORRECT
1677	002050	047407	EM313
1678	002052	052633	EM4014
1679	002054	042632	DT054
1680	002056	043266	DF054
1681	:	:	ERROR 57: ATTEMPTING TO CHECK TRACK INCREMENT
1682	:	:	CYLINDER ADDRESS INCORRECT
1683	002060	047407	EM313
1684	002062	052667	EM4015
1685	002064	042632	DT054
1686	002066	043266	DF054
1687	:	:	ERROR 60: ATTEMPTING TO CHECK CYLINDER INCREMENT
1688	:	:	DISK ADDRESS INCORRECT
1689	002070	047453	EM314
1690	002072	052633	EM4014
1691	002074	042632	DT054
1692	002076	043266	DF054
1693	:	:	ERROR 61: ATTEMPTING TO CHECK CYLINDER INCREMENT
1694	:	:	CYLINDER ADDRESS INCORRECT
1695	002100	047453	EM314
1696	002102	052667	EM4015
1697	002104	042632	DT054
1698	002106	043266	DF054
1699	:	:	ERROR 62: ATTEMPTING TO CHECK SECTOR PULSE DETECTION WITH
1700	:	:	WRITE DATA
1701	:	:	MAINT. REG. 1 INCORRECT
1702	002110	047522	EM315
1703	002112	052727	EM4016
1704	002114	042622	DT053
1705	002116	043242	DF053
1706	:	:	ERROR 63: ATTEMPTING TO FORCE BAD SECTOR ERROR
1707	:	:	CS1 INCORRECT
1708	002120	047616	EM316
1709	002122	052052	EM4000
1710	002124	042554	DT046
1711	002126	043146	DF046
1712	:	:	ERROR 64: ATTEMPTING TO FORCE BAD SECTOR ERROR
1713	:	:	CS2 INCORRECT
1714	002130	047616	EM316
1715	002132	052212	EM4005

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 34
ERROR POINTER TABLE

SEQ 0034

1716	002134	042554	DT046
1717	002136	043146	DF046
1718			ERROR 65: ATTEMPTING TO FORCE BAD SECTOR ERROR
1719			ERROR REG INCORRECT
1720	002140	047616	EM316
1721	002142	052230	EM4006
1722	002144	042554	DT046
1723	002146	043146	DF046
1724			ERROR 66: ATTEMPTING TO FORCE HEADER VRC ERROR
1725			WHEN BAD SECTOR PRESENT
1726			CS1 INCORRECT
1727	002150	047663	EM317
1728	002152	052052	EM4000
1729	002154	042554	DT046
1730	002156	043146	DF046
1731			ERROR 67: ATTEMPTING TO FORCE HEADER VRC ERROR
1732			WHEN BAD SECTOR PRESENT
1733			CS2 INCORRECT
1734	002160	047663	EM317
1735	002162	052212	EM4005
1736	002164	042554	DT046
1737	002166	043146	DF046
1738			ERROR 70: ATTEMPTING TO FORCE HEADER VRC ERROR
1739			WHEN BAD SECTOR PRESENT
1740			ERROR REG INCORRECT
1741	002170	047663	EM317
1742	002172	052230	EM4006
1743	002174	042554	DT046
1744	002176	043146	DF046
1745			ERROR 71: ATTEMPTING TO FORCE HVRC ERROR
1746			CS1 INCORRECT
1747	002200	047761	EM318
1748	002202	052052	EM4000
1749	002204	042646	DT071
1750	002206	043312	DF071
1751			ERROR 72: ATTEMPTING TO FORCE HVRC ERROR
1752			CS2 INCORRECT
1753	002210	047761	EM318
1754	002212	052212	EM4005
1755	002214	042646	DT071
1756	002216	043312	DF071
1757			ERROR 73: ATTEMPTING TO FORCE HVRC ERROR
1758			ERROR REG INCORRECT
1759	002220	047761	EM318
1760	002222	052230	EM4006
1761	002224	042646	DT071
1762	002226	043312	DF071
1763			ERROR 74: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1764			CS1 INCORRECT
1765	002230	050020	EM319
1766	002232	052052	EM4000
1767	002234	042674	DT074
1768	002236	043346	DF074
1769			ERROR 75: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1770			CS2 INCORRECT
1771	002240	050020	EM319

1772	002242	052212	EM4005
1773	002244	042674	DT074
1774	002246	043346	DF074
1775	:	:	ERROR 76: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1776	:	:	ERROR REG INCORRECT
1777	002250	050020	EM319
1778	002252	052230	EM4006
1779	002254	042674	DT074
1780	002256	043346	DF074
1781	:	:	ERROR 77: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1782	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1783	002260	050020	EM319
1784	002262	052565	EM4013
1785	002264	042716	DT077
1786	002266	043372	DF077
1787	:	:	ERROR 100: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1788	:	:	CS1 INCORRECT
1789	002270	050071	EM320
1790	002272	052052	EM4000
1791	002274	042674	DT074
1792	002276	043346	DF074
1793	:	:	ERROR 101: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1794	:	:	CS2 INCORRECT
1795	002300	050071	EM320
1796	002302	052212	EM4005
1797	002304	042674	DT074
1798	002306	043346	DF074
1799	:	:	ERROR 102: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1800	:	:	ERROR REG INCORRECT
1801	002310	050071	EM320
1802	002312	052230	EM4006
1803	002314	042674	DT074
1804	002316	043346	DF074
1805	:	:	ERROR 103: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1806	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1807	002320	050071	EM320
1808	002322	052565	EM4013
1809	002324	042716	DT077
1810	002326	043372	DF077
1811	:	:	ERROR 104: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1812	:	:	CS1 INCORRECT
1813	002330	050155	EM321
1814	002332	052052	EM4000
1815	002334	042674	DT074
1816	002336	043346	DF074
1817	:	:	ERROR 105: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1818	:	:	CS2 INCORRECT
1819	002340	050155	EM321
1820	002342	052212	EM4005
1821	002344	042674	DT074
1822	002346	043346	DF074
1823	:	:	ERROR 106: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1824	:	:	ERROR REG INCORRECT
1825	002350	050155	EM321
1826	002352	052230	EM4006
1827	002354	042674	DT074

K03

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 36
ERROR POINTER TABLE

SEQ 0036

1828	002356	043346	DF074
1829			ERROR 107: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1830			MR1 INCORRECT AFTER GAP IN WRITE DATA
1831	002360	050155	EM321
1832	002362	052565	EM4013
1833	002364	042716	DT077
1834	002366	043372	DF077
1835			ERROR 110: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1836			CS1 INCORRECT
1837	002370	050241	EM322
1838	002372	052052	EM4000
1839	002374	042674	DT074
1840	002376	043346	DF074
1841			ERROR 111: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1842			CS2 INCORRECT
1843	002400	050241	EM322
1844	002402	052212	EM4005
1845	002404	042674	DT074
1846	002406	043346	DF074
1847			ERROR 112: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1848			ERROR REG INCORRECT
1849	002410	050241	EM322
1850	002412	052230	EM4006
1851	002414	042674	DT074
1852	002416	043346	DF074
1853			ERROR 113: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1854			MR1 INCORRECT AFTER GAP IN WRITE DATA
1855	002420	050241	EM322
1856	002422	052565	EM4013
1857	002424	042716	DT077
1858	002426	043372	DF077
1859			ERROR 114: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1860			SECTOR ERROR CS1 INCORRECT
1861	002430	050324	EM323
1862	002432	052052	EM4000
1863	002434	042674	DT074
1864	002436	043346	DF074
1865			ERROR 115: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1866			SECTOR ERROR CS2 INCORRECT
1867	002440	050324	EM323
1868	002442	052212	EM4005
1869	002444	042674	DT074
1870	002446	043346	DF074
1871			ERROR 116: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1872			SECTOR ERROR ERROR INCORRECT
1873	002450	050324	EM323
1874	002452	052230	EM4006
1875	002454	042674	DT074
1876	002456	043346	DF074
1877			ERROR 117: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1878			SECTOR ERROR MR1 INCORRECT
1879	002460	050324	EM323
1880	002462	052565	EM4013
1881	002464	042716	DT077
1882	002466	043372	DF077
1883			ERROR 120: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER

L03

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 37
ERROR POINTER TABLE

SEQ 0037

1884				VRC ERROR CS1 INCORRECT
1885	002470	050417	EM324	
1886	002472	052052	EM4000	
1887	002474	042674	DT074	
1888	002476	043346	DF074	
1889			ERROR 121:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1890				VRC ERROR CS2 INCORRECT
1891	002500	050417	EM324	
1892	002502	052212	EM4005	
1893	002504	042674	DT074	
1894	002506	043346	DF074	
1895			ERROR 122:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1896				VRC ERROR ERROR REG INCORRECT
1897	002510	050417	EM324	
1898	002512	052230	EM4006	
1899	002514	042674	DT074	
1900	002516	043346	DF074	
1901			ERROR 123:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1902				VRC ERROR MR1 INCORRECT
1903	002520	050417	EM324	
1904	002522	052565	EM4013	
1905	002524	042716	DT077	
1906	002526	043372	DF077	
1907			ERROR 124:	FORCING BAD SECTOR ERROR WITH PREV HEADER
1908				VRC ERROR CS1 INCORRECT
1909	002530	050512	EM325	
1910	002532	052052	EM4000	
1911	002534	042674	DT074	
1912	002536	043346	DF074	
1913			ERROR 125:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1914				CS2 INCORRECT
1915	002540	050512	EM325	
1916	002542	052212	EM4005	
1917	002544	042674	DT074	
1918	002546	043346	DF074	
1919			ERROR 126:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1920				ERROR REG INCORRECT
1921	002550	050512	EM325	
1922	002552	052230	EM4006	
1923	002554	042674	DT074	
1924	002556	043346	DF074	
1925			ERROR 127:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1926				MR1 INCORRECT
1927	002560	050512	EM325	
1928	002562	052565	EM4013	
1929	002564	042716	DT077	
1930	002566	043372	DF077	
1931			ERROR 130:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1932				CS1 INCORRECT
1933	002570	050602	EM326	
1934	002572	052052	EM4000	
1935	002574	042674	DT074	
1936	002576	043346	DF074	
1937			ERROR 131:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1938				CS2 INCORRECT
1939	002600	050602	EM326	

1940	002602	052212	EM4005
1941	002604	042674	DT074
1942	002606	043346	DF074
1943	:	:	ERROR 132: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1944	:	:	ERROR REG INCORRECT
1945	002610	050602	EM326
1946	002612	052230	EM4006
1947	002614	042674	DT074
1948	002616	043346	DF074
1949	:	:	ERROR 133: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1950	:	:	MR1 INCORRECT
1951	002620	050602	EM326
1952	002622	052565	EM4013
1953	002624	042716	DT077
1954	002626	043372	DF077
1955	:	:	ERROR 134: ATTEMPTING TO CHECK ECC PATTERN CLEARING
1956	:	:	ECC PATTERN INCORRECT
1957	002630	051021	EM329
1958	002632	052745	EM4017
1959	002634	042730	DT134
1960	002636	043416	DF134
1961	:	:	ERROR 135: ATTEMPTING TO CHECK ECC GENERATION
1962	:	:	ECC PATTERN INCORRECT
1963	002640	051072	EM330
1964	002642	052745	EM4017
1965	002644	042730	DT134
1966	002646	043416	DF134
1967	:	:	ERROR 136: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1968	:	:	CS1 INCORRECT
1969	002650	051135	EM331
1970	002652	052052	EM4000
1971	002654	042554	DT046
1972	002656	043146	DF046
1973	:	:	ERROR 137: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1974	:	:	CS2 INCORRECT
1975	002660	051135	EM331
1976	002662	052212	EM4005
1977	002664	042554	DT046
1978	002666	043146	DF046
1979	:	:	ERROR 140: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1980	:	:	ERROR REG. INCORRECT
1981	002670	051135	EM331
1982	002672	052230	EM4006
1983	002674	042554	DT046
1984	002676	043146	DF046
1985	:	:	ERROR 141: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1986	:	:	CS1 INCORRECT
1987	002700	051222	EM332
1988	002702	052052	EM4000
1989	002704	042554	DT046
1990	002706	043146	DF046
1991	:	:	ERROR 142: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1992	:	:	CS2 INCORRECT
1993	002710	051222	EM332
1994	002712	052212	EM4005
1995	002714	042554	DT046

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 39
ERROR POINTER TABLE

SEQ 0039

1996	002716	043146	DF046
1997			ERROR 143: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1998			ERROR REG. INCORRECT
1999	002720	051222	EM332
2000	002722	052230	EM4006
2001	002724	042554	DT046
2002	002726	043146	DF046
2003			ERROR 144: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2004			CS1 INCORRECT
2005	002730	051403	EM335
2006	002732	052052	EM4000
2007	002734	042742	DT144
2008	002736	043442	DF144
2009			ERROR 145: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2010			CS2 INCORRECT
2011	002740	051403	EM335
2012	002742	052212	EM4005
2013	002744	042742	DT144
2014	002746	043442	DF144
2015			ERROR 146: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2016			ERROR REG. INCORRECT
2017	002750	051403	EM335
2018	002752	052230	EM4006
2019	002754	042742	DT144
2020	002756	043442	DF144
2021			ERROR 147: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2022			BUS ADDRESS INCORRECT
2023	002760	051403	EM335
2024	002762	052254	EM4007
2025	002764	042742	DT144
2026	002766	043442	DF144
2027			ERROR 150: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2028			WORD COUNT INCORRECT
2029	002770	051403	EM335
2030	002772	052302	EM4008
2031	002774	042742	DT144
2032	002776	043442	DF144
2033			ERROR 151: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2034			CYLINDER ADD INCORRECT
2035	003000	051403	EM335
2036	003002	052667	EM4015
2037	003004	042742	DT144
2038	003006	043442	DF144
2039			ERROR 152: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
2040			DISK ADDRESS INCORRECT
2041	003010	051403	EM335
2042	003012	052633	EM4014
2043	003014	042742	DT144
2044	003016	043442	DF144
2045			ERROR 153: ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR
2046			CS1 INCORRECT
2047	003020	051457	EM336
2048	003022	052052	EM4000
2049	003024	042606	DT052
2050	003026	043216	DF052
2051			ERROR 154: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL

2052			:	CS1 INCORRECT
2053	003030	051534	:	EM337
2054	003032	052052	:	EM4000
2055	003034	042742	:	DT144
2056	003036	043442	:	DF144
2057			:	ERROR 155: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2058			:	CS2 INCORRECT
2059	003040	051534	:	EM337
2060	003042	052212	:	EM4005
2061	003044	042742	:	DT144
2062	003046	043442	:	DF144
2063			:	ERROR 156: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2064			:	ERROR REGISTER INCORRECT
2065	003050	051534	:	EM337
2066	003052	052230	:	EM4006
2067	003054	042742	:	DT144
2068	003056	043442	:	DF144
2069			:	ERROR 157: ATTEMPTIN PARTIAL SECTOR WRITE WITH ZERO FILL
2070			:	BUS ADDRESS INCORRECT
2071	003060	051534	:	EM337
2072	003062	052254	:	EM4007
2073	003064	042742	:	DT144
2074	003066	043442	:	DF144
2075			:	ERROR 160: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2076			:	WORD COUNT INCORRECT
2077	003070	051534	:	EM337
2078	003072	052302	:	EM4008
2079	003074	042742	:	DT144
2080	003076	043442	:	DF144
2081			:	ERROR 161: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2082			:	CYLINDER ADDRESS INCORRECT
2083	003100	051534	:	EM337
2084	003102	052667	:	EM4015
2085	003104	042742	:	DT144
2086	003106	043442	:	DF144
2087			:	ERROR 162: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
2088			:	DISK ADDRESS INCORRECT
2089	003110	051534	:	EM337
2090	003112	052633	:	EM4014
2091	003114	042742	:	DT144
2092	003116	043442	:	DF144
2093			:	ERROR 163: ATTEMPTING WRITE DATA IN 18 BIT MODE
2094			:	RKECPT INCORRECT
2095	003120	052005	:	EM340
2096	003122	052745	:	EM4017
2097	003124	043002	:	DT151
2098	003126	043476	:	DF151
2099			:	ERROR 164: ATTEMPTING WRITE DATA IN 18 BIT MODE
2100			:	RKECPT INCORRECT
2101	003130	052005	:	EM340
2102	003132	052745	:	EM4017
2103	003134	042730	:	DT134
2104	003136	043416	:	DF134
2105			:	

```

2106      .SBTTL  TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2107
2108 003140 000000      T.CS1: .WORD 0      ; CONTROL AND STATUS REGISTER 1
2109 003142 000000      T.WC: .WORD 0      ; WORD COUNT REGISTER
2110 003144 000000      T.BA: .WORD 0      ; BUS ADDRESS REGISTER
2111 003146 000000      T.DA: .WORD 0      ; DESIRED TRACK SECTOR REGISTER
2112 003150 000000      T.CS2: .WORD 0      ; CONTROL AND STATUS REGISTER 2
2113 003152 000000      T.DS: .WORD 0      ; DRIVE STATUS REGISTER
2114 003154 000000      T.ER: .WORD 0      ; ERROR REGISTER
2115 003156 000000      T.ASOF: .WORD 0      ; ATTENTION SUMMARY AND OFFSET REGISTER
2116 003160 000000      T.DCYL: .WORD 0      ; DESIRED CYLINDER REGISTER
2117 003162 000000      T.DB: .WORD 0      ; DATA BUFFER
2118 003164 000000      T.MR1: .WORD 0      ; MAINTENANCE REGISTER 1
2119 003166 000000      T.MR2: .WORD 0      ; MAINTENANCE REGISTER 2
2120 003170 000000      T.MR3: .WORD 0      ; MAINTENANCE REGISTER 3
2121 003172 000000      T.ECPS: .WORD 0      ; ECC POSITION INFORMATION
2122 003174 000000      T.ECPT: .WORD 0      ; ECC PATTERN INFORMATION
2123 003176 000000      T.SPARE: .WORD 0      ; SPARE REGISTER
2124
2125      .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS
2126
2127 003200 000000      E.CS1: .WORD 0      ; CONTROL AND STATUS REGISTER 1
2128 003202 000000      E.WC: .WORD 0      ; WORD COUNT REGISTER
2129 003204 000000      E.BA: .WORD 0      ; BUS ADDRESS REGISTER
2130 003206 000000      E.DA: .WORD 0      ; DESIRED TRACK SECTOR REGISTER
2131 003210 000000      E.CS2: .WORD 0      ; CONTROL AND STATUS REGISTER 2
2132 003212 000000      E.DS: .WORD 0      ; DRIVE STATUS REGISTER
2133 003214 000000      E.ER: .WORD 0      ; ERROR REGISTER
2134 003216 000000      E.ASOF: .WORD 0      ; ATTENTION SUMMARY AND OFFSET REGISTER
2135 003220 000000      E.DCYL: .WORD 0      ; DESIRED CYLINDER REGISTER
2136 003222 000000      E.DB: .WORD 0      ; DATA BUFFER
2137 003224 000000      E.MR1: .WORD 0      ; MAINTENANCE REGISTER 1
2138 003226 000000      E.MR2: .WORD 0      ; MAINTENANCE REGISTER 2
2139 003230 000000      E.MR3: .WORD 0      ; MAINTENANCE REGISTER 3
2140 003232 000000      E.ECPS: .WORD 0      ; ECC POSITION INFORMATION
2141 003234 000000      E.ECPT: .WORD 0      ; ECC PATTERN INFORMATION
2142 003236 000000      E.SPARE: .WORD 0      ; SPARE REGISTER

```

2143			.SBTTL	PROGRAM	DEFINED VARIABLES	
2144						
2145	003240	000214	RKVEC:	.WORD	214	;RK611 VECTOR ADDRESS
2146	003242	000240	RKPRI:	.WORD	PR5	;RK611 PRIORITY
2147	003244	000000	TRAPPC:	.WORD	0	;TRAP PC FOR MEMORY PARITY OPTION
2148	003246	000000	SRTFLG:	.WORD	0	;START FLAG
2149						; 0 = 200
2150						; 1 = 214
2151						; -1 = 204
2152	003250	000000	ERRCNT:	.WORD	0	;ERROR COUNT FOR ABORT ERROR THRESHOLD
2153	003252	000000	P1.BIT:	.WORD	0	;NEXT BIT
2154	003254	000000	PR.BIT:	.WORD	0	;PRESENT BIT
2155	003256	000000	M1.BIT:	.WORD	0	;BIT-1
2156	003260	000000	M2.BIT:	.WORD	0	;BIT-2
2157	003262	000000	BITCNT:	.WORD	0	;BIT COUNT
2158	003264	000000	WRDCNT:	.WORD	0	;WORD COUNT FOR NPR TRANSFERS
2159	003266	000000	ECCHI:	.WORD	0	;16 MOST SIGNIFICANT BIT OF ECC
2160	003270	000000	ECCL0:	.WORD	0	;16 LEAST SIGNIFICANT BIT OF ECC
2161	003272	000000	ECCXOR:	.WORD	0	;FLAG FOR ECC GENERATION
2162	003274	000000	MEMPAR:	.WORD	0	;FLAG FOR MEMORY PARITY OPTION
2163	003276	000	SECTOR:	.BYTE	0	;SECTOR FOR INCREMENT TEST
2164	003277	000	TRACK:	.BYTE	0	;TRACK FOR INCREMENT TEST
2165	003300	000000	CYLN:	.WORD	0	;CYLINDER FOR INCREMENT TEST
2166	003302	000000	HEADER:	.WORD	0,0,0	;EXPECTED FOR INCREMENT TEST
2167	003310	000000	HDRCNT:	.WORD	0	;NUMBER OF HEADERS FOR OPI'S
2168	003312	000000	SAVSWR:	.WORD	0	;STORAGE FOR SWITCH REGISTER

```

2169 .SBTTL PROGRAM SETUP
2170
2171 003314 012737 000001 003246 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2172 003322 000406 BR START1
2173
2174 003324 012737 177777 003246 RESTR: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2175 003332 000402 BR START1
2176
2177 003334 005037 003246 START: CLR SRTFLG ;CLEAR START FLAG
2178 003340 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2179 003342 105037 001103 CLR #SERFLG ;CLEAR ERROR FLAG
2180 003346 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2181 003352 004737 040550 JSR PC,STKINT ;INIT KEYBOARD
2182 003356 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2183 003362 012746 003370 MOV #1$,-(SP) ;LOAD START OF PROGRAM
2184 003366 000007 RTI ;LOAD PSW
2185
2186 003370 1$:
2187 .SBTTL INITIALIZE THE COMMON TAGS
2188 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2189 003370 012706 001100 MOV #CMTAG,R6 ;FIRST LOCATION TO BE CLEARED
2190 003374 005026 CLR (R6)+ ;CLEAR MEMORY LOCATION
2191 003376 022706 001140 CMP #SWR,R6 ;;DONE?
2192 003402 001374 BNE -6 ;LOOP BACK IF NO
2193 003404 012706 001100 MOV #STACK,SP ;SETUP THE STACK POINTER
2194 ;;INITIALIZE A FEW VECTORS
2195 003410 012737 034646 000020 MOV #SCOPE,2#IOTVEC ;IOT VECTOR FOR SCOPE ROUTINE
2196 003416 012737 000340 000022 MOV #340,2#IOTVEC+2 ;LEVEL 7
2197 003424 012737 037404 000030 MOV #ERROR,2#EMTVEC ;EMT VECTOR FOR ERROR ROUTINE
2198 003432 012737 000340 000032 MOV #340,2#EMTVEC+2 ;LEVEL 7
2199 003440 012737 042366 000034 MOV #TRAP,2#TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
2200 003446 012737 000340 000036 MOV #340,2#TRAPVEC+2 ;LEVEL 7
2201 003454 012737 042232 000024 MOV #SPWRON,2#PWRVEC ;POWER FAILURE VECTOR
2202 003462 012737 000340 000026 MOV #340,2#PWRVEC+2 ;LEVEL 7
2203 003470 013737 033726 033720 MOV SENDCT,SEOPCT ;SETUP END-OF-PROGRAM COUNTER
2204 003476 005037 001200 CLS TIMES ;INITIALIZE NUMBER OF ITERATIONS
2205 003502 005037 001202 CLS $ESCAPE ;CLEAR THE ESCAPE ON ERROR ADDRESS
2206 003506 112737 000001 001115 MOVB #1,$ERMAX ;ALLOW ONE ERROR PER TEST
2207 003514 012737 003514 001106 MOV #,$LPADR ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2208 003522 012737 003522 001110 MOV #,$LPERR ;SETUP THE ERROR LOOP ADDRESS
2209 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2210 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2211 003530 013746 000004 MOV 2#ERRVEC,-(SP) ;SAVE ERROR VECTOR
2212 003534 012737 003570 000004 MOV #64$,2#ERRVEC ;SET UP ERROR VECTOR
2213 003542 012737 177570 001140 MOV #DSWR,SWR ;SETUP FOR A HARDWARE SWICH REGISTER
2214 003550 012737 177570 001142 MOV #DDISP,DISPLAY ;AND A HARDWARE DISPLAY REGISTER
2215 003556 022777 177777 175354 CMP #-1,2#SWR ;TRY TO REFERENCE HARDWARE SWR
2216 003564 001012 BNE 66$ ;BRANCH IF NO TIMEOUT TRAP OCCURRED
2217 ;AND THE HARDWARE SWR IS NOT = -1
2218 003566 000403 BR 65$ ;BRANCH IF NO TIMEOUT
2219 003570 012716 003576 64$: MOV #65$,(SP) ;SET UP FOR TRAP RETURN
2220 003574 000002 RTI
2221 003576 012737 000176 001140 65$: MOV #SWREG,SWR ;POINT TO SOFTWARE SWR
2222 003604 012737 000174 001142 MOV #DISPREG,DISPLAY
2223 003612 012637 000004 66$: MOV (SP)+,2#ERRVEC ;RESTORE ERROR VECTOR
2224

```

```

2225 003616 005037 001222 CLR $PASS ;: CLEAR PASS COUNT
2226 003622 132737 000200 001235 BITB $APTSIZE,$ENVM ;: TEST USER SIZE UNDER APT
2227 003630 001403 BEQ 67$ ;: YES, USE NON-APT SWITCH
2228 003632 012737 001236 001140 MOV $SWREG,$SWR ;: NO, USE APT SWITCH REGISTER
2229 003640 67$:
2230 003640 005037 003250 CLR EPRCNT
2231 .SBTTL TYPE PROGRAM NAME
2232 ;:TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2233 003644 005227 177777 INC #-1 ;: FIRST TIME?
2234 003650 001055 BNE 68$ ;: BRANCH IF NO
2235 003652 022737 034062 000042 CMP $ENDAD,$#42 ;: ACT-11?
2236 003660 001451 BEQ 68$ ;: BRANCH IF YES
2237 003662 104401 003730 TYPE 69$ ;: TYPE ASCIZ STRING
2238 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2239 003666 005737 000042 TST $#42 ;: ARE WE RUNNING UNDER XXDP/ACT?
2240 003672 001012 BNE 70$ ;: BRANCH IF YES
2241 003674 123727 001234 000001 CMPB $ENV,$#1 ;: ARE WE RUNNING UNDER APT?
2242 003702 001406 BEQ 70$ ;: BRANCH IF YES
2243 003704 023727 001140 000176 CMP $SWR,$SWREG ;: SOFTWARE SWITCH REG SELECTED?
2244 003712 001005 BNE 71$ ;: BRANCH IF NO
2245 003714 104406 GTSWR ;: GET SOFT-SWR SETTINGS
2246 003716 000403 BR 71$
2247 003720 112737 000001 001134 70$: MOVB $#1,$AUTOB ;: SET AUTO-MODE INDICATOR
2248 003726 71$:
2249 003726 000426 BR 68$ ;: GET OVER THE ASCIZ
2250 ;:69$: .ASCIZ <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 4 CZR6DB0/<CRLF>
2251 004004 68$:
2252 004004 005227 177777 INC #-1 ;:TYPE RUN TIME MESSAGE FIRST PASS ONLY
2253 004010 001002 BNE 2$
2254 004012 104401 043671 TYPE ,OPR006
2255 004016 022737 000001 003246 2$: CMP $#1,$RTFLG ;: CHECK IF PARAMETER START
2256 004024 001117 BNE 15$ ;: NO, CONTINUE SETUP
2257 004026 104401 043522 5$: TYPE ,OPR001 ;: TYPE "RK611 BUS ADDRESS ( ) ="
2258 004032 013746 001270 MOV $BASE,-(SP) ;: SAVE $BASE FOR TYPEOUT
2259 004036 104402 TYPOC ;: GO TYPE--OCTAL ASCII(ALL DIGITS)
2260 004040 104401 043551 TYPE ,OPR002
2261 004044 104412 RDOCT ;: GET VALUE
2262 004046 012637 001160 MOV (SP)+,$TMPD
2263 004052 001407 BEQ 7$ ;: CHECK IF <CR>
2264 004054 022737 160000 001160 CMP #160000,$TMPD ;: CHECK IF IN I/O PAGE
2265 004062 101361 BHI 5$
2266 004064 013737 001160 001270 7$: MOV $TMPD,$BASE ;: LOAD NEW BUS ADDRESS
2267 004072 104401 043557 TYPE ,OPR003 ;: TYPE "RK611 VECTOR ADDRESS ( ) ="
2268 004076 013746 001264 MOV $VECT1,-(SP) ;: GET VECTOR ADDRESS FOR TYPE OUT
2269 004102 042716 160000 BIC #160000,(SP) ;: CLEAR PRIORITY BITS
2270 004106 104402 TYPOC
2271 004110 104401 043551 TYPE ,OPR002
2272 004114 104412 RDOCT ;: GET VALUE
2273 004116 012637 001160 MOV (SP)+,$TMPD
2274 004122 001407 BEQ 10$ ;: CHECK IF <CR>
2275 004124 022737 001000 001160 CMP #1000,$TMPD ;: CHECK IF LEGAL
2276 004132 101757 BLOS 7$
2277 004134 013737 001160 001264 10$: MOV $TMPD,$VECT1 ;: LOAD NEW VECTOR ADDRESS
2278 004142 104401 043607 TYPE ,OPR004 ;: TYPE "RK611 PRIORITY ( ) ="
2279 004146 005046 CLR -(SP) ;: MAKE ROOM ON THE STACK
2280 004150 113716 001265 MOVB $VECT1+1,(SP) ;: GET PRIORITY

```

G04

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 45
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0045

2281	004154	006216		ASR	(SP)		;SHIF RIGH 5 BITS
2282	004156	006216		ASR	(SP)		
2283	004160	006216		ASR	(SP)		
2284	004162	006216		ASR	(SP)		
2285	004164	006216		ASR	(SP)		
2286	004166	104402		TYPOC			
2287	004170	104401	043551	TYPE	,OPR002		
2288	004174	104412		RDOCT			;GET VALUE
2289	004176	012637	001160	MOV	(SP)+,STMP0		
2290	004202	001430		BEQ	15\$		;CHECK FOR DEFAULT
2291	004204	022737	000007 001160	CMP	#7,STMP0		;CHECK IF LEGAL
2292	004212	103753		BLO	10\$		
2293	004214	022737	000004 001160	CMP	#4,STMP0		
2294	004222	101347		BHI	10\$		
2295	004224	006337	001160	ASL	STMP0		;SHIFT 5 BITS LEFT
2296	004230	006337	001160	ASL	STMP0		
2297	004234	006337	001160	ASL	STMP0		
2298	004240	006337	001160	ASL	STMP0		
2299	004244	006337	001160	ASL	STMP0		
2300	004250	042737	160000 001264	BIC	#160000,\$VECT1		;GET PRIORITY BITS
2301	004256	153737	001160 001265	BISB	STMP0,\$VECT1+1		
2302	004264	013737	001264 003240	MOV	\$VECT1,RKVEC		
2303	004272	042737	160000 003240	BIC	#160000,RKVEC		;GET VECTOR
2304	004300	113737	001265 003242	MOVB	\$VECT1+1,RKPRI		;GET PRIORITY
2305	004306	013702	001270	MOV	\$BASE,R2		;SET '611 BASE
2306	004312	005037	001202	CLR	\$ESCAPE		;CLEAR ESCAPE
2307	004316	004737	034366	NEWPAS: JSR	PC,PARCHK		;ENABLE MEMORY PARITY
2308	004322	012746	000000	MOV	#PR0,-(SP)		;LOCK OUT INTERRUPTS
2309	004326	012746	004334	MOV	#TST1,-(SP)		
2310	004332	000002		RTI			

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 46
**TYPE C INSTRUCTION'S DRIVE MESSAGES

SEQ 0046

H04

.SBTTL **TYPE C INSTRUCTION'S DRIVE MESSAGES

*TEST 1 READ DATA SEEK MESSAGE

*
* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
* RK06 IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
* 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND
* MAKE SURE IT IS GENERATED PROPERLY.
*

TST1: SCOPE
MOV #50, \$TIMES ; DO 50 ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #-1, RKWC(R2) ; LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV #1777, RKDCYL(R2) ; LOAD CYLINDER ADDRESS REG.
MOV #3400, RKDA(R2) ; LOAD TRACK
MOV #7, RKCS2(R2) ; LOAD DRIVE NUM.
MOV #CDT!CFMT!RDATA, RKCS1(R2) ; ISSUE RDATA WITH CDT SET IN
; 24 SECTOR FORMAT
; CLOCK IN DRIVE MESSAGE
1\$: MOV #3*4+2, R0
MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1\$
MOV RKCS1(R2), T.CS1 ; STORE COMMAND STATUS REG. 1
MOV RKMR2(R2), T.MR2 ; STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2), T.MR3 ; STORE MAINT REG. 3 (MESS B)
MOV #CDT!CFMT!RDATA, E.CS1 ; LOAD EXPECTED CS1
MOV #S.SEEK!S.FMT!70007, E.MR2 ; LOAD EXPECTED MR2
MOV #37760, E.MR3 ; LOAD EXPECTED MR3
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2\$; YES, CHECK MESSAGE A
ERROR 3 ; CS1 INCORRECT
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
BR TST2 ; GO TO NEXT TEST
2\$: CMP E.MR2, T.MR2 ; CHECK MESS A CORRECT
BEQ 3\$; YES, CHECK MESSAGE B
ERROR 4 ; MESS A INCORRECT
3\$: CMP E.MR3, T.MR3 ; CHECK MESS B CORRECT
BEQ TST2 ; YES, GO ON TO NEXT TEST
ERROR 5 ; MESS B INCORRECT

*TEST 2 WRITE DATA SEEK MESSAGE

*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1777,
* TRACK 7, DRIVE 7. CLOCK IN SEEK MESSAGE. CHECK IF
* PROPER MESSAGE IS ASSEMBLED.
*

2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366

```

2367
2368
2369 004566 000004
2370 004570 012737 000062 001200
2371 004576 013702 001270
2372 004602 012762 100000 000000
2373 004610 012762 000040 000026
2374 004616 012762 177777 000002
2375 004624 012762 055216 000004
2376 004632 012762 001777 000020
2377 004640 012762 003400 000006
2378 004646 012762 000007 000010
2379 004654 012762 012023 000000
2380
2381 004662 012700 000016
2382 004666 012762 000440 000026
2383 004674 012762 000040 000026
2384 004702 005300
2385 004704 001370
2386 004706 016237 000000 003140
2387 004714 016237 000034 003166
2388 004722 016237 000036 003170
2389 004730 012737 012023 003200
2390 004736 012737 071227 003226
2391 004744 012737 037760 003230
2392 004752 023737 003200 003140
2393 004760 001405
2394 004762 104006
2395 004764 012762 100000 000000
2396 004772 000412
2397
2398 004774 023737 003226 003166
2399 005002 001401
2400 005004 104007
2401 005006 023737 003230 003170
2402 005014 001401
2403 005016 104010
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416 005020 000004
2417 005022 012737 000050 001200
2418 005030 013702 001270
2419 005034 012762 100000 000000
2420 005042 012762 000040 000026
2421 005050 012762 177777 000002
2422 005056 012762 055216 000004

;*****
;TEST 2: SCOPE
;DO 50. ITERATIONS
;LOAD RK611 BASE
;CLEAR RK611
;PUT RK611 IN DIAGNOSTIC MODE
;LOAD WORD COUNT
;LOAD DUMMY BUS ADDRESS
;LOAD CYLINDER ADDRESS REG.
;LOAD TRACK
;LOAD DRIVE NUM.
;ISSUE WRDATA WITH CDT SET IN
;24 SECTOR FORMAT
;CLOCK IN DRIVE MESSAGE
1$: MOV #3*4+2,R0
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
MOV RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
MOV #CDT!CFMT!WRDATA,E.CS1 ;LOAD EXPECTED CS1
MOV #S.SEEK!S.RTC!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
MOV #37760,E.MR3 ;LOAD EXPECTED MR3
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2$ ;YES, CHECK MESSAGE A
ERROR 6 ;CS1 INCORRECT
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
BR TST3 ;GO TO NEXT TEST

2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
BEQ 3$ ;YES, CHECK MESSAGE B
ERROR 7 ;MESS A INCORRECT
3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
BEQ TST3 ;YES, GO ON TO NEXT TEST
ERROR 10 ;MESS B INCORRECT

;*****
;TEST 3 SEEK MESSAGE FOR WRITE CHECK
;
;CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
;PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE CHECK
;OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
;CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
;IN SEEK COMMAND. MAKE SURE CORRECT MESSAGE
;IS ASSEMBLED.
;*****
;TEST 3: SCOPE
;DO 50 ITERATIONS
;LOAD RK611 BASE
;CLEAR RK611
;PUT RK611 IN DIAGNOSTIC MODE
;LOAD WORD COUNT
;LOAD DUMMY BUS ADDRESS

```


J04

CZR6080 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 48
 CZR608.P11 02-DEC-77 10:00 T3 SEEK MESSAGE FOR WRITE CHECK

SEQ 0048

```

2423 005064 012762 001777 000020      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2424 005072 012762 003400 000006      MOV      #3400,RKDA(R2) ;LOAD TRACK
2425 005100 012762 000007 000010      MOV      #7,RKCS2(R2) ;LOAD DRIVE NUM.
2426 005106 012762 012031 000000      MOV      #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE WRTCHK WITH CDT SET IN
2427                                     ; 24 SECTOR FORMAT
2428 005114 012700 000016                MOV      #3*4+2,R0 ;CLOCK IN DRIVE MESSAGE
2429 005120 012762 000440 000026 1$:    MOV      #DMD!MCLK,RKMR1(R2)
2430 005126 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2431 005134 005300                      DEC      R0
2432 005136 001370                      BNE      1$
2433 005140 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2434 005146 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2435 005154 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2436 005162 012737 012031 003200      MOV      #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2437 005170 012737 071027 003226      MOV      #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2438 005176 012737 037760 003230      MOV      #37760,E.MR3 ;LOAD EXPECTED MR3
2439 005204 023737 003200 003140      CMP      E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2440 005212 001405                      BEQ      2$ ;YES, CHECK MESSAGE A
2441 005214 104011                      ERROR    11 ;CS1 INCORRECT
2442 005216 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2443 005224 000412                      BR       TST4 ;GO TO NEXT TEST
2444
2445 005226 023737 003226 003166 2$:    CMP      E.MR2,T.MR2 ;CHECK MESS A CORRECT
2446 005234 001401                      BEQ      3$ ;YES, CHECK MESSAGE B
2447 005236 104012                      ERROR    12 ;MESS A INCORRECT
2448 005240 023737 003230 003170 3$:    CMP      E.MR3,T.MR3 ;CHECK MESS B CORRECT
2449 005246 001401                      BEQ      TST4 ;YES, GO ON TO NEXT TEST
2450 005250 104013                      ERROR    13 ;MESS B INCORRECT
2451
2452                                     ;*****
2453                                     ;*TEST 4 READ DATA CLEAR MESSAGE
2454                                     ;*
2455                                     ;*
2456                                     ;* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE
2457                                     ;* CONTROLLER ON DIAGNOSTIC MODE. ISSUE A READ DATA TO AN
2458                                     ;* RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE
2459                                     ;* 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND THE
2460                                     ;* CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT.
2461                                     ;*
2462                                     ;*****
2463 ST4: SCOPE
2464 005252 000004                      MOV      #50,$TIMES ;DO 50. ITERATIONS
2465 005254 012737 000062 001200      MOV      $BASE,R2 ;LOAD RK611 BASE
2466 005262 013702 001270                      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2467 005266 012762 100000 000000      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2468 005274 012762 000040 000026      MOV      #-1,RKWC(R2) ;LOAD WORD COUNT
2469 005302 012762 177777 000002      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2470 005310 012762 055216 000004      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2471 005316 012762 001777 000020      MOV      #3400,RKDA(R2) ;LOAD TRACK
2472 005324 012762 003400 000006      MOV      #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2473 005332 012762 000007 000010      MOV      #CDT!CFMT!RDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2474 005340 012762 012021 000000      MOV      ; 24 SECTOR FORMAT
2475 005346 012700 000156                MOV      #27,*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2476 005352 012762 000440 000026 1$:    MOV      #DMD!MCLK,RKMR1(R2)
2477 005360 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2478 005366 005300                      DEC      R0
2479 005370 001370                      BNE      1$

```

K04

CZR6DB0 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 49
 CZR6DB.P11 02-DEC-77 10:00 T4 READ DATA CLEAR MESSAGE

SEQ 0049

```

2479 005372 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2480 005400 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2481 005406 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2482 005414 012737 012021 003200 MOV #CDT!CFMT!RDATA,E.CS1 ;LOAD EXPECTED CS1
2483 005422 012737 071407 003226 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2484 005430 005037 003230 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2485 005434 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2486 005442 001405 BEQ 2$ ;YES, CHECK MESSB
2487 005444 104014 ERROR 14 ;CS1 INCORRECT
2488 005446 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2489 005454 000412 BR TST5 ;GO TO NEXT TEST
2490
2491 005456 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2492 005464 001401 BEQ 3$ ;YES, CHECK MESS B
2493 005466 104015 ERROR 15 ;MESS A INCORRECT
2494 005470 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2495 005476 001401 BEQ TST5 ;YES, GO ON TO NEXT TEST
2496 005500 104016 ERROR 16 ;MESS B INCORRECT
2497
2498 ;*****
2499 ;TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE
2500 ;
2501 ; CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2502 ; CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
2503 ; ONE WORD TO AN RK06 IN 24 SECTOR FORMAT. CYLINDER 1777.
2504 ; TRACK 7. DRIVE 7. CLOCK IN SEEK MESSAGE AND DRIVE CLEAR.
2505 ; CHECK IF PROPER DRIVE CLEAR MESSAGE IS ASSEMBLED.
2506 ;
2507 ;*****
2508 TST5: SCOPE
2509 MOV #50,$TIMES ;DO 50. ITERATIONS
2510 MOV $BASE,R2 ;LOAD RK611 BASE
2511 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2512 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2513 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2514 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2515 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2516 MOV #3400,RKDA(R2) ;LOAD TRACK
2517 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2518 MOV #CDT!CFMT!RDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2519 ; 24 SECTOR FORMAT
2520 MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2521 1$: MOV #DMD!MCLK,RKMR1(R2)
2522 MOV #DMD,RKMR1(R2)
2523 DEC R0
2524 BNE 1$
2525 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2526 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2527 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2528 MOV #CDT!CFMT!RDATA,E.CS1 ;LOAD EXPECTED CS1
2529 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2530 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2531 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2532 BEQ 2$ ;YES, CHECK MESSB
2533 ERROR 17 ;CS1 INCORRECT
2534 MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

L04

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046)
TS02-DEC-77 10:20 PAGE 50
WRITE DATA AND DRIVE CLEAR MESSAGE

SEQ 0050

```

2535 005704 000412 BR TST6 ;;GO TO NEXT TEST
2536
2537 005706 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2538 005714 001401 BEQ 3$ ;YES, CHECK MESS B
2539 005716 104020 ERROR 20 ;MESS A INCORRECT
2540 005720 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2541 005726 001401 BEQ TST6 ;YES, GO ON TO NEXT TEST
2542 005730 104021 ERROR 21 ;MESS B INCORRECT
2543
2544 *****
2545 *TEST 6 DRIVE CLEAR MESSAGE FOR WRITE CHECK
2546 *
2547 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2548 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
2549 * CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
2550 * CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK
2551 * THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR.
2552 * MAKE SURE CORRECT MESSAGE IS ASSEMBLED.
2553 *
2554 *****
2555 TST6: SCOPE
2556 005737 000004 MOV #50,$TIMES ;DO 50. ITERATIONS
2557 005734 012737 000062 001200 MOV $BASE,R2 ;LOAD RK611 BASE
2558 005742 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2559 005746 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2560 005754 012762 000040 000026 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2561 005762 012762 177777 000002 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2562 005770 012762 055216 000004 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2563 005776 012762 001777 000020 MOV #3400,RKDA(R2) ;LOAD TRACK
2564 006004 012762 003400 000706 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2565 006012 012762 000007 000010 MOV #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2566 006020 012762 012031 000000 ;24 SECTOR FORMAT
2567 006026 012700 000156 MOV #27,*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2568 006032 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2569 006040 012762 000040 000026 MOV #DMC,RMR1(R2)
2570 006046 005300 R0
2571 006050 001370 BNE 1$
2572 006052 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2573 006060 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2574 006066 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2575 006074 012737 012031 003200 MOV #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2576 006102 012737 071407 003226 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2577 006110 005037 003230 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2578 006114 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2579 006122 001405 BEQ 2$ ;YES, CHECK MESSB
2580 006124 104022 ERROR 22 ;CS1 INCORRECT
2581 006126 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2582 006134 000412 BR TST7 ;GO TO NEXT TEST
2583
2584 006136 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2585 006144 001401 BEQ 3$ ;YES, CHECK MESS B
2586 006146 104023 ERROR 23 ;MESS A INCORRECT
2587 006150 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2588 006156 001401 BEQ TST7 ;YES, GO ON TO NEXT TEST
2589 006160 104024 ERROR 24 ;MESS B INCORRECT
2590

```

M04

.SBTTL **HEADER GENERATION

*TEST 7 HEADER GENERATION (PART 1)

* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF 1
* WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
* SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
* CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
* TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR CYLINDERS
* 1-1777.

1ST7: SCOPE
MOV #10, \$TIMES ; DO 10. ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
CLR E.DCYL ; INITIALIZE CYLINDER ADD
CLR E.DA ; INITIALIZE TRACK AND SECTOR
MOV #RDDATA, E.CS1 ; INITIALIZE FORMAT
MOV #15, \$LPERR ; LOAD LOOP ON ERROR LOCATION FOR
; SUBTEST LOOP

1\$: JSR PC, CALHDR ; CALCULATE EXPECTED HEADER
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN MAINTENANCE MODE
MOV #-1, RKWC(R2) ; LOAD WORD COUNT
MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV E.DCYL, RKDCYL(R2) ; LOAD CYLINDER NUMBER
MOV E.DA, RKDA(R2) ; LOAD TRACK AND SECTOR
MOV E.CS1, RKCS1(R2) ; LOAD COMMAND AND FORMAT
MOV #69, #4+2, R0 ; LOAD COUNT UNTIL HEADER GENERATION
5\$: MOV #DMD, MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 5\$
MOV RKCS1(R2), T.CS1 ; STORE COMMAND AND STATUS REG 1
MOV RKMR2(R2), T.MR2 ; STORE FIRST HEADER WORD
MOV RKMR3(R2), T.MR3 ; STORE SECOND WORD
CMP E.CS1, T.CS1 ; CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ 6\$; YES, CHECK FIRST LINE OF HEADER
ERROR 25 ; CS1 INCORRECT
MOV #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
BR 15\$; YES, CHECK SECOND WORD OF HEADER
6\$: CMP E.MR2, T.MR2 ; CHECK IF FIRST WORD OF HEADER CORRECT
BEQ 7\$; YES, CHECK IF SECOND WORD OF HEADER CORRECT
ERROR 26 ; FIRST WORD OF HEADER INCORRECT
7\$: CMP E.MR3, T.MR3 ; CHECK IF SECOND WORD OF HEADER CORRECT
BEQ 15\$; YES, CHECK IF LOOP ON ERROR
ERROR 27 ; SECOND WORD INCORRECT
15\$: SCOP1 ; LOOP ON ERROR
INC E.DCYL ; INCREMENT EXPECTED CYLINDER
CMP #633, E.DCYL ; CHECK IF VALUE OF CYLINDER OVERFLOW DETECTION
BNE 16\$; NO, CONTINUE

2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605 006162 000004
2606 006164 012737 000012 001200
2607 006172 013702 001270
2608 006176 005037 003220
2609 006202 005037 003206
2610 006206 012737 000021 003200
2611 006214 012737 006222 001110
2612
2613
2614 006222
2615 006222 004737 036006
2616 006226 012762 100000 000000
2617 006234 012762 000040 000026
2618 006242 012762 177777 000002
2619 006250 012762 055216 000004
2620 006256 013762 003220 000020
2621 006264 013762 003206 000006
2622 006272 013762 003200 000000
2623 006300 012700 000426
2624 006304 012762 000440 000026
2625 006312 012762 000040 000026
2626 006320 005300
2627 006322 001370
2628 006324 016237 000000 003140
2629 006332 016237 000034 003166
2630 006340 016237 000036 003170
2631 006346 023737 003230 003140
2632 006354 001405
2633 006356 104025
2634 006360 012762 100000 000000
2635 006366 000412
2636
2637 006370 023737 003226 003166 6\$:
2638 006376 001401
2639 006400 104026
2640 006402 023737 003230 003170 7\$:
2641 006410 001401
2642 006412 104027
2643 006414 104415 15\$:
2644 006416 005237 003220
2645 006422 022737 000633 003220
2646 006430 001002

N04

CZR6080 PK611 DSKLS CTRL PR14 MACY11 30(1046) 02-DEC-77 10:20 PAGE 52
CZR608.P11 02-DEC-77 10:00 T7 HEADER GENERATION (PART 1)

SEQ 0052

```

2647 006432 005237 003220      INC      E.DCYL      ;USE 634
2648 006436 022737 001777 003220 16$:  CMP      #1777,E.DCYL ;CHECK IF FINISHED
2649 006444 103266      BHIS      1$      ;NO, GO TO NEXT CONFIGURATION
2650
2651 *****
2652 :TEST 10      HEADER GENERATION (PART 2)
2653 :
2654 :      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2655 :      THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
2656 :      1 WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
2657 :      0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGE.
2658 :      CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
2659 :      TO MAKE THAT THE HEADER IS CORRECT. REPEAT FOR HEADS 1-7.
2660 *****
2661 :
2662 :ST10: SCOPE
2663 006446 000004      MOV      #10, $TIMES      ;DO 10. ITERATIONS
2664 006450 012737 000012 001200      MOV      $BASE, R2      ;LOAD RK611 BASE
2665 006456 013702 001270      CLR      E.DCYL      ;INITIALIZE CYLINDER ADD
2666 006462 005037 003220      CLR      E.DA      ;INITIALIZE TRACK AND SECTOR
2667 006466 005037 003206 003200      MOV      #RDATA, E.CS1 ;INITIALIZE FORMAT
2668 006472 012737 000021      MOV      #1$, $LPERA ;LOAD LOOP ON ERROR LOCATION FOR
2669 006500 012737 006506 001110      ; SUBTEST LOOP
2670
2671 006506      1$:      JSR      PC, CALHDR      ;CALULATE EXPECTED HEADER
2672 006506 004737 036006      MOV      #CCLR, RKCS1(R2) ;CLEAR RK611
2673 006512 012762 100000 000000      MOV      #DMD, RKMR1(R2) ;PUT RK611 IM MAINTENANCE MODE
2674 006520 012762 000040 000026      MOV      #-1, AKWC(R2) ;LOAD WORD COUNT
2675 006526 012762 177777 000002      MOV      #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2676 006534 012762 055216 000004      MOV      E.DCYL, RKDCYL(R2) ;LOAD CYLINDER NUMBER
2677 006542 013762 003220 000020      MOV      E.DA, RKDA(R2) ;LOAD TRACK AND SECTOR
2678 006550 013762 003206 000006      MOV      E.CS1, RKCS1(R2) ;LOAD COMMAND AND FORMAT
2679 006556 013762 003200 000000      MOV      #69, *4+2, R0 ;LOAD COUNT UNTIL HEADER GENERATION
2680 006564 012700 000426      5$:      MOV      #DMD, MCLK, RKMR1(R2)
2681 006570 012762 000440 000026      MOV      #DMD, RKMR1(R2)
2682 006576 012762 000040 000026      DEC      R0
2683 006604 005300      BNE      5$
2684 006606 001370      MOV      RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG 1
2685 006610 016237 000000 003140      MOV      RKMR2(R2), T.MR2 ;STORE FIRST HEADER WORD
2686 006616 016237 000034 003166      MOV      RKMR3(R2), T.MR3 ;STORE SECOND WORD
2687 006624 016237 000036 003170      CMP      E.CS1, T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2688 006632 023737 003200 003140      BEQ      6$ ;YES, CHECK FIRST LINE OF HEADER
2689 006640 001405      ERROR 30 ;CS1 INCORRECT
2690 006642 104030      MOV      #CCLR, RKCS1(R2) ;CLEAR CONTROLLER
2691 006644 012762 100000 000000      BR      15$ ;YES, CHECK SECOND WORD OF HEADER
2692 006652 000412
2693
2694 006654 023737 003226 003166 6$:  CMP      E.MR2, T.MR2 ;CHECK IF FIRST WORD OF HEADER CORRECT
2695 006662 001401      BEQ      7$ ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
2696 006664 104031      ERROR 31 ;FIRST WORD OF HEADER INCORRECT
2697 006666 023737 003230 003170 7$:  CMP      E.MR3, T.MR3 ;CHECK IF SECOND WORD OF HEADER CORRECT
2698 006674 001401      BEQ      15$ ;YES, CHECK IF LOOP ON ERROR
2699 006676 104032      ERROR 32 ;SECOND WORD INCORRECT
2700 006700 104415      15$:  SCOP1 ;LOOP ON ERROR
2701 006702 105237 003207      INCB      E.DA+1 ;INCREMENT TRACK
2702 006706 122737 000007 003207      CMPB      #7, E.DA+1 ;CHECK IF FINISHED

```

2703 006714 103274

BHIS 1\$;NO, GO TO NEXT CONFIGURATION

2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758

006716 000004
006720 012737 000012 001200
006726 013702 001270
006732 005037 003220
006736 005037 003206
006742 012737 000021 003200
006750 012737 006756 001110

006756
006756 004737 036006
006762 012762 100000 000000
006770 012762 000040 000026
006776 012762 177777 000002
007004 012762 055216 000004
007012 013762 003220 000020
007020 013762 003206 000006
007026 013762 003200 000000
007034 012700 000426
007040 012762 000440 000026
007046 012762 000040 000026
007054 005300
007056 001370
007060 016237 000000 003140
007066 016237 000034 003166
007074 016237 000036 003170
007102 023737 003200 003140
007110 001405
007112 104033
007114 012762 100000 000000
007122 000412

007124 023737 003226 003166
007132 001401
007134 104034
007136 023737 003230 003170
007144 001401
007146 104035
007150 104415
007152 105237 003206
007156 122737 000025 003206
007164 103274

```

*****
*TEST 11      HEADER GENERATION (PART 3)
*
*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
*      THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
*      ONE WORD FOR AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD
*      0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES.
*      CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE REGISTER 3
*      TO MAKE SURE HEADER IS CORRECT. REPEAT FOR SECTORS 1-25.
*****
*ST11: SCOPE
*      MOV      #10, $TIMES      ;DO 10. ITERATIONS
*      MOV      $BASE, R2        ;LOAD RK611 BASE
*      CLR      E.DCYL          ;INITIALIZE CYLINDER ADD
*      CLR      E.DA            ;INITIALIZE TRACK AND SECTOR
*      MOV      #RDDATA, E.CS1   ;INITIALIZE FORMAT
*      MOV      #1$, $LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
*                               ; SUBTEST LOOP
*
*1$:
*      JSR      PC, CALHDR       ;CALULATE EXPECTED HEADER
*      MOV      #CCLR, RKCS1(R2) ;CLEAR RK611
*      MOV      #DMD, RKMR1(R2)  ;PUT RK611 IN MAINTENANCE MODE
*      MOV      #-1, RKWC(R2)   ;LOAD WORD COUNT
*      MOV      #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
*      MOV      E.DCYL, RKDCYL(R2) ;LOAD CYLINDER NUMBER
*      MOV      E.DA, RKDA(R2)  ;LOAD TRACK AND SECTOR
*      MOV      E.CS1, RKCS1(R2) ;LOAD COMMAND AND FORMAT
*      MOV      #69, #4+2, R0    ;LOAD COUNT UNTIL HEADER GENERATION
*
*5$:
*      MOV      #DMD, MCLK, RKMR1(R2)
*      MOV      #DMD, RKMR1(R2)
*      DEC      R0
*      BNE      5$
*      MOV      RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG 1
*      MOV      RKMR2(R2), T.MR2 ;STORE FIRST HEADER WORD
*      MOV      RKMR3(R2), T.MR3 ;STORE SECOND WORD
*      CMP      E.CS1, T.CS1     ;CHECK COMMAND AND STATUS REG 1 CORRECT
*      BEQ      6$              ;YES, CHECK FIRST LINE OF HEADER
*      ERROR   33               ;CS1 INCORRECT
*      MOV      #CCLR, RKCS1(R2) ;CLEAR CONTROLLER
*      BR       15$             ;YES, CHECK SECOND WORD OF HEADER
*
*6$:
*      CMP      E.MR2, T.MR2     ;CHECK IF FIRST WORD OF HEADER CORRECT
*      BEQ      7$              ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
*      ERROR   34               ;FIRST WORD OF HEADER INCORRECT
*
*7$:
*      CMP      E.MR3, T.MR3     ;CHECK IF SECOND WORD OF HEADER CORRECT
*      BEQ      15$             ;YES, CHECK IF LOOP ON ERROR
*      ERROR   35               ;SECOND WORD INCORRECT
*
*15$:
*      SCOP1
*      INCB    E.DA             ;INCREMENT SECTOR
*      CMPB    #25, E.DA        ;CHECK IF FINISHED
*      BHIS    1$              ;NO, GO TO NEXT CONFIGURATION

```


2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814

007166	000004		
007170	012737	000062	001200
007176	013702	001270	
007202	005037	003220	
007206	005037	003206	
007212	012737	000021	003200
007220	012737	007226	001110
007226			
007226	004737	036006	
007232	012762	100000	000000
007240	012762	000040	000026
007246	012762	177777	000002
007254	012762	055216	000004
007262	013762	003220	000020
007270	013762	003206	000006
007276	013762	003200	000000
007304	012700	000426	
007310	012762	000440	000026
007316	012762	000040	000026
007324	005300		
007326	001370		
007330	016237	000000	003140
007336	016237	000034	003166
007344	016237	000036	003170
007352	023737	003200	003140
007360	001405		
007362	104036		
007364	012762	100000	000000
007372	000412		
007374	023737	003226	003166
007402	001401		
007404	104037		
007406	023737	003230	003170
007414	001401		
007416	104040		
007420	104415		
007422	032737	010000	003200
007430	001004		
007432	052737	010000	003200
007440	000672		

```

*****
*TEST 12      HEADER GENERATION (PART 4)
*
*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
*      THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA OF
*      ONE WORD FOR AND RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
*      HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK AND DRIVE CLEAR
*      MESSAGES, CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE
*      REGISTER 3 TO MAKE SURE HEADER IS CORRECT.  REPEAT FOR 24
*      SECTOR FORMAT.
*****
TST12:  SCOPE
        MOV     #50, $TIMES      ; DO 50. ITERATIONS
        MOV     $BASE, R2       ; LOAD RK611 BASE
        CLR     E.DCYL          ; INITIALIZE CYLINDER ADD
        CLR     E.DA            ; INITIALIZE TRACK AND SECTOR
        MOV     #RDDATA, E.CS1  ; INITIALIZE FORMAT
        MOV     #1$, $LPERR     ; LOAD LOOP ON ERROR LOCATION FOR
                                ; SUBTEST LOOP

1$:     JSR     PC, CALHDR       ; CALCULATE EXPECTED HEADER
        MOV     #CCLR, RKCS1(R2) ; CLEAR RK611
        MOV     #DMD, RKMR1(R2)  ; PUT RK611 IN MAINTENANCE MODE
        MOV     #-1, RKWC(R2)   ; LOAD WORD COUNT
        MOV     #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
        MOV     E.DCYL, RKDCYL(R2) ; LOAD CYLINDER NUMBER
        MOV     E.DA, RKDA(R2)  ; LOAD TRACK AND SECTOR
        MOV     E.CS1, RKCS1(R2) ; LOAD COMMAND AND FORMAT
        MOV     #69, *4+2, R0    ; LOAD COUNT UNTIL HEADER GENERATION
5$:     MOV     #DMD, MCLK, RKMR1(R2)
        MOV     #DMD, RKMR1(R2)
        DEC     R0
        BNE     5$
        MOV     RKCS1(R2), T.CS1 ; STORE COMMAND AND STATUS REG 1
        MOV     RKMR2(R2), T.MR2 ; STORE FIRST HEADER WORD
        MOV     RKMR3(R2), T.MR3 ; STORE SECOND WORD
        CMP     E.CS1, T.CS1     ; CHECK COMMAND AND STATUS REG 1 CORRECT
        BEQ     6$              ; YES, CHECK FIRST LINE OF HEADER
        ERROR   36              ; CS1 INCORRECT
        MOV     #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
        BR      15$             ; YES, CHECK SECOND WORD OF HEADER

6$:     CMP     E.MR2, T.MR2     ; CHECK IF FIRST WORD OF HEADER CORRECT
        BEQ     7$              ; YES, CHECK IF SECOND WORD OF HEADER CORRECT
        ERROR   37              ; FIRST WORD OF HEADER INCORRECT
7$:     CMP     E.MR3, T.MR3     ; CHECK IF SECOND WORD OF HEADER CORRECT
        BEQ     15$             ; YES, CHECK IF LOOP ON ERROR
        ERROR   40              ; SECOND WORD INCORRECT
15$:    SCOP1                    ; LOOP ON ERROR
        BIT     #CFMT, E.CS1     ; CHECK IF FINISHED
        BNE     TST13           ; YES, GO TO NEXT TEST
        BIS     #CFMT, E.CS1     ; USE 24 SECTOR FORMAT
        BR      1$

```


.SBTTL **NPR TRANSFER FOR WRITE DATA

```
*****
*TEST 13      WRITE DATA NPR TRANSFER
*
*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
*      CONTROLLER IN MAINTENANCE MODE.  ISSUE A WRITE DATA OF
*      67 WORDS, TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
*      TRACK 0, DRIVE 0.  CLOCK IN SEEK AND DRIVE CLEAR MESSAGES.
*      GIVE ENOUGH CLOCK PULSE FOR 68 SILO WORDS.  MAKE SURE DATA
*      LATE DOES NOT OCCUR.  READ BACK 66 WORDS AND VERIFY THEY
*      ARE CORRECT.
*****
```

```
*****
TST13:  SCOPE
MOV     #50, $TIMES      ;DO 50. ITERATIONS
MOV     $BASE, R2        ;LOAD RK611 BASE
MOV     #CCLR, RKCS1(R2) ;CLEAR RK611
MOV     #DMD, RKMR1(R2)  ;PUT RK611 IN DIAGNOSTIC MODE
MOV     #-67, RKWC(R2)   ;WORD COUNT=67
MOV     #NPRBUF, RKBA(R2) ;LOAD BUFFER ADDRESS
MOV     #WRDATA, RKCS1(R2) ;ISSUE WRDATA
MOV     #68.*37, R0      ;ISSUE ENOUGH CLOCKS FOR 68
1$:     MOV     #DMD!MCLK, RKMR1(R2) ; NPR TRANSFERS
        MOV     #DMD, RKMR1(R2)
        DEC     R0
        BNE     1$
        MOV     RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG. 1
        MOV     RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG. 2
        MOV     RKWC(R2), T.WC   ;STORE WORD COUNT REG.
        MOV     RKBA(R2), T.BA   ;STORE BUS ADDRESS
        MOV     RKR(R2), T.ER    ;STORE ERROR REG.
        MOV     #WRDATA, E.CS1  ;LOAD EXPECTED CS1
        MOV     #OR, E.CS2      ;LOAD EXPECTED CS2
        MOV     #-1, E.WC       ;LOAD EXPECTED WORD COUNT
        MOV     #NPRBUF+(66.*2), E.BA ;LOAD EXPECTED BUS ADDRESS
        CLR     E.ER            ;LOAD EXPECTED ERROR REG
        CMP     E.CS1, T.CS1    ;CHECK CS1 CORRECT
        BEQ     5$              ;YES, CHECK CS2
        ERROR   41              ;CS1 INCORRECT
        MOV     #CCLR, RKCS1(R2) ;CLEAR RK611
        BR      TST14          ;GO TO NEXT TEST

2858:   CMP     E.CS2, T.CS2    ;CHECK CS2
        BEQ     6$              ;NO, CHECK BUS ADDRESS
        ERROR   42              ;CS2 INCORRECT
2861:   CMP     E.BA, T.BA      ;CHECK BUS ADDRESS
        BEQ     7$              ;YES, CHECK WORD COUNT
        ERROR   44              ;BUS ADDRESS INCORRECT
2864:   CMP     E.WC, T.WC      ;CHECK WORD COUNT
        BEQ     8$              ;YES, CONTINUE
        ERROR   45              ;WORD COUNT INCORRECT
2867:   CMP     E.ER, T.ER      ;CHECK ERROR REG CORRECT
        BEQ     10$             ;YES, CONTINUE
        ERROR   43              ;ERROR REG INCORRECT
2870:   MOV     #NPRBUF, R3     ;LOAD ADDRESS OR START OF BUFFER
```

```
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829 007442 000004
2830 007444 012737 000062 001200
2831 007452 013702 001270
2832 007456 012762 100000 000000
2833 007464 012762 000040 000026
2834 007472 012762 177675 000002
2835 007500 012762 053154 000004
2836 007506 012762 000023 000000
2837 007514 012700 004724
2838 007520 012762 000440 000026
2839 007526 012762 000040 000026
2840 007534 005300
2841 007536 001370
2842 007540 016237 000000 003140
2843 007546 016237 000010 003150
2844 007554 016237 000002 003142
2845 007562 016237 000004 003144
2846 007570 016237 000014 003154
2847 007576 012737 000023 003200
2848 007604 012737 000200 003210
2849 007612 012737 177777 003202
2850 007620 012737 053360 003204
2851 007626 005037 003214
2852 007632 023737 003200 003140
2853 007640 001405
2854 007642 104041
2855 007644 012762 100000 000000
2856 007652 000517
2857
2858 007654 023737 003210 003150 5$:
2859 007662 001401
2860 007664 104042
2861 007666 023737 003204 003144 6$:
2862 007674 001401
2863 007676 104044
2864 007700 023737 003202 003142 7$:
2865 007706 001401
2866 007710 104045
2867 007712 023737 003214 003154 8$:
2868 007720 001401
2869 007722 104043
2870 007724 012703 053154 10$:
```



```

2871 007730 012701 000101      MOV      #65,R1      ;LOAD COUNT FOR SICO
2872 007734 005037 003264      CLR      WRDCNT     ;INITIALIZE WORD COUNT
2873 007740 012737 000300 003210  MOV      #IR!OR,E.CS2  ;LOAD EXPECTED CS2
2874 007746 012337 003222      MOV      (R3)+,E.DB    ;LOAD EXPECTED DATA
2875 007752 016237 000024 003162  MOV      RKDB(R2),T.DB  ;GET DATA READ
2876 007760 012700 000020      MOV      #20,R0      ;SET COUNT TO WAIT FOR SILO
2877 007764 005300      16$: DEC      R0          ;DEC COUNT
2878 007766 001376      BNE      16$          ;WAIT FOR 0
2879 007770 016237 000000 003140  MOV      RKCS1(R2),T.CS1 ;STORE CS1
2880 007776 016237 000010 003150  MOV      RKCS2(R2),T.CS2 ;STORE CS2
2881 010004 016237 000014 003154  MOV      RKER(R2),T.ER   ;STORE ERROR REG.
2882 010012 023737 003200 003140  CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
2883 010020 001405      BEQ      20$          ;YES, CONTINUE
2884 010022 104046      ERROR    46          ;CS1 INCORRECT
2885 010024 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2886 010032 000427      BR       TST14        ;GO ON TO NEXT TEST
2887
2888 010034 023737 003210 003150 20$: CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
2889 010042 001401      BEQ      21$          ;YES, CONTINUE
2890 010044 104047      ERROR    47          ;CS2 INCORRECT
2891 010046 023737 003214 003154 21$: CMP      E.ER,T.ER     ;CHECK ERROR REG CORRECT
2892 010054 001401      BEQ      22$          ;YES, CONTINUE
2893 010056 104050      ERROR    50          ;ERROR REG INCORRECT
2894 010060 023737 003222 003162 22$: CMP      E.DB,T.DB     ;CHECK DATA CORRECT
2895 010066 001401      BEQ      25$          ;YES, CONTINUE
2896 010070 104051      ERROR    51          ;DATA INCORRECT
2897 010072 005237 003264      25$: INC      WRDCNT     ;INCREMENT WORD COUNT
2898 010076 005301      DEC      R1
2899 010100 001003      BNE      26$          ;CHECK IF LAST WORD
2900 010102 012737 000100 003210  MOV      #IR,E.CS2    ;UPDATE EXPECTED CS2
2901 010110 100316      26$: BPL      15$          ;CHECK IF FINISHED
2902
2903      .SBTTL  **HEADER RECOGNITION TESTS
2904
2905      ;*****
2906      ;*TEST 14      WRITE DATA HEADER RECOGNITION
2907      ;*
2908      ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2909      ;*      CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE
2910      ;*      WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
2911      ;*      SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
2912      ;*      SIMULATE A SECTOR PULSE AND A HEADER WITH THE FOLLOWING
2913      ;*      DATA:
2914      ;*
2915      ;*      000000
2916      ;*      140000
2917      ;*      140000
2918      ;*
2919      ;*      MAKE SURE WRITE GATE SETS SHOWING CORRECT HEADER RECOGNITION.
2920      ;*
2921      ;*****
2922 010112 000004      TST14: SCOPE
2923 010114 012737 000012 001200  MOV      #10,$TIMES    ;DO 10. ITERATIONS
2924 010122 013702 001270      MOV      $BASE,R2      ;LOAD RK611 BASE
2925 010126 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2926 010134 012762 000040 000026  MOV      #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE

```

F05

CZP6080 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 57
 CZP608.P11 02-DEC-77 10:00 T14 WRITE DATA HEADER RECOGNITION

SEQ 0057

2927	010142	012762	177777	000002		MOV	#-1,RKWC(R2)	;WORD COUNT=1
2928	010150	012762	055216	000004		MOV	#BUFF,RKBA(R2)	;LOAD DUMMY BUS ADDRESS
2929	010156	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA
2930	010164	012700	000426			MOV	#69.*4+2,R0	;ISSUE ENOUGH CLOCKS UNIT READY
2931								;FOR SECTOR PULSE
2932	010170	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
2933	010176	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2934	010204	005300				DEC	R0	
2935	010206	001370				BNE	1\$	
2936	010210	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
2937	010216	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
2938	010224	012737	000023	003200		MOV	#WRDATA,E.CS1	;STORE EXPECTED CS1
2939	010232	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
2940	010236	005037	003256			CLR	M1.BIT	
2941	010242	005037	003262			CLR	BITCNT	
2942	010246	012700	000377			MOV	#255,R0	
2943	010252	004737	037026		5\$:	JSR	PC,R0BIT	
2944	010256	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2945	010264	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2946	010272	001112				BNE	63\$	;NO, REPORT ERROR
2947	010274	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2948	010300	005300				DEC	R0	;CHECK IF SYNCH FINISHED
2949	010302	001363				BNE	5\$	
2950	010304	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
2951	010312	004737	037026			JSR	PC,R0BIT	
2952	010316	016237	000000	003140		MOV	RKCS1(R2),T.CS1	
2953	010324	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2954	010332	001072				BNE	63\$	;NO, REPORT CS1
2955	010334	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2956	010340	012703	053364			MOV	#HEAD1,R3	;SIMULATE GOOD HEADER
2957	010344	012701	000003			MOV	#3,R1	
2958	010350	012304			12\$:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
2959	010352	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
2960	010356	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
2961	010364	006004				ROR	R4	;GET NEXT BIT
2962	010366	103403				BCS	17\$	
2963	010370	005037	003254			CLR	PR.BIT	
2964	010374	000403				BR	18\$	
2965								
2966	010376	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
2967	010404	004737	037026		18\$:	JSR	PC,R0BIT	;SIMULATE NEXT BIT
2968	010410	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
2969	010416	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
2970	010424	001035				BNE	63\$	;NO, REPORT ERROR
2971	010426	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
2972	010432	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
2973	010434	001350				BNE	15\$	;NO, CONTINUE
2974	010436	005301				DEC	R1	;CHECK IF FINISHED WITH HEAD
2975	010440	001343				BNE	12\$	;NO, CONTINUE
2976	010442	012700	000105			MOV	#69,R0	;LOAD COUNT FOR GAP
2977	010446	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
2978	010454	005037	003254			CLR	PR.BIT	
2979	010460	004737	037026			JSR	PC,R0BIT	
2980	010464	005300				DEC	R0	
2981	010466	001367				BNE	25\$	
2982	010470	016237	000026	003164		MOV	RKMR1(R2),T.MR1	;GET MR1

G05

CZR6080 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 58
 CZR608.P11 02-DEC-77 10:00 T14 WRITE DATA HEADER RECOGNITION

SEQ 0058

```

2983 010476 012737 062040 003224      MOV      #DMD!ECCW!1,LD!WRTGAT,E.MR1 ;LOAD EXPECTED MR1
2984 010504 023737 003224 003164      CMP      E.MR1,T.MR1 ;CHECK IF WRITE GATE SET
2985 010512 001411                BEQ      TST15 ;;;YES, GO ON TO NEXT TEST
2986 010514 104053                ERROR    S3
2987 010516 000407                BR       TST15 ;;GO ON TO NEXT TEST
2988
2989 010520 016237 000010 003150 63$:    MOV      RKCS2(R2),T.CS2 ;STORE CS2 FOR PRINT OUT
2990 010526 016237 000014 003154      MOV      RKER(R2),T.ER ;STORE ERROR REG FOR PRINT OUT
2991 010534 104052                ERROR    S2 ;REPORT ERROR
2992
2993      ;*****
2994      *TEST 15      SECTOR PULSE DETECTION FOR WRITE DATA
2995      *
2996      *      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2997      *      CONTROLLER IN MAINTENANCE MODE ISSUE A WRITE DATA OF ONE
2998      *      WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
2999      *      SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
3000      *      SIMULATE AN INDEX PULSE AND A HEADER WITH THE FOLLOWING
3001      *      DATA:
3002      *
3003      *      000000
3004      *      140000
3005      *      140000
3006      *
3007      *      MAKE SURE WRITE GATE DOES NOT SET.
3008      *
3009      ;*****
3010      *ST15:      SCOPE
3011      *      MOV      #10,S.TIMES ;DO 10. ITERATIONS
3012      *      MOV      $BASE,R2 ;LOAD RK611 BASE
3013      *      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3014      *      MOV      #700,R0 ;SET STALL COUNT
3015      *      DEC      R0 ;DEC COUNT
3016      *      BNE      4$ ;LOOP UNTIL ZERO
3017      *      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3018      *      MOV      #-1,RKWC(R2) ;WORD COUNT = 1
3019      *      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3020      *      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3021      *      MOV      #69,*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTILL READY
3022      *      ;FOR SECTOR PULSE
3023      *      MOV      #DMD!MCLK,RKMR1(R2)
3024      *      MOV      #DMD,RKMR1(R2)
3025      *      DEC      R0
3026      *      BNE      1$
3027      *      MOV      #4,R0 ;SIMULATE INDEX PULSE
3028      *      MOV      #DMD!MIND,RKMR1(R2)
3029      *      MOV      #DMD!MIND!MCLK,RKMR1(R2)
3030      *      MOV      #DMD!MIND,RKMR1(R2)
3031      *      DEC      R0
3032      *      BNE      2$
3033      *      MOV      #DMD,RKMR1(R2)
3034      *      CLR      PR.BIT ;GENERATE SYNCH
3035      *      CLR      M1.BIT
3036      *      MOV      #255,R0
3037      *      JSR      PC,RDBIT
3038      *      DEC      R0

```

H05

CZP6080 RK611 DSKLS CTRL PRTH
CZP608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 59
T15 SECTOR PULSE DETECTION FOR WRITE DATA

SEQ 0059

```

3039 010726 001374      BNE      5$
3040 010730 012737 000001 003254      MOV      #1,PR.BIT      ;SIMULATE SYNCH BIT
3041 010736 004737 037026      JSR      PC,RDBIT
3042 010742 012703 053364      MOV      #HEAD1,R3      ;SIMULATE GOOD HEADER
3043 010746 012701 000003      MOV      #3,R1
3044 010752 012304      12$:      MOV      (P3)+,R4      ;GET NEXT HEADER WORD
3045 010754 012700 000020      MOV      #16,R0      ;LOAD BITS PER WORDS
3046 010760 013737 003254 003256 15$:      MOV      PR.BIT,M1.BIT      ;STORE PREVIOUS BIT
3047 010766 006004      ROR      R4      ;GET NEXT BIT
3048 010770 103403      BCS      17$
3049 010772 005037 003254      CLR      PR.BIT
3050 010776 000403      BR      18$
3051
3052 011000 012737 000001 003254 17$:      MOV      #1,PR.BIT
3053 011006 004737 037026 18$:      JSR      PC,RDBIT      ;SIMULATE NEXT BIT
3054 011012 005300      DEC      R0      ;CHECK IF READY FOR NEXT HEADER WORD
3055 011014 001361      BNE      15$      ;NO, CONTINUE
3056 011016 005301      DEC      R1      ;CHECK IF FINISHED WITH HEADER
3057 011020 001354      BNE      12$      ;NO, CONTINUE
3058 011022 012700 000100      MOV      #64,R0      ;LOAD COUNT FOR GAP
3059 011026 013737 003254 003256 25$:      MOV      PR.BIT,M1.BIT      ;SIMULATE GAP
3060 011034 005037 003254      CLR      PR.BIT
3061 011040 004737 037026      JSR      PC,RDBIT
3062 011044 005300      DEC      R0
3063 011046 001367      BNE      25$
3064 011050 016237 000026 003164      MOV      RKMRI(R2),T.MRI ;GET MRI
3065 011056 012737 022040 003224      MOV      #DMD!ECCW!MEWD,E.MRI ;LOAD EXPECTDED MRI
3066 011064 023737 003164 003224      CMP      T.MRI,E.MRI      ;CHECK IF WRITE GATE DID NOT SET
3067 011072 001401      BEQ      TST16      ;;YES, GO ON TO NEXT TEST
3068 011074 104053      ERROR      53
3069
3070 *****
3071 *TEST 16      SECTOR INCREMENT
3072 *
3073 *      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3074 *      CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD
3075 *      TO AN RK06 IN 26 SECTOR FORMAT. CYLINDER 0, HEAD 0.
3076 *      SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
3077 *      SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
3078 *      THAT WRITE GATE SETS,
3079 *
3080 *      REPEAT FOR SECTOR 1-24.
3081 *
3082 *****
3083 *ST16:      SCOPE
3084 011076 000004      MOV      #10,$TIMES      ;DO 10. ITERATIONS
3085 011100 012737 000012 001200      MOV      $BASE,R2      ;LOAD RK611 BASE
3086 011106 013702 001270      CLR      CYLN      ;INITIALIZE CYLINDER
3087 011112 005037 003300      CLR      SECTOR      ;INITIALIZE TRACK AND SECTOR
3088 011116 005037 003276      MOV      #1,$LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
3089 011122 012737 011130 001110      ;      SUBTEST LOOP
3090
3091 011130      1$:      JSR      PC,INCHDR      ;GENERATE HEADER WORDS
3092 011130 004737 035624      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3093 011134 012762 100000 000000      MOV      #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3094 011142 012762 000040 000026

```

Address	Hex	Hex	Hex	Hex	Label	Assembly	Comments
3095	011150	013762	003300	000020		MOV	CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3096	011156	013762	003276	000006		MOV	SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3097	011164	012762	177777	000002		MOV	#-1,RKWC(R2) ;WORD COUNT=1
3098	011172	012762	05521E	000004		MOV	#BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3099	011200	012762	000023	000000		MOV	#WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3100	011206	012700	070426			MOV	#69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY FOR SECTOR PULSE
3101							
3102	011212	012762	000440	000026	5\$:	MOV	#DMD:MCLK,RKMR1(R2)
3103	011220	012762	000040	000026		MOV	#DMD,RKMR1(R2)
3104	011226	005300				DEC	R0
3105	011230	001370				BNE	5\$
3106	011232	012762	000140	000026		MOV	#DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3107	011240	012762	000040	000026		MOV	#DMD,RKMR1(R2)
3108	011246	005037	003254			CLR	PR.BIT ;GENERATE SYNCH
3109	011252	005037	003256			CLR	M1.BIT
3110	011256	012700	000377			MOV	#255,R0
3111	011262	004737	037026		10\$:	JSR	PC,RDBIT
3112	011266	005300				DEC	R0
3113	011270	001374				BNE	10\$
3114	011272	012737	000001	003254		MOV	#1,PR.BIT ;SIMULATE SYNCH BIT
3115	011300	004737	037026			JSR	PC,RDBIT
3116	011304	012703	003302			MOV	#HEADER,R3 ;SIMULATE HEADER
3117	011310	012701	000003			MOV	#3,R1
3118	011314	012304			12\$:	MOV	(R3)+,R4 ;GET NEXT HEADER WORD
3119	011316	012700	000020			MOV	#16,R0 ;LOAD BITS PER WORD
3120	011322	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT
3121	011330	006004				ROR	R4
3122	011332	103403				BCS	17\$
3123	011334	005037	003254			CLR	PR.BIT
3124	011340	000403				BR	18\$
3125							
3126	011342	012737	000001	003254	17\$:	MOV	#1,PR.BIT
3127	011350	004737	037026		18\$:	JSR	PC,RDBIT ;SIMULATE NEXT BIT
3128	011354	005300				DEC	R0 ;CHECK IF READY FOR NEXT HEADER WORD
3129	011356	001361				BNE	15\$;NO, CONTINUE
3130	011360	005301				DEC	R1 ;CHECK IF FINISHED WITH HEADER
3131	011362	001354				BNE	12\$;NO, CONTINUE
3132	011364	012700	000100			MOV	#64,R0 ;LOAD COUNT FOR GAP
3133	011370	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT ;SIMULATE GAP
3134	011376	005037	003254			CLR	PR.BIT
3135	011402	004737	037026			JSR	PC,RDBIT
3136	011406	005300				DEC	R0 ;CHECK IF GAP FINISHED
3137	011410	001367				BNE	25\$;NO, CONTINUE
3138	011412	016237	000006	003146		MOV	RKDA(R2),T.DA ;GET DISK ADDRESS REG
3139	011420	016237	000020	003160		MOV	RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3140	011426	023737	003206	003146		CMP	E.DA,T.DA ;CHECK DISK ADD CORRECT
3141	011434	001401				BEQ	30\$;YES, CONTINUE
3142	011436	104054				ERROR	54 ;DISK ADDRESS INCORRECT

```

3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164 011470 000004
3165 011472 012737 000012 001200
3166 011500 013702 001270
3167 011504 005037 003300
3168 011510 012737 000025 003276
3169 011516 012737 011524 001110
3170
3171
3172 011524
3173 011524 004737 035624
3174 011530 012762 100000 000000
3175 011536 012762 000040 000026
3176 011544 013762 003300 000020
3177 011552 013762 003276 000006
3178 011560 012762 177777 000002
3179 011566 012762 055216 000004
3180 011574 012762 000023 000000
3181 011602 012700 000426
3182
3183 011606 012762 000440 000026
3184 011614 012762 000040 000026
3185 011622 005300
3186 011624 001370
3187 011626 012762 000140 000026
3188 011634 012762 000040 000026
3189 011642 005037 003254
3190 011646 005037 003256
3191 011652 012700 000377
3192 011656 004737 037026
3193 011662 005300
3194 011664 001374
3195 011666 012737 000001 003254
3196 011674 004737 037026
3197 011700 012703 003302
3198 011704 012701 000003
3199 011710 012304
3200 011712 012700 000020
3201 011716 013737 003254 003256
3202 011724 006004
3203 011726 103403
3204 011730 005037 003254
3205 011734 000403
3206

*****
TEST 17 TRACK INCREMENT
*****
CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD
TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0,
SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
THAT WRITE GATE SETS.

REPEAT FOR TRACK = 1.
*****
TST17: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
CLR CYLN ;INITIALIZE CYLINDER
MOV #25, SECTOR ;INITIALIZE TRACK AND SECTOR
MOV #1$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
; SUBTEST LOOP

1$: JSR PC, INCHDR ;GENERATE HEADER WORDS
MOV #CCLR, RKCS1(R2) ;CLEAR RK611 CONTROLLER
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV CYLN, RKDCYL(R2) ;LOAD DESIRED CYLINDER
MOV SECTOR, RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
MOV #-1, RKWC(R2) ;WORD COUNT=1
MOV #BUFF, RKBA(R2) ;LOAD DUMMY ADDRESS
MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
MOV #69, *4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE

5$: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 5$
MOV #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ;GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
10$: JSR PC, RDBIT
DEC R0
BNE 10$
MOV #1, PR.BIT ;SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #HEADER, R3 ;SIMULATE HEADER
MOV #3, R1
MOV (R3)+, R4 ;GET NEXT HEADER WORD
MOV #16, R0 ;LOAD BITS PER WORD
15$: MOV PR.BIT, M1.BIT
ROR R4
BCS 17$
CLR PR.BIT
BR 18$

```



```

3207 011736 012737 000001 003254 17$: MOV #1,PR.BIT
3208 011744 004737 037026 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3209 011750 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
3210 011752 001361 BNE 15$ ;NO, CONTINUE
3211 011754 005301 R1 ;CHECK IF FINISHED WITH HEADER
3212 011756 001354 BNE 12$ ;NO, CONTINUE
3213 011760 012700 000100 MOV #64,RO ;LOAD COUNT FOR GAP
3214 011764 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3215 011772 005037 003254 CLR PR.BIT
3216 011776 004737 037026 JSR PC,RDBIT
3217 012002 005300 DEC RO ;CHECK IF GAP FINISHED
3218 012004 001367 BNE 25$ ;NO, CONTINUE
3219 012006 016237 000006 003146 MOV RKDA(R2),T.DA ;GET DISK ADDRESS REG
3220 012014 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3221 012022 023737 003206 003146 CMP E.DA,T.DA ;CHECK DISK ADD CORRECT
3222 012030 001401 BEQ 30$ ;YES, CONTINUE
3223 012032 104056 ERROR 56 ;DISK ADDRESS INCORRECT
3224 012034 023737 003220 003160 30$: CMP E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3225 012042 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3226 012044 104002 ERROR R2 ;CYLINDER INCORRECT
3227 012046 104415 SCOP1 ;CHECK IF LOOP ON ERROR
3228 012050 105237 003277 INCB TRACK ;INCREMENT TRACK
3229 012054 122737 000003 003277 CMPB #3,TRACK ;CHECK IF ALL TRACKS TRIED
3230 012062 001220 BNE 1$ ;NO, TRY NEXT VALUE
3231
3232 ;*****
3233 ;TEST 20 CYLINDER INCREMENT
3234 ;
3235 ; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3236 ; CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ONE WORD
3237 ; TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2.
3238 ; SECTOR 25. CLOCK IN BOTH SEEK AND DRIVE CLEAR MESSAGES.
3239 ; SIMULATE SECTOR PULSE AND PROPER HEADER, MAKE SURE
3240 ; THAT WRITE GATE SETS.
3241 ;
3242 ; REPEAT FOR CYLINDER = 1-632.
3243 ;
3244 ;*****
3245 ;ST20: SCOPE
3246 012064 000004 MOV #10,$TIMES ;DO 10. ITERATIONS
3247 012066 012737 000012 001200 MOV $BASE,R2 ;LOAD RK611 BASE
3248 012074 013702 001270 MOV CLN,R2 ;INITIALIZE CYLINDER
3249 012100 005037 003300 MOV #1025,SECTOR ;INITIALIZE TRACK AND SECTOR
3250 012104 012737 001025 003276 MOV #1$,SLPERR ;LOAD LOOP ON ERROR LOCATION FOR
3251 012112 012737 012120 001110 ; SUBTEST LOOP
3252
3253 1$: JSR PC,INCHDR ;GENERATE HEADER WORDS
3254 012120 004737 035624 MOV #CCLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3255 012124 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3256 012132 012762 000040 000026 MOV CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3257 012140 013762 003300 000020 MOV SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3258 012146 013762 003276 000006 MOV #-1,RKWC(R2) ;WORD COUNT=1
3259 012154 012762 177777 000002 MOV #BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3260 012162 012762 055216 000004 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3261 012170 012762 000023 000000 MOV #69.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL READY
3262 012176 012700 000426

```

```

3263                                     ; FOR SECTOR PULSE
3264 012202 012762 000440 000026 5$: MOV #DMD:MCLK,RKMR1(R2)
3265 012210 012762 000040 000026 MOV #DMD,RKMR1(R2)
3266 012216 005300 DEC R0
3267 012220 001370 BNE 5$
3268 012222 012762 000140 000026 MOV #DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3269 012230 012762 000040 000026 MOV #DMD,RKMR1(R2)
3270 012236 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3271 012242 005037 003256 CLR M1.BIT
3272 012246 012700 000377 MOV #255,R0
3273 012252 004737 037026 10$: JSR PC,RDBIT
3274 012256 005300 DEC R0
3275 012260 001374 BNE 10$
3276 012262 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3277 012270 004737 037026 JSR PC,RDBIT
3278 012274 012703 003302 MOV #HEADER,R3 ;SIMULATE HEADER
3279 012300 012701 000003 MOV #3,R1
3280 012304 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3281 012306 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
3282 012312 013737 003254 003256 15$: MOV PR.BIT,M1.BIT
3283 012320 006004 ROR R4
3284 012322 103403 BCS 17$
3285 012324 005037 003254 CLR PR.BIT
3286 012330 000403 BR 18$
3287
3288 012332 012737 000001 003254 17$: MOV #1,PR.BIT
3289 012340 004737 037026 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3290 012344 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3291 012346 001361 BNE 15$ ;NO, CONTINUE
3292 012350 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3293 012352 001354 BNE 12$ ;NO, CONTINUE
3294 012354 012700 000100 MOV #64,R0 ;LOAD COUNT FOR GAP
3295 012360 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3296 012366 005037 003254 CLR PR.BIT
3297 012372 004737 037026 JSR PC,RDBIT
3298 012376 005300 DEC R0 ;CHECK IF GAP FINISHED
3299 012400 001367 BNE 25$ ;NO, CONTINUE
3300 012402 016237 000006 003146 MOV RKDA(R2),T.DA ;GET DISK ADDRESS REG
3301 012410 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3302 012416 023737 003206 003146 CMP E.DA,T.DA ;CHECK DISK ADD CORRECT
3303 012424 001401 BEQ 30$ ;YES, CONTINUE
3304 012426 104060 ERROR 60 ;DISK ADDRESS INCORRECT
3305 012430 023737 003220 003160 30$: CMP E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3306 012436 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3307 012440 104002 ERROR R2 ;CYLINDER INCORRECT
3308 012442 104415 32$: SCOP1 ;CHECK IF LOOP ON ERROR
3309 012444 005237 003300 INC CYLN ;INCREMENT CYLINDER
3310 012450 022737 000633 003300 CMP #633,CYLN ;CHECK IF ALL CYLINDER TRIED
3311 012456 001220 BNE 1$ ;NO, TRY NEXT VALUE
3312
3313 *****
3314 *TEST 21 BAD SECTOR ERROR (PART 1)
3315 *
3316 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3317 * CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3318 * ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,

```


3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374

012460	000004		
012462	012737	000012	001200
012470	013702	001270	
012474	012762	100000	000000
012502	012762	000040	000026
012510	012762	000000	000020
012516	012762	000000	000006
012524	012762	177777	000002
012532	012762	055216	000004
012540	012762	000023	000000
012546	012700	000426	
012552	012762	000440	000026
012560	012762	000040	000026
012566	005300		
012570	001370		
012572	012762	000140	000026
012600	012762	000040	000026
012606	005037	003254	
012612	005037	003256	
012616	012700	000377	
012622	004737	037026	
012626	005300		
012630	001374		
012632	012737	000001	003254
012640	004737	037026	
012644	012703	053372	
012650	012701	000003	
012654	012304		
012656	012700	000020	
012662	013737	003254	003256
012670	006004		
012672	103403		
012674	005037	003254	
012700	000403		
012702	012737	000001	003254
012710	004737	037026	
012714	005300		
012716	001361		
012720	005301		
012722	001354		
012724	012700	000100	
012730	013737	003254	003256

```

: * HEAD 0, SECTOR 0. CLOCK IN BOTH SEEK AND DRIVE CLEAR
: * MESSAGES. SIMULATE A SECTOR PULSE AND A HEADER WITH
: * THE FOLLOWING DATA:
: *
: * 000000
: * 040000
: * 040000
: *
: * MAKE SURE BAD SECTOR ERROR SETS. CHECK THAN DISK ADDRESS
: * IS NOT INCREMENTED.
: *
: *****
: ST21: SCOPE
: MOV #10, $TIMES ; DO 10. ITERATIONS
: MOV $BASE, R2 ; LOAD RK611 BASE
: MOV #CLR, RKCS1(R2) ; CLEAR RK611
: MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
: MOV #0, RKDCYL(R2) ; LOAD CYLINDER
: MOV #0, RKDA(R2) ; LOAD TRACK AND SECTOR
: MOV #-1, RKWC(R2) ; WORD COUNT=1
: MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
: MOV #WRDATA, RKCS1(R2) ; ISSUE WRITE DATA
: MOV #69.*4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
: ; FOR SECTOR PULSE
: 1$: MOV #DMD!MCLK, RKMR1(R2)
: MOV #DMD, RKMR1(R2)
: DEC R0
: BNE 1$
: MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
: MOV #DMD, RKMR1(R2)
: CLR PR.BIT ; GENERATE SYNCH
: CLR M1.BIT
: MOV #255, R0
: JSR PC, RDBIT
: DEC R0 ; CHECK IF SYNCH FINISHED
: BNE 5$
: MOV #1, PR.BIT ; SIMULATE SYNCH BIT
: JSR PC, RDBIT
: MOV #HEAD2, R3 ; SIMULATE HEADER ERROR
: MOV #3, R1
: 12$: MOV (R3)+, R4 ; GET NEXT HEADER WORD
: MOV #16, R0 ; LOAD BITS PER WORD
: 15$: MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
: ROR R4 ; GET NEXT BIT
: BCS 17$
: CLR PR.BIT
: BR 18$
: 17$: MOV #1, PR.BIT
: 18$: JSR PC, RDBIT ; SIMULATE NEXT BIT
: DEC R0 ; CHECK IF READY FOR NEXT HEADER WORD
: BNE 15$ ; NO, CONTINUE
: DEC R1 ; CHECK IF FINISHED WITH HEAD2
: BNE 12$ ; NO, CONTINUE
: MOV #64, R0 ; LOAD COUNT FOR GAP
: 25$: MOV PR.BIT, M1.BIT ; SIMULATE GAP

```

NOS

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 65
T21 BAD SECTOR ERROR (PART 1)

SEQ 0065

3375	012736	005037	003254		CLR	PR.BIT	
3376	012742	004737	037026		JSR	PC,RDBIT	
3377	012746	005300			DEC	RO	:CHECK IF GAP FINISHED
3378	012750	001367			BNE	25\$	:NO CONTINUE
3379	012752	016237	000000	003140	MOV	RKCS1(R2),T.CS1	:GET CS1
3380	012760	016237	000010	003150	MOV	RKCS2(R2),T.CS2	:GET CS2
3381	012766	016237	000014	003154	MOV	RKER(R2),T.ER	:GET ERROR REG
3382	012774	012737	000023	003200	MOV	#WRDATA,E.CS1	:LOAD EXPECTED CS1
3383	013002	012737	000300	003210	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3384	013010	012737	000200	003214	MOV	#BSE,E.ER	:LOAD EXPECTED ERROR REG.
3385	013016	023737	003200	003140	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
3386	013024	001401			BEQ	30\$	:YES, CHECK CS2
3387	013026	104063			ERROR	63	:CS1 INCORRECT
3388	013030	023737	003210	003150	30\$: CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
3389	013036	001401			BEQ	32\$	:YES, CHECK ERROR REG.
3390	013040	104064			ERROR	64	:CS2 INCORRECT
3391	013042	023737	003214	003154	32\$: CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3392	013050	001401			BEQ	TST2	:YES GO ON TO NEXT TEST
3393	013052	104065			ERROR	65	:ERROR REG INCORRECT
3394							
3395							
3396							
3397							
3398							
3399							
3400							
3401							
3402							
3403							
3404							
3405							
3406							
3407							
3408							
3409							
3410							
3411							
3412							
3413	013054	000004			TST2: SCOPE		
3414	013056	012737	000012	001200	MOV	#10,\$TIMES	:DO 10. ITERATIONS
3415	013064	013702	001270		MOV	\$BASE,R2	:LOAD RK611 BASE
3416	013070	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3417	013076	012762	000040	000026	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3418	013104	012762	000000	000020	MOV	#0,RKDCYL(R2)	:LOAD CYLINDER
3419	013112	012762	000000	000006	MOV	#0,RKDA(R2)	:LOAD TRACK AND SECTOR
3420	013120	012762	177777	000002	MOV	#-1,RKWC(R2)	:WORD COUNT=1
3421	013126	012762	055216	000004	MOV	#8UFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
3422	013134	012762	000023	000000	MOV	#WRDATA,RKCS1(R2)	:ISSUE WRITE DATA
3423	013142	012700	000426		MOV	#69,*4+2,RO	:ISSUE ENOUGH CLOCKS UNTIL READY
3424							:FOR SECTOR PULSE
3425	013146	012762	000440	000026	1\$: MOV	#DMD!MCLK,RKMR1(R2)	
3426	013154	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3427	013162	005300			DEC	RO	
3428	013164	001370			BNE	1\$	
3429	013166	012762	000140	000026	MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
3430	013174	012762	000040	000026	MOV	#DMD,RKMR1(R2)	

```

3431 013202 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3432 013206 005037 003256 CLR M1.BIT
3433 013212 012700 000377 MOV #255, R0
3434 013216 004737 037026 5$: JSR PC, RDBIT
3435 013222 005300 DEC R0 ;CHECK IF SYNCH FINISHED
3436 013224 001374 BNE 5$
3437 013226 012737 000001 003254 MOV #1, PR.BIT ;SIMULATE SYNCH BIT
3438 013234 004737 037026 JSR PC, RDBIT
3439 013240 012703 053400 MOV #HEAD3, R3 ;SIMULATE HEADER ERROR
3440 013244 012701 000003 MOV #3, R1
3441 013250 012304 12$: MOV (R3)+, R4 ;GET NEXT HEADER WORD
3442 013252 012700 000020 MOV #16, R0 ;LOAD BITS PER WORD
3443 013256 013737 003254 003256 15$: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
3444 013264 006004 ROR R4 ;GET NEXT BIT
3445 013266 103403 BCS 17$
3446 013270 005037 003254 CLR PR.BIT
3447 013274 000403 BR 18$
3448
3449 013276 012737 000001 003254 17$: MOV #1, PR.BIT
3450 013304 004737 037026 18$: JSR PC, RDBIT ;SIMULATE NEXT BIT
3451 013310 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3452 013312 001361 BNE 15$ ;NO, CONTINUE
3453 013314 005301 DEC R1 ;CHECK IF FINISHED WITH HEAD3
3454 013316 001354 BNE 12$ ;NO, CONTINUE
3455 013320 012700 000100 MOV #64, R0 ;LOAD COUNT FOR GAP
3456 013324 013737 003254 003256 25$: MOV PR.BIT, M1.BIT ;SIMULATE GAP
3457 013332 005037 003254 CLR PR.BIT
3458 013336 004737 037026 JSR PC, RDBIT
3459 013342 005300 DEC R0 ;CHECK IF GAP FINISHED
3460 013344 001367 BNE 25$ ;NO, CONTINUE
3461 013346 016237 000000 003140 MOV RKCS1(R2), T.CS1 ;GET CS1
3462 013354 016237 000010 003150 MOV RKCS2(R2), T.CS2 ;GET CS2
3463 013362 016237 000014 003154 MOV RKER(R2), T.ER ;GET ERROR REG
3464 013370 012737 000023 003200 MOV #WADATA, E.CS1 ;LOAD EXPECTED CS1
3465 013376 012737 000300 003210 MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
3466 013404 012737 000200 003214 MOV #BSE E.ER ;LOAD EXPECTED ERROR REG.
3467 013412 023737 003200 003140 CMP E.CS1, T.CS1 ;CHECK CS1 CORRECT
3468 013420 001401 BEQ 30$ ;YES, CHECK CS2
3469 013422 104063 ERROR 63 ;CS1 INCORRECT
3470 013424 023737 003210 003150 30$: CMP E.CS2, T.CS2 ;CHECK CS2 CORRECT
3471 013432 001401 BEQ 32$ ;YES, CHECK ERROR REG.
3472 013434 104064 ERROR 64 ;CS2 INCORRECT
3473 013436 023737 003214 003154 32$: CMP E.ER, T.ER ;CHECK ERROR REG CORRECT
3474 013444 001401 BEQ TST23 ;YES, GO ON TO NEXT TEST
3475 013446 104065 ERROR 65 ;ERROR REG INCORRECT

```

```

*****
*TEST 23 OPERATION INCOMPLETE
*
* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253,
* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH
* 1 BIT DIFFERENT IN 30 BITS OF OPI DETERMINATION. ALL
* SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY

```

```

3487 ;* OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS
3488 ;* THAT SET.
3489 ;*
3490 ;*****
3491 ST23: SCOPE
3492 MOV #10, $TIMES ;DO 10. ITERATIONS
3493 MOV $BASE, R2 ;LOAD RK611 BASE
3494 MOV #CCLR, RKCS1(R2) ;CLEAR RK611
3495 MOV #2500, R0 ;SET COUNT FOR STALL
3496 2$: DEC R0 ;DEC COUNT
3497 BNE 2$ ;LOOP UNTIL 0
3498 MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3499 MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
3500 MOV #-1, RKWC(R2) ;WORD COUNT = 1
3501 MOV #1253, RKDCYL(R2) ;LOAD CYLINDER
3502 MOV #1023, RKDA(R2) ;LOAD TRACK AND SECTOR
3503 MOV #WRDATA!CFMT, RKCS1(R2) ;ISSUE WRITE DATA
3504 MOV #69, #4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3505 ; FOR SECTOR PULSE
3506 1$: MOV #DMD!MCLK, RKMR1(R2)
3507 MOV #DMD, RKMR1(R2)
3508 DEC R0
3509 BNE 1$
3510 MOV #32, R5 ;LOAD HEADER COUNT
3511 MOV #OPI1, R3 ;LOAD ADDRESS OF HEADERS
3512 MOV #WRDATA!CFMT, E.CS1 ;LOAD EXPECTED CS1
3513 MOV #IR!OR, E.CS2 ;LOAD EXPECTED CS2
3514 MOV #E, ER ;LOAD EXPECTED ERROR REG
3515 MOV #DMD!MEWD!ECCW, E.MR1 ;LOAD EXPECTED MRI
3516 CLR HDRCNT ;INITIALIZE HEADER COUNT
3517 5$: MOV #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
3518 MOV #DMD, RKMR1(R2)
3519 CMP #31, HDRCNT ;CHECK IF ALL HEADERS DONE
3520 BEQ 26$ ;YES - SKIP TO ERROR TEST
3521 CLR PR.BIT ;GENERATE SYNCH
3522 CLR M1.BIT
3523 MOV #255, R0
3524 10$: JSR PC, RDBIT
3525 DEC R0 ;CHECK IF SYNCH FINISHED
3526 BNE 10$
3527 MOV #1, PR.BIT ;SIMULATE SYNCH BIT
3528 JSR PC, RDBIT
3529 MOV #3, R1 ;SIMULATE OPI
3530 MOV (R3)+, R4 ;GET NEXT HEADER WORD
3531 MOV #16, R0 ;LOAD BITS PER WORD
3532 12$: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
3533 ROR R4 ;GET NEXT BIT
3534 BCS 17$
3535 CLR PR.BIT
3536 BR 18$
3537
3538 17$: MOV #1, PR.BIT
3539 18$: JSR PC, RDBIT ;SIMULATE NEXT BIT
3540 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3541 BNE 15$ ;NO, CONTINUE
3542 DEC R1 ;CHECK IF FINISHED WITH HEADER

```

```
3543 013770 001354 BNE 125 ;NO CONTINUE
3544 013772 012700 000100 MOV #64, R0 ;LOAD COUNT FOR GAP
3545 013776 013737 003254 003256 25$: MOV PR.BIT, M1.BIT ;SIMULATE GAP
3546 014004 005037 003254 CLR PR.BIT
3547 014010 004737 037026 JSR PC, R0BIT
3548 014014 005300 DEC R0 ;CHECK IF GAP IS FINISHED
3549 014016 001367 BNE 25$ ;NO CONTINUE
3550 014020 016237 000000 003140 26$: MOV RKCS1(R2), T.CS1 ;GET CS1
3551 014026 016237 000010 003150 MOV RKCS2(R2), T.CS2 ;GET CS2
3552 014034 016237 000014 003154 MOV RKER(R2), T.ER ;GET ERROR REG.
3553 014042 023737 003200 003140 CMP E.CS1, T.CS1 ;CHECK CS1 CORRECT
3554 014050 001401 BEQ 30$ ;YES, CONTINUE
3555 014052 104074 ERROR 74 ;CS1 INCORRECT
3556 014054 023737 003210 003150 30$: CMP E.CS2, T.CS2 ;CHECK CS2 CORRECT
3557 014062 001401 BEQ 31$ ;YES, CONTINUE
3558 014064 104075 ERROR 75 ;CS2 INCORRECT
3559 014066 023737 003214 003154 31$: CMP E.ER, T.ER ;CHECK ERROR REG CORRECT
3560 014074 001401 BEQ 32$ ;YES, CONTINUE
3561 014076 104076 ERROR 76 ;ERROR REG INCORRECT
3562 014100 016237 000026 003164 32$: MOV RKMR1(R2), T.MR1 ;GET MR1
3563 014106 023737 003224 003164 CMP E.MR1, T.MR1 ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
3564 014114 001401 BEQ 34$ ;YES, CONTINUE
3565 014116 104077 ERROR 77 ;MR1 INCORRECT
3566 014120 005237 003310 34$: INC HDRCNT ;INCREMENT HEADER COUNT
3567 014124 022737 000037 003310 CMP #31, HDRCNT ;CHECK IF LAST HEADER
3568 014132 001011 BNE 35$ ;NO CHECK IF FINISHED
3569 014134 012737 020000 003214 MOV #0PI, E.ER ;LOAD ERROR BIT
3570 014142 042737 000001 003200 BIC #GO, E.CS1 ;ADJUST E.CS1 FOR END OF OP CONTENTS
3571 014150 052737 100200 003200 BIS #RDY!CERR, E.CS1
3572 014156 005305 35$: DEC R5 ;CHECK IF FINISHED
3573 014160 001402 BEQ 37$ ;YES - SKIP
3574 014162 000137 JMP 5$ ;DO NEXT SECTOR
3575 014166 012762 100000 000000 37$: MOV #CCLR, RKCS1(R2) ;CLEAR CONTROLLER
3576 014174 016237 000000 003140 MOV RKCS1(R2), T.CS1 ;GET CS1
3577 014202 016237 000010 003150 MOV RKCS2(R2), T.CS2 ;CS2
3578 014210 016237 000014 003154 MOV RKER(R2), T.ER ;ER
3579 014216 012737 000200 003200 MOV #RDY, E.CS1 ;SET EXPECTED CS1
3580 014224 023737 003140 003200 CMP T.CS1, E.CS1 ;CHECK IF CORRECT
3581 014232 001401 BEQ TST24 ;GO TO NEXT TEST
3582 014234 104153 ERROR 153
```

```
*****
*TEST 24 HEADER VRC
*
* CLEAR RK611 CONTROLLER WITH A CONTR LER CLEAR. PUT
* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
* ONE WORD WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1253,
* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
* MESSAGES SIMULATE A SECTOR PULSE AND HEADER WITH BIT 0
* OF THE VRC INCORRECT. MAKE SURE ONLY HEADER VRC AND
* CONTROLLER ERROR ARE THE ONLY ERRORS SET. REPEAT FOR
* BITS 1-15 OF VRC.
*****
TST24: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
```

3599	014246	013702	001270		MOV	\$BASE,R2	;LOAD RK611 BASE
3600	014252	012737	012023	003200	MOV	#CFMT!CDT!WRDATA,E.CS1	;LOAD EXPECTED CS1
3601	014260	012737	000300	003210	MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
3602	014266	012737	000400	003214	MOV	#HVR,E.ER	;LOAD EXPECTED ERROR REGISTER
3603	014274	012737	000001	001170	MOV	#1,STMP4	;INITIALIZE BIT FOR VRC ERROR
3604	014302	012737	014310	001110	MOV	#1\$,SLPERR	;LOAD LOOP ON ERROR LOCATION FOR
3605							; SUBTEST LOOP
3606							
3607	014310				1\$: MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
3608	014310	012762	100000	000000	MOV	#2000,R0	;SET COUNT FOR STALL
3609	014316	012700	002000		2\$: DEC	R0	;DEC COUNT
3610	014322	005300			BNE	2\$	;LOOP UNTIL 0
3611	014324	001376			MOV	#DMD,RKMR1(R2)	;PUT RK611 IN MAINTENANCE MODE
3612	014326	012762	000040	000026	MOV	#140370,VRC HDR+4	;LOAD VRC
3613	014334	012737	140370	055200	BIT	STMP4,VRC HDR+4	;MAKE ONE BIT BAD
3614	014342	033737	001170	055200	BEQ	3\$	
3615	014350	001404			BIC	STMP4,VRC HDR+4	
3616	014352	043737	001170	055200	BR	5\$	
3617	014360	000403					
3618							
3619	014362	053737	001170	055200	3\$: BIS	STMP4,VRC HDR+4	
3620	014370	012762	055216	000004	5\$: MOV	#BUFF,RKBA(R2)	;LOAD DUMMY ADDRESS
3621	014376	012762	177777	000002	MOV	#-1,RKWC(R2)	;WORD COUNT = 1
3622	014404	012762	001023	000006	MOV	#1023,RKDA(R2)	;LOAD TRACK 2 SECTOR 23
3623	014412	012762	001253	000020	MOV	#1253,RKDCYL(R2)	;LOAD CYLINDER 1253
3624	014420	012762	012023	000000	MOV	#CFMT!CDT!WRDATA,RKCS1(R2)	;ISSUE COMMAND
3625	014426	012700	000426		MOV	#69,*4+2,R0	;LOAD COUNT UNTIL READY FOR SECTOR
3626	014432	012762	000440	000026	10\$: MOV	#DMD!MCLK,RKMR1(R2)	
3627	014440	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3628	014446	005300			DEC	R0	
3629	014450	001370			BNE	10\$	
3630	014452	012762	000140	000026	MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
3631	014460	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3632	014466	005037	003254		CLR	PR.BIT	;INITIAL BITS
3633	014472	005037	003256		CLR	M1.BIT	
3634	014476	012700	000377		MOV	#255,R0	;SIMULATE SYNCH
3635	014502	004737	037026		15\$: JSR	PC,R0BIT	
3636	014506	005300			DEC	R0	
3637	014510	001374			BNE	15\$	
3638	014512	012737	000001	003254	MOV	#1,PR.BIT	;GENERATE SYNCH BIT
3639	014520	004737	037026		JSR	PC,R0BIT	;SIMULSTE SYNC 1 BIT
3640	014524	012703	055174		MOV	#VRC HDR,R3	;LOAD ADDRESS OF HEADER
3641	014530	012701	000003		MOV	#3,R1	;LOAD HEADER COUNT
3642	014534	012304			17\$: MOV	(R3)+,R4	;GET NEXT HEADER WORD
3643	014536	012700	000020		MOV	#16,R0	;LOAD BITS PER WORD
3644	014542	013737	003254	003256	20\$: MOV	PR.BIT,M1.BIT	;SIMULATE NEXT BIT
3645	014550	006004			ROR	R4	
3646	014552	103403			BCS	23\$	
3647	014554	005037	003254		CLR	PR.BIT	
3648	014560	000403			BR	25\$	
3649							
3650	014562	012737	000001	003254	23\$: MOV	#1,PR.BIT	
3651	014570	004737	037026		25\$: JSR	PC,R0BIT	
3652	014574	005300			DEC	R0	;CHECK IF FINISHED WITH WORD
3653	014576	001361			BNE	20\$	;NO, CONTINUE
3654	014600	005301			DEC	R1	;CHECK IF HEADER FINISHED


```

3655 014602 001354      BNE      17$          ;NO, CONTINUE
3656 014604 012700 000100  MOV      864, R0          ;SIMULATE GAP
3657 014610 013737 003254 003256 30$: MOV      PR.BIT, M1.BIT
3658 014616 005037 003254      CLR      PR.BIT
3659 014622 004737 037026      JSR      PC, R0BIT
3660 014626 005300      DEC      R0
3661 014630 001367      BNE      30$
3662 014632 016237 000000 003140  MOV      RKCS1(R2), T.CS1 ;STORE CS1
3663 014640 016237 000010 003150  MOV      RKCS2(R2), T.CS2 ;STORE CS2
3664 014646 016237 000014 003154  MOV      RKER(R2), T.ER  ;STORE ERROR REG
3665 014654 023737 003200 003140  CMP      E.CS1, T.CS1    ;CHECK CS1 CORRECT
3666 014662 001401      BEQ      35$          ;YES, CHECK CS2
3667 014664 104071      ERROR     71          ;CS1 INCORRECT
3668 014666 023737 003210 003150 35$: CMP      E.CS2, T.CS2    ;CHECK CS2 CORRECT
3669 014674 001401      BEQ      37$          ;YES, CHECK ERROR REG
3670 014676 104072      ERROR     72          ;CS2 INCORRECT
3671 014700 023737 003214 003154 37$: CMP      E.ER, T.ER      ;CHECK ERROR REG CORRECT
3672 014706 001401      BEQ      40$          ;YES, CHECK IF LOOP ON ERROR
3673 014710 104073      ERROR     73          ;ERROR REG INCORRECT
3674 014712 104415      SCOP1          ;CHECK IF LOOP ON ERROR
3675 014714 006337 001170      ASL      $TMP4      ;GET NEXT PATTERN
3676 014720 103402      BCS      TST25        ;CHECK IF FINISHED
3677 014722 000137 014310      JMP      1$          ;NO, CONTINUE
3678
3679
3680 *****
3681 *TEST 25      BAD SECTOR ERROR AND HEADER VRC
3682 *
3683 *      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3684 *      CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3685 *      ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300.
3686 *      HEAD 1, SECTOR 17, CLOCK IN BOTH SEEK AND DRIVE CLEAR
3687 *      MESSAGES. SIMULATE THE FOLLOWING HEADER:
3688 *
3689 *      000300
3690 *      040057
3691 *      040356
3692 *
3693 *      MAKE SURE ONLY HEADER VRC ERROR SETS.
3694 *****
3695 *ST25: SCOPE
3696 014726 000004      MOV      #10, $TIMES ;DO 10. ITERATIONS
3697 014730 012737 000012 001200  MOV      $BASE, R2 ;LOAD RK611 BASE
3698 014736 013702 001270      MOV      #CCLR, RKCS1(R2) ;CLEAR RK611
3699 014742 012762 100000 000000  MOV      #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3700 014756 012762 000040 000026  MOV      #300, RKDCYL(R2) ;LOAD CYLINDER
3701 014764 012762 000300 000020  MOV      #417, RKDA(R2) ;LOAD TRACK AND SECTOR
3702 014772 012762 000417 000006  MOV      #-1, RKWC(R2) ;WORD COUNT=1
3703 015000 012762 177777 000002  MOV      #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3704 015006 012762 055216 000004  MOV      #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
3705 015014 012700 000426      MOV      #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3706                                     ;FOR SECTOR PULSE
3707 015020 012762 000440 000026 1$: MOV      #DMD!MCLK, RKMR1(R2)
3708 015026 012762 000040 000026  MOV      #DMD, RKMR1(R2)
3709 015034 005300      DEC      R0
3710 015036 001370      BNE      1$

```

3711	015040	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
3712	015046	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3713	015054	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
3714	015060	005037	003256			CLR	M1.BIT	
3715	015064	012700	000377			MOV	#255,R0	
3716	015070	004737	037026		5\$:	JSR	PC,RDBIT	
3717	015074	005300				DEC	R0	;CHECK IF SYNCH FINISHED
3718	015076	001374				BNE	5\$	
3719	015100	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
3720	015106	004737	037026			JSR	PC,RDBIT	
3721	015112	012703	053406			MOV	#HEAD4,R3	;SIMULATE HEADER ERROR
3722	015116	012701	000003			MOV	#3,R1	
3723	015122	012304			12\$:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
3724	015124	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
3725	015130	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
3726	015136	006004				ROR	R4	;GET NEXT BIT
3727	015140	103403				BCS	17\$	
3728	015142	005037	003254			CLR	PR.BIT	
3729	015146	000403				BR	18\$	
3730								
3731	015150	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
3732	015156	004737	037026		18\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
3733	015162	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
3734	015164	001361				BNE	15\$	;NO, CONTINUE
3735	015166	005301				DEC	R1	;CHECK IF FINISHED WITH HEAD4
3736	015170	001354				BNE	12\$	;NO, CONTINUE
3737	015172	012700	000100			MOV	#64,R0	;LOAD COUNT FOR GAP
3738	015176	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
3739	015204	005037	003254			CLR	PR.BIT	
3740	015210	004737	037026			JSR	PC,RDBIT	
3741	015214	005300				DEC	R0	;CHECK IF GAP FINISHED
3742	015216	001367				BNE	25\$	;NO, CONTINUE
3743	015220	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
3744	015226	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
3745	015234	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ERROR REG
3746	015242	012737	000023	003200		MOV	#WRDATA,E.CS1	;LOAD EXPECTED CS1
3747	015250	012737	000300	003210		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
3748	015256	012737	000400	003214		MOV	#HVRG,E.ER	;LOAD EXPECTED ERROR REG.
3749	015264	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
3750	015272	001401				BEQ	30\$	;YES, CHECK CS2
3751	015274	104066				ERROR	66	;CS1 INCORRECT
3752	015276	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
3753	015304	001401				BEQ	32\$	;YES, CHECK ERROR REG.
3754	015306	104067				ERROR	67	;CS2 INCORRECT
3755	015310	023737	003214	003154	32\$:	CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
3756	015316	001401				BEQ	TST26	;YES, GO ON TO NEXT TEST
3757	015320	104070				ERROR	70	;ERROR REG INCORRECT

```

3758
3759
3760
3761
3762
3763
3764
3765
3766

```

```

*****
;TEST 26      GOOD HEADER AND PREVIOUS BSE
;
;      CLEAR RK611 CONTROLLER WITH CONTROLLER CLEAR.  PUT
;      CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA OF
;      ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 100,
;      HEAD 0, SECTOR 1.  CLOCK THROUGH SEEK AND DRIVE CLEAR
;      MESSAGES.  SIMULATE A SECTOR PULSE AND A HEADER OF THE

```


H06

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 72
T26 GOOD HEADER AND PREVIOUS BSE

SEQ 0072

FOLLOWING 3 WORDS WITH A BAD SECTOR INDICATION:

000100
040000
040100MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
SET. SIMULATE A SECTOR PULSE AND A HEADER OF THE FOLLOWING
3 WORDS:000100
140001
140101MAKE SURE WRITE GATE SETS INDICATING THAT HEADER HAS
BEEN RECOGNIZED.

+ST26: SCOPE

```

MOV #10, $TIMES ; DO 10. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF, RKBA(R2) ; LOAD DUMMY BUS ADDRESS
MOV #-1, RKWC(R2) ; WORD COUNT -1
MOV #1, RKDA(R2) ; LOAD TRACK AND SECTOR
MOV #100, RKDCYL(R2) ; LOAD CYLINDER
MOV #WRDATA, RKCS1(R2) ; ISSUE WRITE DATA
MOV #69, *4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE

MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
1$:
MOV #2, R5 ; LOAD HEADER COUNT
MOV #HEADS, R3 ; LOAD ADDRESS OF HEADERS
MOV #WRDATA, E.CS1 ; LOAD EXPECTED CS1
MOV #IR!OR, E.CS2 ; LOAD EXPECTED CS2
CLR E.ER ; LOAD EXPECTED MR1
MOV #DMD!MEWD!ECCW, E.MR1 ; LOAD EXPECTED MR1
CLR HDRCNT ; INITIALIZE HEADER COUNT
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; GENERATE SYNCH
CLR M1.BIT
MOV #255, R0
JSR PC, RDBIT
DEC R0 ; CHECK IF SYNCH FINISHED
BNE 10$
MOV #1, PR.BIT ; SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #3, R1 ; SIMULATE HEADER
MOV (R3)+, R4 ; GET NEXT HEADER WORD
MOV #16, R0 ; LOAD BITS PER WORD
MOV PR.BIT, M1.BIT ; STORE PREVIOUS BIT
ROR R4 ; GET NEXT BIT

```

```

3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785 015322 000004
3786 015324 012737 000012 001200
3787 015332 013702 001270
3788 015336 012762 100000 000000
3789 015344 012762 000040 000026
3790 015352 012762 055216 000004
3791 015360 012762 177777 000002
3792 015366 012762 000001 000006
3793 015374 012762 000100 000020
3794 015402 012762 000023 000000
3795 015410 012700 000426
3796
3797 015414 012762 000440 000026 1$:
3798 015422 012762 000040 000026
3799 015430 005300
3800 015432 001370
3801 015434 012705 000002
3802 015440 012703 053414
3803 015444 012737 000023 003200
3804 015452 012737 000300 003210
3805 015460 005037 003214
3806 015464 012737 022040 003224
3807 015472 005037 003310
3808 015476 012762 000140 000026 5$:
3809 015504 012762 000040 000026
3810 015512 005037 003254
3811 015516 005037 003256
3812 015522 012700 000377
3813 015526 004737 037026 10$:
3814 015532 005300
3815 015534 001374
3816 015536 012737 000001 003254
3817 015544 004737 037026
3818 015550 012701 000003
3819 015554 012304 12$:
3820 015556 012700 000020
3821 015562 013737 003254 15$:
3822 015570 006004

```

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046)
T26

02-DEC-77 10:20 PAGE 73
GOOD HEADER AND PREVIOUS BSE

SEQ 0073

3823	015572	103403				BCS	17\$	
3824	015574	005037	003254			CLR	PR.BIT	
3825	015600	000403				BR	18\$	
3826								
3827	015602	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
3828	015610	004737	037026		18\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
3829	015614	005300				DEC	RO	;CHECK IF READY FOR NEXT HEADER WORD
3830	015616	001361				BNE	15\$	;NO, CONTINUE
3831	015620	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
3832	015622	001354				BNE	12\$	;NO, CONTINUE
3833	015624	012700	000102			MOV	#66.,RO	;LOAD COUNT FOR GAP
3834								;PLUS 2 COUNTS FOR WRTGAT TO SET
3835	015630	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
3836	015636	005037	003254			CLR	PR.BIT	
3837	015642	004737	037026			JSR	PC,RDBIT	
3838	015646	005300				DEC	RO	;CHECK IF GAP IS FINISHED
3839	015650	001367				BNE	25\$	;NO, CONTINUE
3840	015652	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
3841	015660	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
3842	015666	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ERROR REG
3843	015674	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
3844	015702	001401				BEQ	30\$	;YES, CONTINUE
3845	015704	104114				ERROR	114	;CS1 INCORRECT
3846	015706	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
3847	015714	001401				BEQ	31\$	;YES, CONTINUE
3848	015716	104115				ERROR	115	;CS2 INCORRECT
3849	015720	023737	003214	003154	31\$:	CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
3850	015726	001401				BEQ	32\$	;YES, CONTINUE
3851	015730	104116				ERROR	116	;ERROR REG INCORRECT
3852	015732	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	;GET MR1
3853	015740	023737	003224	003164		CMP	E.MR1,T.MR1	;CHECK WRITE GATE CORRECT
3854	015746	001401				BEQ	34\$	;YES, CONTINUE
3855	015750	104117				ERROR	117	;MR1 INCORRECT
3856	015752	005237	003310		34\$:	INC	HDRCNT	;INCREMENT HEADER COUNT
3857	015756	012737	062040	003224		MOV	#WRTGAT!DMD!MEWD	;ECCW,E.MR1 ;LOAD EXPECTED MR1
3858	015764	005305				DEC	R5	;CHECK IF FINISHED
3859	015766	001402				BEQ	TST27	;YES, GO ON TO NEXT TEST
3860	015770	000137	015476			JMP	5\$	

```

3861
3862
3863 *****
3864 *TEST 27      GOOD HEADER AND PREVIOUS HVRC
3865 *
3866 *      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
3867 *      CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA OF
3868 *      ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 200,
3869 *      HEAD 0, SECTOR 1.  CLOCK THROUGH SEEK AND DRIVE CLEAR
3870 *      MESSAGES.  SIMULATE A SECTOR PULSE AND A HEADER OF THE
3871 *      FOLLOWING 3 WORDS WITH A BAD HEADER VRC:
3872 *
3873 *      000200
3874 *      140000
3875 *      140000
3876 *
3877 *      MAKE SURE NO ERROR IS REPORTED AND WRITE GATE DOES NOT
3878 *      SET.  SIMULATE A SECTOR PULSE AND A HEADER OF THE
3879 *      FOLLOWING 3 WORDS:

```

```

3879      ;*
3880      ;*      000200
3881      ;*      140001
3882      ;*      140201
3883      ;*
3884      ;*      MAKE SURE WRITE GATE SEES INDICATING THAT HEADER HAS
3885      ;*      BEEN RECOGNIZED.
3886      ;*
3887      ;*      *****
3888      ST27: SCOPE
3889      015774 000004      000012 001200      MOV      #10, $TIMES      ;DO 10. ITERATIONS
3890      015776 012737      000012 001200      MOV      $BASE, R2      ;LOAD RK611 BASE
3891      016004 013702      001270      MOV      #CCLR, RKCS1(R2) ;CLEAR RK611
3892      016010 012762      100000 000000      MOV      #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3893      016016 012762      000040 000026      MOV      #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3894      016024 012762      055216 000004      MOV      #-1, RKWC(R2) ;WORD COUNT -1
3895      016032 012762      177777 000002      MOV      #1, RKDA(R2) ;LOAD TRACK AND SECTOR
3896      016040 012762      000001 000006      MOV      #200, RKDCYL(R2) ;LOAD CYLINDER
3897      016046 012762      000020 000020      MOV      #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
3898      016054 012762      000023 000000      MOV      #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3899      016062 012700      000426      MOV      ;FOR SECTOR PULSE
3900      016066 012762      000440 000026 1$: MOV      #DMD!MCLK, RKMR1(R2)
3901      016074 012762      000040 000026      MOV      #DMD, RKMR1(R2)
3902      016102 005300      DEC      R0
3903      016104 001370      BNE     1$
3904      016106 012705      000002      MOV      #2, R5 ;LOAD HEADER COUNT
3905      016112 012703      053430      MOV      #HEAD6, R3 ;LOAD ADDRESS OF HEADERS
3906      016116 012737      000023 003200      MOV      #WRDATA, E.CS1 ;LOAD EXPECTED CS1
3907      016124 012737      000300 003210      MOV      #IR!OR, E.CS2 ;LOAD EXPECTED CS2
3908      016132 005037      003214      CLR      E.ER ;LOAD EXPECTED MR1
3909      016136 012737      022040 003224      MOV      #DMD!MEWD!ECCW, E.MR1 ;LOAD EXPECTED MR1
3910      016144 005037      003310      CLR      HDRCNT ;INITIALIZE HEADER COUNT
3911      016150 012762      000140 000026 5$: MOV      #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
3912      016156 012762      000040 000026      MOV      #DMD, RKMR1(R2)
3913      016164 005037      003254      CLR      PR.BIT ;GENERATE SYNCH
3914      016170 005037      003256      CLR      M1.BIT
3915      016174 012700      000377      MOV      #255, R0
3916      016200 004737      037026 10$: JSR      PC, RDBIT
3917      016204 005300      DEC      R0 ;CHECK IF SYNCH FINISHED
3918      016206 001374      BNE     10$
3919      016210 012737      000001 003254      MOV      #1, PR.BIT ;SIMULATE SYNCH BIT
3920      016216 004737      037026      JSR      PC, RDBIT
3921      016222 012701      000003      MOV      #3, R1 ;SIMULATE HEADER
3922      016226 012304      12$: MOV      (R3)+, R4 ;GET NEXT HEADER WORD
3923      016230 012700      000020      MOV      #16, R0 ;LOAD BITS PER WORD
3924      016234 013737      003254 003256 15$: MOV      PR.BIT, M1.BIT ;STORE PREVIOUS BIT
3925      016242 006004      ROR      R4 ;GET NEXT BIT
3926      016244 103403      BCS     17$
3927      016246 005037      003254      CLR      PR.BIT
3928      016252 000403      BR      18$
3929
3930      016254 012737      000001 003254 17$: MOV      #1, PR.BIT
3931      016262 004737      037026 18$: JSR      PC, RDBIT ;SIMULATE NEXT BIT
3932      016266 005300      DEC      R0 ;CHECK IF READY FOR NEXT HEADER WORD
3933      016270 001361      BNE     15$ ;NO, CONTINUE
3934      016272 005301      DEC      R1 ;CHECK IF FINISHED WITH HEADER

```

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046)
T27

02-DEC-77 10:20 PAGE 75
GOOD HEADER AND PREVIOUS HVRC

K06

SEQ 0075

```
3935 016274 001354 BNE 12$ ;NO, CONTINUE
3936 016276 012700 000102 MOV #66.,R0 ;LOAD COUNT FOR GAP
3937 ;PLUS 2 COUNTS FOR WRTGAT TO SET
3938 016302 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3939 016310 005037 003254 CLR PR.BIT
3940 016314 004737 037026 JSR PC,RDBIT
3941 016320 005300 DEC R0 ;CHECK IF GAP IS FINISHED
3942 016322 001367 BNE 25$ ;NO, CONTINUE
3943 016324 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3944 016332 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3945 016340 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
3946 016346 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3947 016354 001401 BEQ 30$ ;YES, CONTINUE
3948 016356 104120 ERROR 120 ;CS1 INCORRECT
3949 016360 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3950 016366 001401 BEQ 31$ ;YES, CONTINUE
3951 016370 104121 ERROR 121 ;CS2 INCORRECT
3952 016372 023737 003214 003154 31$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3953 016400 001401 BEQ 32$ ;YES, CONTINUE
3954 016402 104122 ERROR 122 ;ERROR REG INCORRECT
3955 016404 016237 000026 003164 32$: MOV RKMRI(R2),T.MRI ;GET MRI
3956 016406 023737 003224 003164 CMP E.MRI,T.MRI ;CHECK WRITE GATE CORRECT
3957 016408 001401 BEQ 34$ ;YES, CONTINUE
3958 016410 104123 ERROR 123 ;MRI INCORRECT
3959 016424 005237 003310 34$: INC HDRCNT ;INCREMENT HEADER COUNT
3960 016430 012737 062040 003224 MOV #WRTGAT!DMD!MEWD,ECCW,E.MRI ;LOAD EXPECTED MRI
3961 016436 005305 DEC R5 ;CHECK IF FINISHED
3962 016440 001402 BEQ TST30 ;YES, GO ON TO NEXT TEST
3963 016442 000137 JMP 5$
```

```
*****
*TEST 30 BAD SECTOR ERROR AND PREVIOUS HVRC
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A
* WRITE DATA OF ONE WORD TO AN RK06 IN 26 SECTOR
* FORMAT, CYLINDER 400, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A HEADER OF THE FOLLOWING 3 WORDS WITH A
* BAD HEADER VRC:
*
* 000400
* 140000
* 140000
*
* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND A
* HEADER CONSISTING OF THE FOLLOWING 3 WORDS:
*
* 000400
* 040001
* 040401
*
* MAKE SURE BAD SECTOR ERROR SETS AND HEADER VRC
* ERROR DOES NOT SET.
*
```

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 016446 000004
3993 016450 012737 000012 001200
3994 016456 013702 001270
3995 016462 012762 100000 000000
3996 016470 012762 000040 000026
3997 016476 012762 055216 000004
3998 016504 012762 177777 000002
3999 016512 012762 000001 000006
4000 016520 012762 000400 000020
4001 016526 012762 000023 000000
4002 016534 012700 000426
4003
4004 016540 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4005 016546 012762 000040 000026 MOV #DMD,RKMR1(R2)
4006 016554 005300 DEC RO
4007 016556 001370 BNE 1$
4008 016560 012705 000002 MOV #2,R5 ;LOAD HEADER COUNT
4009 016564 012703 053444 MOV #HEAD7,R3 ;LOAD ADDRESS OF HEADERS
4010 016570 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
4011 016576 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4012 016604 005037 003214 CLR E.ER ;LOAD EXPECTED MR1
4013 016610 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
4014 016616 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT
4015 016622 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4016 016630 012762 000040 000026 MOV #DMD,RKMR1(R2)
4017 016636 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4018 016642 005037 003256 CLR M1.BIT
4019 016646 012700 000377 MOV #255,RO
4020 016652 004737 037026 10$: JSR PC,RDBIT
4021 016656 005300 DEC RO ;CHECK IF SYNCH FINISHED
4022 016660 001374 BNE 10$
4023 016662 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4024 016670 004737 037026 JSR PC,RDBIT
4025 016674 012701 000003 MOV #3,R1 ;SIMULATE HEADER
4026 016700 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4027 016702 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
4028 016706 013737 003254 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4029 016714 006004 ROR R4 ;GET NEXT BIT
4030 016716 103403 BCS 17$
4031 016720 005037 003254 CLR PR.BIT
4032 016724 000403 BR 18$
4033
4034 016726 012737 000001 003254 17$: MOV #1,PR.BIT
4035 016734 004737 037026 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4036 016740 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
4037 016742 001361 BNE 15$ ;NO, CONTINUE
4038 016744 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4039 016746 001354 BNE 12$ ;NO, CONTINUE
4040 016750 012700 000102 MOV #66.,RO ;LOAD COUNT FOR GAP
4041 ; PLUS 2 COUNTS FOR WRTGAT TO SET
4042 016754 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4043 016762 005037 003254 CLR PR.BIT
4044 016766 004737 037026 JSR PC,RDBIT
4045 016772 005300 DEC RO ;CHECK IF GAP IS FINISHED
4046 016774 001367 BNE 25$ ;NO, CONTINUE
```

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 77
T30 BAD SECTOR ERROR AND PREVIOUS HVRC

SEQ 0077

4047	016776	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
4048	017004	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:GET CS2
4049	017012	016237	000014	003154		MOV	RKER(R2),T.ER	:GET ERROR REG
4050	017020	023737	003200	003140		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4051	017026	001401				BEQ	30\$	:YES, CONTINUE
4052	017030	104124				ERROR	124	:CS1 INCORRECT
4053	017032	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4054	017040	001401				BEQ	31\$	:YES, CONTINUE
4055	017042	104125				ERROR	125	:CS2 INCORRECT
4056	017044	023737	003214	003154	31\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
4057	017052	001401				BEQ	32\$	:YES, CONTINUE
4058	017054	104126				ERROR	126	:ERROR REG INCORRECT
4059	017056	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	:GET MR1
4060	017064	023737	003224	003164		CMP	E.MR1,T.MR1	:CHECK WRITE GATE CORRECT
4061	017072	001401				BEQ	34\$	:YES, CONTINUE
4062	017074	104127				ERROR	127	:MR1 INCORRECT
4063	017076	005237	003310		34\$:	INC	HDRCNT	:INCREMENT HEADER COUNT
4064	017102	012737	000200	003214		MOV	#BSE,E.ER	:LOAD EXPECTER ERROR REG
4065	017110	005305				DEC	RS	:CHECK IF FINISHED
4066	017112	001402				BEQ	TST31	:;YES, GO ON TO NEXT TEST
4067	017114	000137	016622			JMP	5\$	

*TEST 31 HEADER VRC AND PREVIOUS BSE

*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A
* WRITE DATA OR ONE WORD TO AN RK06 IN 26 SECTOR
* FORMAT, CYLINDER 140, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
* AND A HEADER OF THE FOLLOWING 3 WORDS WITH A HEADER
* VRC ERROR:

*
* 000140
* 040000
* 040140

*
* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. STIMULATE A SECTOR PULSE AND A
* HEADER CONSISTING OF THE FOLLOWING THREE WORDS:

*
* 000140
* 140001
* 140101

*
* MAKE SURE HEADER VRC ERROR SETS AND BAD SECTOR
* ERROR DOES NOT SET.

*ST31: SCOPE

MOV	#10,\$TIMES	:DO 10. ITERATIONS
MOV	\$BASE,R2	:LOAD RK611 BASE
MOV	#CLR,RKCS1(R2)	:CLEAR RK611
MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
MOV	#BUFF,RKBA(R2)	:LOAD DUMMY BUS ADDRESS
MOV	#-1,RKWC(R2)	:WORD COUNT -1

4096	017120	000004		
4097	017122	012737	000012	001200
4098	017130	013702	001270	
4099	017134	012762	100000	000000
4100	017142	012762	000040	000026
4101	017150	012762	055216	000004
4102	017156	012762	177777	000002

4103	017164	012762	000001	000006		MOV	#1,RKDA(R2)	;LOAD TRACK AND SECTOR
4104	017172	012762	000140	000020		MOV	#140,RKDCYL(R2)	;LOAD CYLINDER
4105	017200	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA
4106	017206	012700	000426			MOV	#69,*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL READY
4107								FOR SECTOR PULSE
4108	017212	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4109	017220	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4110	017226	005300				DEC	R0	
4111	017230	001370				BNE	1\$	
4112	017232	012705	000002			MOV	#2,R5	;LOAD HEADER COUNT
4113	017236	012703	053460			MOV	#HEAD8,R3	;LOAD ADDRESS OF HEADERS
4114	017242	012737	000023	003200		MOV	#WRDATA,E.CS1	;LOAD EXPECTED CS1
4115	017250	012737	000300	003210		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
4116	017256	005037	003214			CLR	E.ER	;LOAD EXPECTED MR1
4117	017262	012737	022040	003224		MOV	#DMD!MEWD!ECCW,E.MR1	;LOAD EXPECTED MR1
4118	017270	005037	003310			CLR	HDRCNT	;INITIALIZE HEADER COUNT
4119	017274	012762	000140	000026	5\$:	MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
4120	017302	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4121	017310	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
4122	017314	005037	003256			CLR	M1.BIT	
4123	017320	012700	000377			MOV	#255,R0	
4124	017324	004737	037026		10\$:	JSR	PC,R0BIT	
4125	017330	005300				DEC	R0	;CHECK IF SYNCH FINISHED
4126	017332	001374				BNE	10\$	
4127	017334	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
4128	017342	004737	037026			JSR	PC,R0BIT	
4129	017346	012701	000003			MOV	#3,R1	;SIMULATE HEADER
4130	017352	012304			12\$:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
4131	017354	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
4132	017360	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
4133	017366	006004				ROR	R4	;GET NEXT BIT
4134	017370	103403				BCS	17\$	
4135	017372	005037	003254			CLR	PR.BIT	
4136	017376	000403				BR	18\$	
4137								
4138	017400	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
4139	017406	004737	037026		18\$:	JSR	PC,R0BIT	;SIMULATE NEXT BIT
4140	017412	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
4141	017414	001361				BNE	15\$	NO, CONTINUE
4142	017416	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
4143	017420	001354				BNE	12\$	NO, CONTINUE
4144	017422	012700	000102			MOV	#66,R0	;LOAD COUNT FOR GAP
4145								PLUS 2 COUNTS FOR WRTGAT TO SET
4146	017426	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
4147	017434	005037	003254			CLR	PR.BIT	
4148	017440	004737	037026			JSR	PC,R0BIT	
4149	017444	005300				DEC	R0	;CHECK IF GAP IS FINISHED
4150	017446	001367				BNE	25\$	NO, CONTINUE
4151	017450	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
4152	017456	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
4153	017464	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ERROR REG
4154	017472	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4155	017500	001401				BEQ	30\$	YES, CONTINUE
4156	017502	104130				ERROR	130	CS1 INCORRECT
4157	017504	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4158	017512	001401				BEQ	31\$	YES, CONTINUE

```

4159 017514 104131          ERROR 131          ;CS2 INCORRECT
4160 017516 023737 003214 003154 31$: CMP      E.ER,T.ER          ;CHECK ERROR REG CORRECT
4161 017524 001401          BEQ      32$          ;YES, CONTINUE
4162 017526 104132          ERROR 132          ;ERROR REG INCORRECT
4163 017530 016237 000026 003164 32$: MOV      RKMR1(R2),T.MR1 ;GET MR1
4164 017536 023737 003224 003164      CMP      E.MR1,T.MR1 ;CHECK WRITE GATE CORRECT
4165 017544 001401          BEQ      34$          ;YES, CONTINUE
4166 017546 104133          ERROR 133          ;MR1 INCORRECT
4167 017550 005237 003310      INC      HDRCNT          ;INCREMENT HEADER COUNT
4168 017554 012737 000400 003214      MOV      #HVRC,E.ER ;LOAD EXPECTER ERROR REG
4169 017562 005305          DEC      R5              ;CHECK IF FINISHED
4170 017564 001402          BEQ      TST32          ;YES, GO ON TO NEXT TEST
4171 017566 000137 017274      JMP      5$
4172
4173 *****
4174 *TEST 32          OPI AND HVRC ON LAST HEADER
4175 *
4176 *          CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4177 *          PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
4178 *          DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
4179 *          CYLINDER 240, HEAD 0, SECTOR 1.  CLOCK THROUGH
4180 *          SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
4181 *          AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:
4182 *
4183 *          000240
4184 *          140000
4185 *          140240
4186 *
4187 *          MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
4188 *          DOES NOT SET.  SIMULATE A SECTOR PULSE AND A
4189 *          HEADER CONSISTING OF THE FOLLOWING THREE WORDS:
4190 *
4191 *          000240
4192 *          140000
4193 *          140040
4194 *
4195 *          MAKE SURE HEADER VRC AND OPI ERROR SET.
4196 *
4197 *****
4198 017572 000004          TST32: SCOPE
4199 017574 012737 000012 001200      MOV      #10,$TIMES          ;DO 10. ITERATIONS
4200 017602 013702 001270          MOV      $BASE,R2          ;LOAD RK611 BASE
4201 017606 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4202 017614 012700 002500          MOV      #2500,R0          ;SET COUNT FOR STALL
4203 017620 005300      2$: DEC      R0              ;DEC COUNT
4204 017622 001376      BNE      2$              ;LOOP UNTIL 0
4205 017624 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4206 017632 012762 055216 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
4207 017640 012762 177777 000002      MOV      #-1,RKWC(R2) ;WORD COUNT = 1
4208 017646 012762 000240 000020      MOV      #240,RKDCYL(R2) ;LOAD CYLINDER
4209 017654 012762 000001 000006      MOV      #1,RKDA(R2) ;LOAD TRACK AND SECTOR
4210 017662 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
4211 017670 012700 000426          MOV      #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4212 *          FOR SECTOR PULSE
4213 017674 012762 000440 000026 1$: MOV      #DMD!MCLK,RKMR1(R2)
4214 017702 012762 000040 000026      MOV      #DMD,RKMR1(R2)

```


4215	017710	005300		DEC	R0	
4216	017712	001370		BNE	1\$	
4217	017714	012705	000040	MOV	#32, R5	; LOAD HEADER COUNT
4218	017720	012703	054074	MOV	#OPI2, R3	; LOAD ADDRESS OF HEADERS
4219	017724	012737	000023	MOV	#WRDATA, E.CS1	; LOAD EXPECTED CS1
4220	017732	012737	000300	MOV	#IR!OR, E.CS2	; LOAD EXPECTED CS2
4221	017740	005037	003214	CLR	E, ER	; LOAD EXPECTED ERROR REG
4222	017744	012737	022040	MOV	#DMD!MEWD!ECCW, E.MR1	; LOAD EXPECTED MRI
4223	017752	005037	003310	CLR	HDRCNT	; INITIALIZE HEADER COUNT
4224	017756	012762	000140	MOV	#DMD!MSP, RKMR1(R2)	; SIMULATE SECTOR PULSE
4225	017764	012762	000040	MOV	#DMD, RKMR1(R2)	
4226	017772	022737	000037	CMP	#31, HDRCNT	; CHECK IF ALL HEADERS DONE
4227	020000	001460		BEQ	26\$	; YES - SKIP TO ERROR TEST
4228	020002	005037	003254	CLR	PR.BIT	; GENERATE SYNCH
4229	020006	005037	003256	CLR	M1.BIT	
4230	020012	012700	000377	MOV	#255, R0	
4231	020016	004737	037026	JSR	PC, RDBIT	
4232	020022	005300		DEC	R0	; CHECK IF SYNCH FINISHED
4233	020024	001374		BNE	10\$	
4234	020026	012737	000001	MOV	#1, PR.BIT	; SIMULATE SYNCH BIT
4235	020034	004737	037026	JSR	PC, RDBIT	
4236	020040	012701	000003	MOV	#3, R1	; SIMULATE OPI
4237	020044	012304		MOV	(R3)+, R4	; GET NEXT HEADER WORD
4238	020046	012700	000020	MOV	#16, R0	; LOAD BITS PER WORD
4239	020052	013737	003254	MOV	PR.BIT, M1.BIT	; STORE PREVIOUS BIT
4240	020060	006004		ROR	R4	; GET NEXT BIT
4241	020062	103403		BCS	17\$	
4242	020064	005037	003254	CLR	PR.BIT	
4243	020070	000403		BR	18\$	
4244						
4245	020072	012737	000001	MOV	#1, PR.BIT	
4246	020100	004737	037026	JSR	PC, RDBIT	; SIMULATE NEXT BIT
4247	020104	005300		DEC	R0	; CHECK IF READY FOR NEXT HEADER WORD
4248	020106	001361		BNE	15\$	; NO, CONTINUE
4249	020110	005301		DEC	R1	; CHECK IF FINISHED WITH HEADER
4250	020112	001354		BNE	12\$	; NO, CONTINUE
4251	020114	012700	000100	MOV	#64, R0	; LOAD COUNT FOR GAP
4252	020120	013737	003254	MOV	PR.BIT, M1.BIT	; SIMULATE GAP
4253	020126	005037	003254	CLR	PR.BIT	
4254	020132	004737	037026	JSR	PC, RDBIT	
4255	020136	005300		DEC	R0	; CHECK IF GAP IS FINISHED
4256	020140	001367		BNE	25\$	; NO, CONTINUE
4257	020142	016237	000000	MOV	RKCS1(R2), T.CS1	; GET CS1
4258	020150	016237	000010	MOV	RKCS2(R2), T.CS2	; GET CS2
4259	020156	016237	000014	MOV	RKER(R2), T.ER	; GET ERROR REG.
4260	020164	023737	003200	CMP	E.CS1, T.CS1	; CHECK CS1 CORRECT
4261	020172	001401		BEQ	30\$	; YES, CONTINUE
4262	020174	104100		ERROR	100	; CS1 INCORRECT
4263	020176	023737	003210	CMP	E.CS2, T.CS2	; CHECK CS2 CORRECT
4264	020204	001401		BEQ	31\$	; YES, CONTINUE
4265	020206	104101		ERROR	101	; CS2 INCORRECT
4266	020210	023737	003214	CMP	E.ER, T.ER	; CHECK ERROR REG CORRECT
4267	020216	001401		BEQ	32\$	; YES, CONTINUE
4268	020220	104102		ERROR	102	; ERROR REG INCORRECT
4269	020222	016237	000026	MOV	RKMR1(R2), T.MR1	; GET MR1
4270	020230	023737	003224	CMP	E.MR1, T.MR1	; CHECK TO MAKE SURE WRITE GATE DID NOT SET

007

CZR6080 RK611 DSKLS CTRL PRTH
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 81
T32 OPI AND HVRC ON LAST HEADER

SEQ 0081

```

4271 020236 001401 BEQ 34$ ;YES, CONTINUE
4272 020240 104103 ERROR 103 ;MRI INCORRECT
4273 020242 005237 003310 34$: INC HDRCNT ;INCREMENT HEADER COUNT
4274 020246 022737 000037 003310 CMP #31.,HDRCNT ;CHECK IF LAST HEADER
4275 020254 001011 BNE 35$ ;NO, CHECK IF FINSHED
4276 020256 012737 020400 003214 MOV #OPI:HVRC,E.ER ;LOAD ERROR BIT
4277 020264 042737 000001 003200 BIC #GO,E.CS1 ;ADJUST E.CS1 FOR END OF OP CONTENTS
4278 020272 052737 100200 003200 BIS #RDY:CERR,E.CS1
4279 020300 005305 35$: DEC R5 ;CHECK IF FINISHED
4280 020302 001402 BEQ 37$ ;YES - SKIP
4281 020304 000137 017756 JMP 5$ ;DO NEXT SECTOR
4282 020310 012762 100000 000000 37$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4283 020316 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4284 020324 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;CS2
4285 020332 016237 000014 003154 MOV RKER(R2),T.ER ;ER
4286 020340 012737 000200 003200 MOV #RDY,E.CS1 ;SET EXPECTED CS1
4287 020346 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4288 020354 001401 BEQ TST33 ;GO TO NEXT TEST
4289 020356 104153 ERROR 153

*****
*TEST 33 OPI AND PREVIOUS HEADER VRC (PART 1)
*****
*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 300, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A
* SECTOR PULSE AND 30 HEADERS CONSISTING OF THE
* FOLLOWING 3 WORDS:
*
* 000300
* 140000
* 140300
*
* THEN SIMULATE A 3 WORD HEADER CONSISTING ON
* THE FOLLOWING DATA:
*
* 000300
* 140000
* 140200
*
* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND A
* HEADER CONSISTING OF THE FOLLOWING THREE WORDS:
*
* 000300
* 140000
* 140300
*
* MAKE SURE HEADER VRC AND OPI ERRORS SET.
*****
4324 020360 000004 TST33: SCOPE
4325 020362 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
4326 020370 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE

```

4327	020374	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
4328	020402	012700	002500			MOV	#2500,R0	;SET COUNT FOR STALL
4329	020406	005300			2\$:	DEC	R0	;DEC COUNT
4330	020410	001376				BNE	2\$	;LOOP UNTIL 0
4331	020412	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
4332	020420	012762	055216	000004		MOV	#BUFF,RKBA(R2)	;LOAD DUMMY BUFFER ADDRESS
4333	020426	012762	177777	000002		MOV	#-1,RKWC(R2)	;WORD COUNT = 1
4334	020434	012762	000300	000020		MOV	#300,RKDCYL(R2)	;LOAD CYLINDER
4335	020442	012762	000001	000006		MOV	#1,RKDA(R2)	;LOAD TRACK AND SECTOR
4336	020450	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA
4337	020456	012700	000426			MOV	#6*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL READY
4338								;FOR SECTOR PULSE
4339	020462	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4340	020470	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4341	020476	005300				DEC	R0	
4342	020500	001370				BNE	1\$	
4343	020502	012705	000040			MOV	#32,R5	;LOAD HEADER COUNT
4344	020506	012703	054374			MOV	#OPI3,R3	;LOAD ADDRESS OF HEADERS
4345	020512	012737	000023	003200		MOV	#WRDATA,E.CS1	;LOAD EXPECTED CS1
4346	020520	012737	000300	003210		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
4347	020526	005037	003214			CLR	E.ER	;LOAD EXPECTED ERROR REG
4348	020532	012737	022040	003224		MOV	#DMD!MEWD!ECCW,E.MR1	;LOAD EXPECTED MR1
4349	020540	005037	003310			CLR	HDRCNT	;INITIALIZE HEADER COUNT
4350	020544	012762	000140	000026	5\$:	MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
4351	020552	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4352	020560	022737	000037	003310		CMP	#31,HDRCNT	;CHECK IF ALL HEADERS DONE
4353	020566	001460				BEQ	26\$	;YES - SKIP TO ERROR TEST
4354	020570	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
4355	020574	005037	003256			CLR	M1.BIT	
4356	020600	012700	000377			MOV	#255,R0	
4357	020604	004737	037026		10\$:	JSR	PC,RDBIT	
4358	020610	005300				DEC	R0	;CHECK IF SYNCH FINISHED
4359	020612	001374				BNE	10\$	
4360	020614	012737	000001	003254		MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
4361	020622	004737	037026			JSR	PC,RDBIT	
4362	020626	012701	000003			MOV	#3,R1	;SIMULATE OPI
4363	020632	012304			12\$:	MOV	(R3)+,R4	;GET NEXT HEADER WORD
4364	020634	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
4365	020640	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
4366	020646	006004				ROR	R4	;GET NEXT BIT
4367	020650	103403				BCS	17\$	
4368	020652	005037	003254			CLR	PR.BIT	
4369	020656	000403				BR	18\$	
4370								
4371	020660	012737	000001	003254	17\$:	MOV	#1,PR.BIT	
4372	020666	004737	037026		18\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
4373	020672	005300				DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
4374	020674	001361				BNE	15\$	;NO, CONTINUE
4375	020676	005301				DEC	R1	;CHECK IF FINISHED WITH HEADER
4376	020700	001354				BNE	12\$	;NO, CONTINUE
4377	020702	012700	000100			MOV	#64,R0	;LOAD COUNT FOR GAP
4378	020706	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
4379	020714	005037	003254			CLR	PR.BIT	
4380	020720	004737	037026			JSR	PC,RDBIT	
4381	020724	005300				DEC	R0	;CHECK IF GAP IS FINISHED
4382	020726	001367				BNE	25\$	;NO, CONTINUE

4383	020730	016237	000000	003140	26\$:	MOV	RKCS1(R2),T.CS1	;GET CS1
4384	020736	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
4385	020744	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ERROR REG.
4386	020752	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4387	020760	001401				BEQ	30\$	;YES, CONTINUE
4388	020762	104104				ERROR	104	;CS1 INCORRECT
4389	020764	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4390	020772	001401				BEQ	31\$	;YES, CONTINUE
4391	020774	104105				ERROR	105	;CS2 INCORRECT
4392	020776	023737	003214	003154	31\$:	CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
4393	021004	001401				BEQ	32\$	;YES, CONTINUE
4394	021006	104106				ERROR	106	;ERROR REG INCORRECT
4395	021010	016237	000026	003164	32\$:	MOV	RKMR1(R2),T.MR1	;GET MR1
4396	021016	023737	003224	003164		CMP	E.MR1,T.MR1	;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4397	021024	001401				BEQ	34\$	;YES, CONTINUE
4398	021026	104107				ERROR	107	;MR1 INCORRECT
4399	021030	005237	003310		34\$:	INC	HDRCNT	;INCREMENT HEADER COUNT
4400	021034	022737	000037	003310		CMP	#31.,HDRCNT	;CHECK IF LAST HEADER
4401	021042	001011				BNE	35\$	;NO, CHECK IF FINISHED
4402	021044	012737	020400	003214		MOV	#OPI:HVRG,E.ER	;LOAD ERROR BIT
4403	021052	042737	000001	003200		BIC	#GO,E.CS1	;ADJUST E.CS1 FOR END OF OP CONTENTS
4404	021060	052737	100200	003200		BIS	#RDY:CERR,E.CS1	
4405	021066	005305			35\$:	DEC	R5	;CHECK IF FINISHED
4406	021070	001402				BEQ	37\$	;YES - SKIP
4407	021072	000137	020544			JMP	5\$	;DO NEXT SECTOR
4408	021076	012762	100000	000000	37\$:	MOV	#CCLR,RKCS1(R2)	;CLEAR CONTROLLER
4409	021104	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
4410	021112	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;GET CS2
4411	021120	016237	000014	003154		MOV	RKER(R2),T.ER	;GET ER
4412	021126	012737	000200	003200		MOV	#RDY,E.CS1	;SET EXPECTED CS1
4413	021134	023737	003140	003200		CMP	T.CS1,E.CS1	;CHECK IF CORRECT
4414	021142	001401				BEQ	TST34	;GO TO NEXT TEST
4415	021144	104153				ERROR	153	

*TEST 34 OPI AND PREVIOUS HEADER VRC (PART 2)

*
* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
* DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
* CYLINDER 40, HEAD 0, SECTOR 1. CLOCK THROUGH
* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
* PULSE AND A HEADER CONSISTING OF THE FOLLOWING
* THREE WORDS HAVING A BAD HEADER VRC:

*
* 000040
* 140000
* 140000
*

*
* MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
* DOES NOT SET. SIMULATE A SECTOR PULSE AND 31
* HEADERS CONSISTING OF THE FOLLOWING THREE WORDS:

*
* 000040
* 140000
* 140040
*

438

```
4439      ;*
4440      ;*      MAKE SURE HEADER VRC AND OPI ERRORS SET.
4441      ;*
4442      ;*****
4443      ST34: SCOPE
4444      MOV     #10, $TIMES      ;DO 10. ITERATIONS
4445      MOV     $BASE, R2       ;LOAD RK611 BASE
4446      MOV     #CCLR, RKCS1(R2) ;CLEAR RK611
4447      MOV     #2500, R0       ;SET COUNT FOR STALL
4448      2$:     DEC     R0       ;DEC COUNT
4449      BNE     2$             ;LOOP UNTIL 0
4450      MOV     #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4451      MOV     #BUFF, RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
4452      MOV     #-1, RKWC(R2)   ;WORD COUNT = 1
4453      MOV     #40, RKDCYL(R2) ;LOAD CYLINDER
4454      MOV     #1, RKDA(R2)    ;LOAD TRACK AND SECTOR
4455      MOV     #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
4456      MOV     #69, #4+2, R0   ;ISSUE ENOUGH CLOCKS UNTIL READY
4457      ;           FOR SECTOR PULSE
4458      1$:     MOV     #DMD, MCLK, RKMR1(R2)
4459      MOV     #DMD, RKMR1(R2)
4460      DEC     R0
4461      BNE     1$
4462      MOV     #32, R5         ;LOAD HEADER COUNT
4463      MOV     #OPI4, R3       ;LOAD ADDRESS OF HEADERS
4464      MOV     #WRDATA, E.CS1   ;LOAD EXPECTED CS1
4465      MOV     #IR1OR, E.CS2   ;LOAD EXPECTED CS2
4466      CLR     E.ER           ;LOAD EXPECTED ERROR REG
4467      MOV     #DMD, MEWD, ECCW, E.MR1 ;LOAD EXPECTED MR1
4468      CLR     HDRCNT         ;INITIALIZE HEADER COUNT
4469      5$:     MOV     #DMD, MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
4470      MOV     #DMD, RKMR1(R2)
4471      CMP     #31, HDRCNT    ;CHECK IF ALL HEADERS DONE
4472      BEQ     26$           ;YES - SKIP TO ERROR TEST
4473      CLR     PR.BIT         ;GENERATE SYNCH
4474      CLR     M1.BIT
4475      MOV     #255, R0
4476      10$:    JSR     PC, RDBIT
4477      DEC     R0             ;CHECK IF SYNCH FINISHED
4478      BNE     10$
4479      MOV     #1, PR.BIT     ;SIMULATE SYNCH BIT
4480      JSR     PC, RDBIT
4481      MOV     #3, R1         ;SIMULATE OPI
4482      12$:    MOV     (R3)+, R4 ;GET NEXT HEADER WORD
4483      MOV     #16, R0        ;LOAD BITS PER WORD
4484      15$:    MOV     PR.BIT, M1.BIT ;STORE PREVIOUS BIT
4485      FOR     R4             ;GET NEXT BIT
4486      JCS     17$
4487      CLR     PR.BIT
4488      BR      18$
4489
4490      17$:    MOV     #1, PR.BIT
4491      18$:    JSR     PC, RDBIT ;SIMULATE NEXT BIT
4492      DEC     R0             ;CHECK IF READY FOR NEXT HEADER WORD
4493      BNE     15$           ;NO, CONTINUE
4494      DEC     R1             ;CHECK IF FINISHED WITH HEADER
```

H07

CZR6080 RK611 DSKLS CTRL PRY4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046)
T3402-DEC-77 10:20 PAGE 85
OPI AND PREVIOUS HEADER VRC (PART 2)

SEQ 0085

4495	021466	001354				BNE	12\$		;NO, CONTINUE
4496	021470	012700	000100			MOV	#64, R0		;LOAD COUNT FOR GAP
4497	021474	013737	003254	003256	25\$:	MOV	PR.BIT, M1.BIT		;SIMULATE GAP
4498	021502	005037	003254			CLR	PR.BIT		
4499	021506	004737	037026			JSR	PC, R0BIT		
4500	021512	005300				DEC	R0		;CHECK IF GAP IS FINISHED
4501	021514	001367				BNE	25\$		;NO, CONTINUE
4502	021516	016237	000000	003140	26\$:	MOV	RKCS1(R2), T.CS1		;GET CS1
4503	021524	016237	000010	003150		MOV	RKCS2(R2), T.CS2		;GET CS2
4504	021532	016237	000014	003154		MOV	RKER(R2), T.ER		;GET ERROR REG.
4505	021540	023737	003200	003140		CMP	E.CS1, T.CS1		;CHECK CS1 CORRECT
4506	021546	001401				BEQ	30\$		;YES, CONTINUE
4507	021550	104110				ERROR	110		;CS1 INCORRECT
4508	021552	023737	003210	003150	30\$:	CMP	E.CS2, T.CS2		;CHECK CS2 CORRECT
4509	021560	001401				BEQ	31\$		;YES, CONTINUE
4510	021562	104111				ERROR	111		;CS2 INCORRECT
4511	021564	023737	003214	003154	31\$:	CMP	E.ER, T.ER		;CHECK ERROR REG CORRECT
4512	021572	001401				BEQ	32\$		;YES, CONTINUE
4513	021574	104112				ERROR	112		;ERROR REG INCORRECT
4514	021576	016237	000026	003164	32\$:	MOV	RKMR1(R2), T.MR1		;GET MR1
4515	021604	023737	003224	003164		CMP	E.MR1, T.MR1		;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4516	021612	001401				BEQ	34\$		;YES, CONTINUE
4517	021614	104113				ERROR	113		;MR1 INCORRECT
4518	021616	005237	003310		34\$:	INC	HDRCNT		;INCREMENT HEADER COUNT
4519	021622	022737	000037	003310		CMP	#31, HDRCNT		;CHECK IF LAST HEADER
4520	021-30	001011				BNE	35\$		;NO, CHECK IF FINISHED
4521	021632	012737	020400	003214		MOV	#OPI:HVRC, E.ER		;LOAD ERROR BIT
4522	021640	042737	000001	003200		BIC	#GO, E.CS1		;ADJUST E.CS1 FOR END OF OP CONTENTS
4523	021646	052737	100200	003200		BIS	#RDY:CERR, E.CS1		
4524	021654	005305			35\$:	DEC	R5		;CHECK IF FINISHED
4525	021656	001402				BEQ	37\$		;YES - SKIP
4526	021660	000137	021332			JMP	5\$		;DO NEXT SECTOR
4527	021664	012762	100000	000000	37\$:	MOV	#CCLR, RKCS1(R2)		;CLEAR CONTROLLER
4528	021672	016237	000000	003140		MOV	RKCS1(R2), T.CS1		;GET CS1
4529	021700	016237	000010	003150		MOV	RKCS2(R2), T.CS2		;GET CS2
4530	021706	016237	000014	003154		MOV	RKER(R2), T.ER		;GET ER
4531	021714	012737	000200	003200		MOV	#RDY, E.CS1		;SET EXPECTED CS1
4532	021722	023737	003140	003200		CMP	T.CS1, E.CS1		;CHECK IF CORRECT
4533	021730	001401				BEQ	TST35		;GO TO NEXT TEST
4534	021732	104153				ERROR	153		

```

*****
*TEST 35          BSE AND CONTROLLER ERROR
*
*   CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
*   PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
*   OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT.
*   CYLINDER 0. HEAD 0. SECTOR 0.  CLOCK THROUGH SEEK
*   AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
*   AND A HEADER WITH A BSE ERROR.  MAKE SURE CONTROLLER
*   ERROR AND HVRC SET.  CLEAR CONTROLLER AND MAKE SURE
*   CONTROLLER ERROR RESETS.
*****
*ST35:  SCOPE

```

4550 021734 000004

4551	021736	012737	000010	001200		MOV	#10, \$TIMES	:: DO 10 ITERATIONS
4552	021744	013702	001270			MOV	\$BASE, R2	:: LOAD RK611 BASE
4553	021750	012762	100000	000000		MOV	#CCLR, RKCS1(R2)	:: CLEAR RK611
4554	021756	012762	000040	000026		MOV	#DMD, RKMR1(R2)	:: PUT RK611 IN DIAGNOSTIC MODE
4555	021764	012762	055216	000004		MOV	#BJFF, RKBA(R2)	:: LOAD DUMMY BUS ADDRESS
4556	021772	012762	177777	000002		MOV	#-1, RKWC(R2)	:: WORD COUNT=1
4557	022000	012762	000000	000020		MOV	#0, RKDCYL(R2)	:: LOAD CYLINDER 110MS
4558	022006	012762	000000	000006		MOV	#0, RKDA(R2)	:: LOAD TRACK AND SECTOR
4559	022014	012762	000023	000000		MOV	#WRDATA, RKCS1(R2)	:: ISSUE COMMAND
4560	022022	012700	000426			MOV	#69, *4+2, R0	:: ISSUE ENOUGH CLOCKS UNTIL READY FOR SECTOR PULSE.
4561								
4562	022026	012762	000440	000026	1\$:	MOV	#DMD!MCLK, RKMR1(R2)	
4563	022034	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4564	022042	005300				DEC	R0	
4565	022044	001370				BNE	1\$	
4566	022046	012762	000140	000026		MOV	#DMD!MSP, RKMR1(R2)	:: SIMULATE SECTOR PULSE
4567	022054	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4568	022062	005037	003254			CLR	PR.BIT	:: GENERATE SYNCH
4569	022066	005037	003256			CLR	M1.BIT	
4570	022072	012700	000377			MOV	#255, R0	
4571	022076	004737	037026		5\$:	JSR	PC, RDBIT	
4572	022102	005300				DEC	R0	:: CHECK IF SYNCH FINISHED
4573	022104	001374				BNE	5\$	
4574	022106	012737	000001	003254		MOV	#1, PR.BIT	:: SIMULATE SYNCH
4575	022114	004737	037026			JSR	PC, RDBIT	
4576	022120	012703	053372			MOV	#HEAD2, R3	:: LOAD HEADER
4577	022124	012701	000003			MOV	#3, R1	:: LOAD WORDS PER HEADER
4578	022130	012304			10\$:	MOV	(R3)+, R4	:: GET NEXT WORD
4579	022132	012700	000020			MOV	#16, R0	:: LOAD BITS PER WORD
4580	022136	013737	003254	003256	12\$:	MOV	PR.BIT, M1.BIT	:: STORE PREVIOUS BIT
4581	022144	006004				ROR	R4	:: GET NEXT BIT
4582	022146	103403				BCS	15\$	
4583	022150	005037	003254			CLR	PR.BIT	
4584	022154	000403				BR	16\$	
4585								
4586	022156	012737	000001	003254	15\$:	MOV	#1, PR.BIT	
4587	022164	004737	037026		16\$:	JSR	PC, RDBIT	:: SIMULATE NEXT BIT
4588	022170	005300				DEC	R0	:: CHECK IF READY FOR NEXT WORD
4589	022172	001361				BNE	12\$	:: NO, CONTINUE
4590	022174	005301				DEC	R1	:: CHECK IF FINISHED WITH HEADER
4591	022176	001354				BNE	10\$	:: NO, CONTINUE
4592	022200	012700	000101			MOV	#65, R0	
4593	022204	013737	003254	003256	20\$:	MOV	PR.BIT, M1.BIT	:: SIMULATE GAP
4594	022212	005037	003254			CLR	PR.BIT	
4595	022216	004737	037026			JSR	PC, RDBIT	
4596	022222	005300				DEC	R0	:: CHECK IF GAP FINISHED
4597	022224	001367				BNE	20\$	:: NO, CONTINUE
4598	022226	012700	021200			MOV	#2*(256.+(16.*256.)+64.), R0	:: LOAD COUNT UNTIL POSTAMBLE
4599	022232	012762	000440	000026	25\$:	MOV	#DMD!MCLK, RKMR1(R2)	
4600	022240	012762	000040	000026		MOV	#DMD, RKMR1(R2)	
4601	022246	005300				DEC	R0	
4602	022250	001370				BNE	25\$	
4603	022252	016237	000000	003140		MOV	RKCS1(R2), T.CS1	:: GET CS1
4604	022260	016237	000010	003150		MOV	RKCS2(R2), T.CS2	:: GET CS2
4605	022266	016237	000014	003154		MOV	RKER(R2), T.ER	:: GET ERROR REG
4606	022274	012737	100222	003200		MOV	#CERR!RDY!WRDATA<T.CS1>, E.CS1	:: LOAD EXPECTED CS1

J07

CZP6DB0 RK611 DSKLS CTRL PRT4
CZP6DB.P11 U2 DEC-77 10:00MACY11 30(1046)
T3502-DEC-77 10:20 PAGE 87
BSE AND CONTROLLER ERROR

SEQ 0087

4607	022302	012737	000300	003210		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
4608	022310	012737	000200	003214		MOV	#BSE,E.ER	;LOAD EXPECTED ERROR REG
4609	022316	023737	003200	003140		CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4610	022324	001401				BEG	30\$	;YES, CONTINUE
4611	022326	104136				ERROR	136	;CS1 INCORRECT
4612	022330	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4613	022336	001401				BEG	32\$	;YES, CONTINUE
4614	022340	104137				ERROR	137	;CS2 INCORRECT
4615	022342	023737	003214	003154	32\$:	CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
4616	022350	001401				BEG	35\$	;YES - SKIP
4617	022352	104140				ERROR	140	;ERROR REG INCORRECT
4618	022354	012762	100000	000000	35\$:	MOV	#CCLR,RKCS1(R2)	;CLEAR CONTROLLER
4619	022362	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
4620	022370	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;CS2
4621	022376	016237	000014	003154		MOV	RKER(R2),T.ER	;ER
4622	022404	012737	000200	003200		MOV	#200,E.CS1	;SET EXPECTED CS1
4623	022412	023737	003140	003200		CMP	T.CS1,E.CS1	;CHECK IF CORRECT
4624	022420	001401				BEG	TST36	;GO TO NEXT TEST
4625	022422	104153				ERROR	153	;ERROR DID NOT CLEAR
4626								
4627								
4628								
4629								
4630								
4631								
4632								
4633								
4634								
4635								
4636								
4637								
4638								
4639								
4640	022424	000004				TST36:	SCOPE	
4641	022426	012737	000012	001200		MOV	#10,\$TIMES	;DO 10. ITERATIONS
4642	022434	013702	001270			MOV	\$BASE,R2	;LOAD RK611 BASE
4643	022440	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
4644	022446	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
4645	022454	012762	055216	000004		MOV	#BUFF,RKBA(R2)	;LOAD DUMMY BUS ADDRESS
4646	022462	012762	177777	000002		MOV	#-1,RKWC(R2)	;WORD COUNT=1
4647	022470	012762	000300	000020		MOV	#300,RKDCYL(R2)	;LOAD CYLINDER 110MS
4648	022476	012762	000000	000006		MOV	#0,RKDA(R2)	;LOAD TRACK AND SECTOR
4649	022504	012762	000023	0000C0		MOV	#WRDATA,RKCS1(R2)	;ISSUE COMMAND
4650	022512	012700	000426			MOV	#69,*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL READY FOR SECTOR PULSE.
4651								
4652	022516	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4653	022524	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4654	022532	005300				DEC	R0	
4655	022534	001370				BNE	1\$	
4656	022536	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
4657	022544	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4658	022552	005037	003254			CLR	PR.BIT	;GENERATE SYNCH
4659	022556	005037	003256			CLR	M1.BIT	
4660	022562	012700	000377			MOV	#255,R0	
4661	022566	004737	037026		5\$:	JSR	PC,R0BIT	
4662	022572	005300				DEC	R0	;CHECK IF SYNCH FINISHED

K07

CZR6DB0 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 88
 CZR6DB.P11 02-DEC-77 10:00 T36 HVRC AND CONTROLLER ERROR

SEQ 0088

```

4663 022574 001374 BNE 5$
4664 022576 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4665 022604 004737 037026 JSR PC,RDBIT
4666 022610 012703 053474 MOV #HEAD10,R3 ;LOAD HEADER
4667 022614 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4668 022620 012304 10$: MOV (P3)+,R4 ;GET NEXT WORD
4669 022622 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4670 022626 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4671 022634 006004 ROR R4 ;GET NEXT BIT
4672 022636 103403 BCS 15$
4673 022640 005037 003254 CLR PR.BIT
4674 022644 000403 BR 16$
4675
4676 022646 012737 000001 003254 15$: MOV #1,PR.BIT
4677 022654 004737 037026 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4678 022660 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4679 022662 001361 BNE 12$ ;NO, CONTINUE
4680 022664 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4681 022666 001354 BNE 10$ ;NO, CONTINUE
4682 022670 012700 000101 MOV #65,R0
4683 022674 013737 003254 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4684 022702 005037 003254 CLR PR.BIT
4685 022706 004737 037026 JSR PC,RDBIT
4686 022712 005300 DEC R0 ;CHECK IF GAP FINISHED
4687 022714 001367 BNE 20$ ;NO, CONTINUE
4688 022716 012700 021200 MOV #2*(256.+<16.*256.>+64.),R0 ;LOAD COUNT UNTIL POSTAMBLE
4689 022722 012762 000440 25$: MOV #DMD!MCLK,RKMR1(R2)
4690 022730 012762 000040 000026 MOV #DMD,RKMR1(R2)
4691 022736 005300 DEC R0
4692 022740 001370 BNE 25$
4693 022742 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4694 022750 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4695 022756 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4696 022764 012737 100222 003200 MOV #CERR!RDY!WRDATA<T.C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4697 022772 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4698 023000 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4699 023006 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4700 023014 001401 BEQ 30$ ;YES, CONTINUE
4701 023016 104141 ERROR 141 ;CS1 INCORRECT
4702 023020 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4703 023026 001401 BEQ 32$ ;YES, CONTINUE
4704 023030 104142 ERROR 142 ;CS2 INCORRECT
4705 023032 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4706 023040 001401 BEQ 35$ ;YES - SKIP
4707 023042 104143 ERROR 143 ;ERROR REG INCORRECT
4708 023044 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4709 023052 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4710 023060 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4711 023066 016237 000014 003154 MOV RKER(R2),T.ER ;GET ER
4712 023074 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4713 023102 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4714 023110 001401 BEQ TST37 ;GO TO NEXT TEST
4715 023112 104153 ERROR 153 ;ERROR DID NOT CLEAR
4716
4717 ;*****
4718 ;*TEST 37 READ DATA AND HVRC ERROR

```

L07

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 89
T37 READ DATA AND HVRC ERROR

SEQ 0089

```

4719          *
4720          * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4721          * PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4722          * OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT.
4723          * CYLINDER 300, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4724          * AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
4725          * AND A HEADER WITH A HVRC ERROR.  MAKE SURE CONTROLLER
4726          * ERROR AND HVRC SET.  CLEAR CONTROLLER AND MAKE SURE
4727          * CONTROLLER ERROR RESETS.
4728          *
4729          * *****
4730          * ST37: SCOPE
4731          *
4732          * DO 10. ITERATIONS
4733          * $BASE,R2 ; LOAD RK611 BASE
4734          * $CCLR,RKCS1(R2) ; CLEAR RK611
4735          * $DMD,RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
4736          * $BUFF,RKBA(R2) ; LOAD DUMMY BUS ADDRESS
4737          * $-1,RKWC(R2) ; WORD COUNT=1
4738          * $300,RKDCYL(R2) ; LOAD CYLINDER 110MS
4739          * $0,RKDA(R2) ; LOAD TRACK AND SECTOR
4740          * $RDATA,RKCS1(R2) ; ISSUE COMMAND
4741          * $69,$4+2,R0 ; ISSUE ENOUGH CLOCKS UNTIL
4742          * ; READY FOR SECTOR PULSE.
4743          *
4744          * $DMD,$MCLK,RKMR1(R2)
4745          * $DMD,RKMR1(R2)
4746          * DEC R0
4747          * BNE 1$
4748          * $DMD,$MSP,RKMR1(R2) ; SIMULATE SECTOR PULSE
4749          * $DMD,RKMR1(R2)
4750          * CLR PR.BIT ; GENERATE SYNCH
4751          * CLR M1.BIT
4752          * MOV $255,R0
4753          * JSR PC,R0BIT
4754          * DEC R0 ; CHECK IF SYNCH FINISHED
4755          * BNE 5$
4756          * $1,PR.BIT ; SIMULATE SYNCH
4757          * JSR PC,R0BIT
4758          * MOV $HEAD10,R3 ; LOAD HEADER
4759          * MOV $3,R1 ; LOAD WORDS PER HEADER
4760          * MOV (R3)+,R4 ; GET NEXT WORD
4761          * MOV $16,R0 ; LOAD BITS PER WORD
4762          * PR.BIT,M1.BIT ; STORE PREVIOUS BIT
4763          * ROR R4 ; GET NEXT BIT
4764          * BCS 15$
4765          * CLR PR.BIT
4766          * BR 16$
4767          *
4768          * $1,PR.BIT
4769          * JSR PC,R0BIT ; SIMULATE NEXT BIT
4770          * DEC R0 ; CHECK IF READY FOR NEXT WORD
4771          * BNE 12$ ; NO, CONTINUE
4772          * DEC R1 ; CHECK IF FINISHED WITH HEADER
4773          * BNE 10$ ; NO, CONTINUE
4774          * MOV $65,R0
4775          * PR.BIT,M1.BIT ; SIMULATE GAP
4776          * CLR PR.BIT

```

```

4775 023376 004737 037026 JSR PC,RDBIT
4776 023402 005300 DEC R0 ;CHECK IF GAP FINISHED
4777 023404 001367 BNE 20$ ;NO CONTINUE
4778 023406 012700 021200 MOV #2*(256.+(16.*256.)+64.),R0 ;LOAD COUNT UNTIL POSTAMBLE
4779 023412 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4780 023420 012762 000040 000026 MOV #DMD,RKMR1(R2)
4781 023426 005300 DEC R0
4782 023430 001370 BNE 25$
4783 023432 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4784 023440 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4785 023446 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4786 023454 012737 100220 003200 MOV #CERR!RDY!RDATA<1C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4787 023462 012737 000100 003210 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4788 023470 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4789 023476 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4790 023504 001401 BEQ 30$ ;YES, CONTINUE
4791 023506 104141 ERROR 141 ;CS1 INCORRECT
4792 023510 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4793 023516 001401 BEQ 32$ ;YES, CONTINUE
4794 023520 104142 ERROR 142 ;CS2 INCORRECT
4795 023522 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4796 023530 001401 BEQ 35$ ;YES - SKIP
4797 023532 104143 ERROR 143 ;ERROR REG INCORRECT
4798 023534 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4799 023542 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4800 023550 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4801 023556 016237 000014 003154 MOV RKER(R2),T.ER ;GET ER
4802 023564 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4803 023572 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4804 023600 001401 BEQ TST40 ;GO TO NEXT TEST
4805 023602 104153 ERROR 153 ;ERROR DID NOT CLEAR
4806
4807
4808 *****
4809 *TEST 40 WRITE CHECK AND HVRC
4810 *
4811 * CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4812 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4813 * CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT
4814 * CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4815 * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
4816 * A HEADER WITH A HEADER VRC ERROR. MAKE SURE
4817 * HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
4818 * AND MAKE SURE CONTROLLER ERROR RESETS.
4819 *****
4820 †ST40: SCOPE
4821 023604 000004 MOV #10,$TIMES ;DO 10 ITERATIONS
4822 023606 012737 000012 001200 MOV $BASE,R2 ;LOAD RK611 BASE
4823 023614 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4824 023620 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4825 023626 012762 000040 000026 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
4826 023634 012762 055216 000004 MOV #-1,RKWC(R2) ;WORD COUNT=1
4827 023642 012762 177777 000002 MOV #300,RKDCYL(R2) ;LOAD CYLINDER 110MS
4828 023650 012762 000300 000020 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
4829 023656 012762 000000 000006 MOV #WATCHK,RKCS1(R2) ;ISSUE COMMAND
4830 023664 012762 000031 000000 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL

```

NO7

CZR6DBO RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 91
T40 WRITE CHECK AND HVRC

SEQ 0091

```

4831                                ; READY FOR SECTOR PULSE.
4832 023676 012762 000440 000026 1$: MOV #DMD:MCLK,RKMR1(R2)
4833 023704 012762 000040 000026 MOV #DMD,RKMR1(R2)
4834 023712 005300 DEC R0
4835 023714 001370 BNE 1$
4836 023716 012762 000140 000026 MOV #DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4837 023724 012762 000040 000026 MOV #DMD,RKMR1(R2)
4838 023732 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4839 023736 005037 003256 CLR M1.BIT
4840 023742 012700 000377 MOV #255,R0
4841 023746 004737 037026 5$: JSR PC,RDBIT
4842 023752 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4843 023754 001374 BNE 5$
4844 023756 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4845 023764 004737 037026 JSR PC,RDBIT
4846 023770 012703 053474 MOV #HEAD10,R3 ;LOAD HEADER
4847 023774 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4848 024000 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4849 024002 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4850 024006 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4851 024014 006004 ROR R4 ;GET NEXT BIT
4852 024016 103403 BCS 15$
4853 024020 005037 003254 CLR PR.BIT
4854 024024 000403 BR 16$
4855
4856 024026 012737 000001 003254 15$: MOV #1,PR.BIT
4857 024034 004737 037026 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4858 024040 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4859 024042 001361 BNE 12$ ;NO, CONTINUE
4860 024044 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4861 024046 001354 BNE 10$ ;NO, CONTINUE
4862 024050 012700 000101 MOV #65,R0
4863 024054 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4864 024062 005037 003254 CLR PR.BIT
4865 024066 004737 037026 JSR PC,RDBIT
4866 024072 005300 DEC R0 ;CHECK IF GAP FINISHED
4867 024074 001367 BNE 20$ ;NO, CONTINUE
4868 024076 012700 021200 MOV #2*(256.+(16.*256.)/64.),R0 ;LOAD COUNT UNTIL POSTAMELE
4869 024102 012762 000440 000026 25$: MOV #DMD:MCLK,RKMR1(R2)
4870 024110 012762 000040 000026 MOV #DMD,RKMR1(R2)
4871 024116 005300 DEC R0
4872 024120 001370 BNE 25$
4873 024122 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4874 024130 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4875 024136 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4876 024144 012737 100230 003200 MOV #CERR:RDY!WATCHK<C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4877 024152 012737 000100 003210 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4878 024160 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4879 024166 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4880 024174 001401 BEQ 30$ ;YES, CONTINUE
4881 024176 104141 ERROR 141 ;CS1 INCORRECT
4882 024200 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4883 024206 001401 BEQ 32$ ;YES, CONTINUE
4884 024210 104142 ERROR 142 ;CS2 INCORRECT
4885 024212 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4886 024220 001401 BEQ 35$ ;YES - SKIP

```

```

4887 024222 104143          ERROR 143          ;ERROR REG INCORRECT
4888 024224 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4889 024232 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4890 024240 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;CS2
4891 024246 016237 000014 003154 MOV RKER(R2),T.ER ;ER
4892 024254 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4893 024262 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4894 024270 001401 BEQ TST41 ;GO TO NEXT TEST
4895 024272 104153 ERROR 153 ;ERROR DID NOT CLEAR

```

.SBTTL **ECC GENERATION TESTS

```

*****
;TEST 41 ECC INITIALIZATION
;
; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
; PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
; DATA OF 10 WORDS TO AN RK06, IN 26 SECTOR
; FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
; SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
; A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY FOUR
; DATA WORDS OF ZEROES. MAKE SURE THE ECC PATTERN
; REGISTER REMAINS ZERO.
*****

```

```

4913 024274 000004          TST41: SCOPE
4914 024276 012737 000012 001200 MOV #10,STIMES ;DO 10. ITERATIONS
4915 024304 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4916 024310 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4917 024316 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4918 024324 012762 053510 000004 MOV #ECC1,RKBA(R2) ;LOAD ADDRESS OF ECC DATA
4919 024332 012762 177770 000002 MOV #-10,RKWC(R2) ;WORD COUNT =10
4920 024340 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
4921 024346 012700 000450 MOV #8.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE
4922
4923 024352 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4924 024360 012762 000040 000026 MOV #DMD,RKMR1(R2)
4925 024366 00530C DEC R0
4926 024370 001370 BNE 1$
4927 024372 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4928 024400 012762 000040 000026 MOV #DMD,RKMR1(R2)
4929 024406 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4930 024412 005037 003256 CLR M1.BIT
4931 024416 012700 000377 MOV #255.,R0
4932 024422 004737 037026 5$: JSR PC,ROBIT
4933 024426 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4934 024430 001374 BNE 5$
4935 024432 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4936 024440 004737 037026 JSR PC,ROBIT
4937 024444 012701 000003 MOV #3,R1 ;SIMULATE HEADER
4938 024450 012703 053364 MOV #HEAD1,R3 ;CYL 0, TRK 0, SECTOR 0
4939 024454 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4940 024456 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4941 024462 013737 003254 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4942 024470 006004 ROR R4 ;GET NEXT BIT

```

4943	024472	103403				BCS	17\$		
4944	024474	005037	003254			CLR	PR.BIT		
4945	024500	000403				BR	18\$		
4946									
4947	024502	012737	000001	003254	17\$:	MOV	#1,PR.BIT		
4948	024510	004737	037026		18\$:	JSR	PC,RDBIT		;SIMULATE NEXT BIT
4949	024514	005300				DEC	R0		;CHECK IF READY FOR NEXT HEADER WORD
4950	024516	001361				BNE	15\$		;NO, CONTINUE
4951	024520	005301				DEC	R1		;CHECK IF FINISHED WITH HEADER
4952	024522	001354				BNE	12\$		;NO, CONTINUE
4953	024524	012700	000101			MOV	#65,R0		;SIMULATE GAP +1 FOR SWITCH FROM READ TO WRITE
4954	024530	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT		
4955	024536	005037	003254			CLR	PR.BIT		
4956	024542	004737	037026			JSR	PC,RDBIT		
4957	024546	005300				DEC	R0		;CHECK IF GAP FINISHED
4958	024550	001367				BNE	20\$		;NO, CONTINUE
4959	024552	012700	000400			MOV	#256,R0		;LOAD COUNT FOR SYNCH FIELD
4960	024556	012737	050670	001310		MOV	#EM327,EMW		;LOAD ERROR MESSAGE
4961	024564	005037	003252			CLR	P1.BIT		;INITIALIZE BITS
4962	024570	005037	003254			CLR	PR.BIT		
4963	024574	005037	003256			CLR	M1.BIT		
4964	024600	005037	003260			CLR	M2.BIT		
4965	024604	005037	003262			CLR	BITCNT		;INITIALIZE BIT COUNT
4966	024610	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WRTGAT E.MR1		;LOAD EXPECTED MR1
4967	024616	004737	036300		25\$:	JSR	PC,WRTBIT		;WRITE BIT
4968	024622	104002				ERROR	2		;DATA INCORRECT
4969	024624	005237	003262			INC	BITCNT		;INCREMENT BIT COUNT
4970	024630	005300				DEC	R0		;CHECK IF FINISHED
4971	024632	001371				BNE	25\$		;NO, CONTINUE
4972	024634	012737	000001	003252		MOV	#1,P1.BIT		;SIMULATE SYNCH BIT
4973	024642	004737	036300			JSR	PC,WRTBIT		
4974	024646	104002				ERROR	2		
4975	024650	005037	003262			CLR	BITCNT		;CLEAR BIT COUNT
4976	024654	012737	050742	001310		MOV	#EM328,EMW		;LOAD ERROR MESSAGE
4977	024662	005037	003234			CLR	E.ECPT		;CLEAR EXPECTED ECC PATTERN
4978	024666	012703	053510			MOV	#ECC1,R3		;LOAD START OF DATA
4979	024672	012701	000004			MOV	#4,R1		;LOAD COUNT
4980	024676	012304			30\$:	MOV	(R3)+,R4		;GET NEXT WORD
4981	024700	012700	000020			MOV	#16,R0		;LOAD BITS PER WORD
4982	024704	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT		;SHIFT BITS
4983	024712	013737	003254	003256		MOV	PR.BIT,M1.BIT		
4984	024720	013737	003252	003254		MOV	P1.BIT,PR.BIT		
4985	024726	006004				ROR	R4		;GET NEXT BIT
4986	024730	103403				BCS	34\$		
4987	024732	005037	003252			CLR	P1.BIT		
4988	024736	000403				BR	35\$		
4989									
4990	024740	012737	000001	003252	34\$:	MOV	#1,P1.BIT		
4991	024746	004737	036300		35\$:	JSR	PC,WRTBIT		;SIMULATE BIT
4992	024752	104002				ERROR	2		
4993	024754	016237	000032	003174		MOV	RKECPT(R2),T.ECPT		;STORE ECC WORD
4994	024762	023737	003234	003174		CMP	E.ECPT,T.ECPT		;CHECK ECC PATTERN CORRECT
4995	024770	001401				BEQ	37\$		;YES, CONTINUE
4996	024772	104134				ERROR	134		;ECC PATTERN NOT ZERO
4997	024774	005237	003262		37\$:	INC	BITCNT		;INCREMENT BIT COUNT
4998	025000	005300				DEC	R0		;CHECK IF FINISHED WITH WORD


```

4999 025002 001340      BNE 32$      ;NO, CONTINUE
5000 025004 005301      DEC R1      ;CHECK IF FINISHED WITH TEST
5001 025006 001333      BNE 30$      ;NO, CONTINUE

```

```

*****
TEST 42      ECC GENERATION (PART 1)

```

```

*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR
*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
*      DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
*      FORMAT CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH
*      SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR
*      PULSE AND A GOOD HEADER.  CLOCK THROUGH ONLY THE
*      FOLLOWING SIX WORDS OF DATA:

```

```

005001
040040
020004
000064
000000
000000

```

```

*      CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
*      GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

```

```

*****
T42: SCOPE

```

```

5025 025010 000004      #10, $TIMES      ;DO 10. ITERATIONS
5026 025012 012737 000012 001200      MOV $BASE, R2      ;LOAD RK611 BASE
5027 025020 013702 001270      MOV #CCLR, RKCS1(R2) ;CLEAR RK611
5028 025024 012762 100000 000000      MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5029 025032 012762 000040 000026      MOV #ECC2, RKBA(R2) ;LOAD ECC DATA
5030 025040 012762 053530 000004      MOV #-12, RKWC(R2) ;WORD COUNT=12
5031 025046 012762 177766 000002      MOV #WADATA, RKCS1(R2) ;ISSUE WRITE DATA
5032 025054 012762 000023 000000      MOV #10, *37, R0 ;ISSUE ENOUGH CLOCKS UNTIL
5033 025062 012700 000562      MOV ;READY FOR SECTOR PULSE
5034
5035 025066 012762 000440 000026 1$: MOV #DMD:MCLK, RKMR1(R2)
5036 025074 012762 000040 000026      MOV #DMD, RKMR1(R2)
5037 025102 005300      DEC R0
5038 025104 001370      BNE 1$
5039 025106 012762 000140 000026      MOV #DMD:MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
5040 025114 012762 000040 000026      MOV #DMD, RKMR1(R2)
5041 025122 005037 003254      CLR PR.BIT ;GENERATE SYNCH
5042 025126 005037 003256      CLR M1.BIT
5043 025132 012700 000377      MOV #255, R0
5044 025136 004737 037026 5$: JSR PC, R0BIT
5045 025142 005300      DEC R0 ;CHECK IF SYNCH FINISHED
5046 025144 001374      BNE 5$
5047 025146 012737 000001 003254      MOV #1, PR.BIT ;SIMULATE SYNCH BIT
5048 025154 004737 037026      JSR PC, R0BIT
5049 025160 012701 000003      MOV #3, R1 ;SIMULATE HEADER
5050 025164 012703 053364      MOV #HEAD1, R3 ;CYL 0, TRK 0, SECTOR 0
5051 025170 012304 12$: MOV (R3)+, R4 ;GET NEXT HEADER WORD
5052 025172 012700 000020      MOV #16, R0 ;LOAD BITS PER WORD
5053 025176 013737 003254 15$: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
5054 025204 006004      ROR R4 ;GET NEXT BIT

```

5055	025206	103403				BCS	17\$		
5056	025210	005037	003254			CLR	PR.BIT		
5057	025214	000403				BR	18\$		
5058									
5059	025216	012737	000001	003254	17\$:	MOV	#1,PR.BIT		
5060	025224	004737	037026		18\$:	JSR	PC,RDBIT		;SIMULATE NEXT BIT
5061	025230	005300				DEC	R0		;CHECK IF READY FOR NEXT HEADER WORD
5062	025232	001361				BNE	15\$		;NO, CONTINUE
5063	025234	005301				DEC	R1		;CHECK IF FINISHED WITH HEADER
5064	025236	001354				BNE	12\$		;NO, CONTINUE
5065	025240	012700	000101			MOV	#65,R0		;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5066	025244	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT		
5067	025252	005037	003254			CLR	PR.BIT		
5068	025256	004737	037026			JSR	PC,RDBIT		
5069	025262	005300				DEC	R0		;CHECK IF GAP FINISHED
5070	025264	001367				BNE	20\$		;NO, CONTINUE
5071	025266	012700	000400			MOV	#256,R0		;LOAD COUNT FOR SYNCH FIELD
5072	025272	012737	050670	001310		MOV	#EM327,EMW		;LOAD ERROR
5073	025300	005037	003252			CLR	P1.BIT		;INITIALIZE BITS
5074	025304	005037	003254			CLR	PR.BIT		
5075	025310	005037	003256			CLR	M1.BIT		
5076	025314	005037	003260			CLR	M2.BIT		
5077	025320	005037	003262			CLR	BITCNT		;INITIALIZE BIT COUNT
5078	025324	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WATGAT E.MR1		;SET EXPECTED MR1
5079	025332	004737	036300		25\$:	JSR	PC,WRTBIT		;WRITE BIT
5080	025336	104002				ERROR	2		;DATA INCORRECT
5081	025340	005237	003262			INC	BITCNT		;INCREMENT BIT COUNT
5082	025344	005300				DEC	R0		;CHECK IF FINISHED
5083	025346	001371				BNE	25\$		;NO, CONTINUE
5084	025350	012737	000001	003252		MOV	#1,P1.BIT		;SIMULATE SYNCH BIT
5085	025356	004737	036300			JSR	PC,WRTBIT		
5086	025362	104002				ERROR	2		
5087	025364	005037	003262			CLR	BITCNT		;CLEAR BIT COUNT
5088	025370	012737	050742	001310		MOV	#EM328,EMW		;LOAD ERROR MESSAGE
5089	025376	005037	003266			CLR	ECCHI		;INITIALIZE ECC
5090	025402	005037	003270			CLR	ECCL0		
5091	025406	012703	053530			MOV	#ECC2,R3		;LOAD START OF DATA
5092	025412	012701	000006			MOV	#6,R1		;LOAD COUNT
5093	025416	012304			30\$:	MOV	(R3)+,R4		;GET NEXT BIT
5094	025420	012700	000020			MOV	#16,R0		;LOAD BITS PER WORD
5095	025424	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT		;SHIFT BITS
5096	025432	013737	003254	003256		MOV	PR.BIT,M1.BIT		
5097	025440	013737	003252	003254		MOV	P1.BIT,PR.BIT		
5098	025446	006004				ROR	R4		;GET NEXT BIT
5099	025450	103403				BCS	34\$		
5100	025452	005037	003252			CLR	P1.BIT		
5101	025456	000403				BR	35\$		
5102									
5103	025460	012737	000001	003252	34\$:	MOV	#1,P1.BIT		
5104	025466	004737	036300		35\$:	JSR	PC,WRTBIT		;SIMULATE BIT
5105	025472	104002				ERROR	2		
5106	025474	016237	000032	003174		MOV	RKECPT(R2),T.ECPT		;STORE ECC WORD
5107	025502	004737	036132			JSR	PC,ECCGEN		;GENERATE EXPECTED ECC PAT
5108	025506	023737	003234	003174		CMP	E.ECPT,T.ECPT		;CHECK ECC PATTERN CORRECT
5109	025514	001401				BEQ	37\$		;YES, CONTINUE
5110	025516	104135				ERROR	135		;ECC PATTERN INCORRECT

5111 025520 005237 003262
5112 025524 005300
5113 025526 001336
5114 025530 005301
5115 025532 001331
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138

37\$: INC BITCNT ; INCREMENT BIT COUNT
DEC R0 ; CHECK IF FINISHED WITH WORD
BNE 32\$; NO, CONTINUE
DEC R1 ; CHECK IF FINISHED WITH TEST
BNE 30\$; NO, CONTINUE

TEST 43 ECC GENERATION (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
AND A GOOD HEADER. CLOCK THROUGH ONLY THE FOLLOWING
SIX WORDS OF DATA:

177777
177777
177777
177777
177777
177777

CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.

ST43: SCOPE

5139 025534 000004
5140 025536 012737 000012 001200
5141 025544 013702 001270
5142 025550 012762 100000 000000
5143 025556 012762 000040 000026
5144 025564 012762 053550 000004
5145 025572 012762 177766 000002
5146 025600 012762 000023 000000
5147 025606 012700 000562
5148
5149 025612 012762 000440 000026
5150 025620 012762 000040 000026
5151 025626 005300
5152 025630 001370
5153 025632 012762 000140 000026
5154 025640 012762 000040 000026
5155 025646 005037 003254
5156 025652 005037 003256
5157 025656 012700 000377
5158 025662 004737 037026
5159 025666 005300
5160 025670 001374
5161 025672 012737 000001 003254
5162 025700 004737 037026
5163 025704 012701 000003
5164 025710 012703 053364
5165 025714 012304
5166 025716 012700 000020

MOV #10, \$TIMES ; DO 10. ITERATIONS
MOV \$BASE, R2 ; LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 IN DIAGNOSTIC MODE
MOV #ECC3, RKBA(R2) ; LOAD ECC DATA
MOV #-12, RKWC(R2) ; WORD COUNT=12
MOV #WRDATA, RKCS1(R2) ; ISSUE WRITE DATA
MOV #10.*37., R0 ; ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE
1\$: MOV #DMD!MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 1\$
MOV #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
CLR PR.BIT ; GENERATE SYNCH
CLR M1.BIT
MOV #255., R0
JSR PC, RDBIT
DEC R0 ; CHECK IF SYNCH FINISHED
BNE 5\$
MOV #1, PR.BIT ; SIMULATE SYNCH BIT
JSR PC, RDBIT
MOV #3, R1 ; SIMULATE HEADER
MOV #HEAD1, R3 ; CYL 0, TRK 0, SECTOR 0
12\$: MOV (R3)+, R4 ; GET NEXT HEADER WORD
MOV #16., R0 ; LOAD BITS PER WORD

5167	025722	013737	003254	003256	15\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
5168	025730	006004				ROR	R4	:GET NEXT BIT
5169	025732	103403				BCS	17\$	
5170	025734	005037	003254			CLR	PR.BIT	
5171	025740	000403				BR	18\$	
5172								
5173	025742	012737	000001	003254	17\$:	MOV	#1,FR.BIT	
5174	025750	004737	037026		18\$:	JSR	PC,RDBIT	:SIMULATE NEXT BIT
5175	025754	005300				DEC	R0	:CHECK IF READY FOR NEXT HEADER WORD
5176	025756	001361				BNE	15\$	:NO, CONTINUE
5177	025760	005301				DEC	R1	:CHECK IF FINISHED WITH HEADER
5178	025762	001354				BNE	12\$	:NO, CONTINUE
5179	025764	012700	000101			MOV	#65,R0	:SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5180	025770	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT	
5181	025776	005037	003254			CLR	PR.BIT	
5182	026002	004737	037026			JSR	PC,RDBIT	
5183	026006	005300				DEC	R0	:CHECK IF GAP FINISHED
5184	026010	001367				BNE	20\$	:NO, CONTINUE
5185	026012	012700	000400			MOV	#256,R0	:LOAD COUNT FOR SYNCH FIELD
5186	026016	012737	050670	001310		MOV	#EM327,EMW	:LOAD ERROR
5187	026024	005037	003252			CLR	P1.BIT	:INITIALIZE BITS
5188	026030	005037	003254			CLR	PR.BIT	
5189	026034	005037	003256			CLR	M1.BIT	
5190	026040	005037	003260			CLR	M2.BIT	
5191	026044	005037	003262			CLR	BITCNT	:INITIALIZE BIT COUNT
5192	026050	012737	062040	003224	25\$:	MOV	#0MD!ECCW!MEWD!WRTGAT,E.MR1	:SET EXPECTED MR1
5193	026056	004737	036300			JSR	PC,WRTBIT	:WRITE BIT
5194	026062	104002				ERROR	2	:DATA INCORRECT
5195	026064	005237	003262			INC	BITCNT	:INCREMENT BIT COUNT
5196	026070	005300				DEC	R0	:CHECK IF FINISHED
5197	026072	001371				BNE	25\$	:NO, CONTINUE
5198	026074	012737	000001	003252		MOV	#1,P1.BIT	:SIMULATE SYNCH BIT
5199	026102	004737	036300			JSR	PC,WRTBIT	
5200	026106	104002				ERROR	2	
5201	026110	005037	003262			CLR	BITCNT	:CLEAR BIT COUNT
5202	026114	012737	050742	001310		MOV	#EM328,EMW	:LOAD ERROR MESSAGE
5203	026122	005037	003256			CLR	ECCHI	:INITIALIZE ECC
5204	026126	005037	003270			CLR	ECCLO	
5205	026132	012703	053550			MOV	#ECC3,R3	:LOAD START OF DATA
5206	026136	012701	000006			MOV	#6,R1	:LOAD COUNT
5207	026142	012304			30\$:	MOV	(R3)+,R4	:GET NEXT BIT
5208	026144	012700	000020			MOV	#16,R0	:LOAD BITS PER WORD
5209	026150	013737	003256	003260	32\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
5210	026156	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5211	026164	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5212	026172	006004				ROR	R4	:GET NEXT BIT
5213	026174	103403				BCS	34\$	
5214	026176	005037	003252			CLR	P1.BIT	
5215	026202	000403				BR	35\$	
5216								
5217	026204	012737	000001	003252	34\$:	MOV	#1,P1.BIT	
5218	026212	004737	036300		35\$:	JSR	PC,WRTBIT	:SIMULATE BIT
5219	026216	104002				ERROR	2	
5220	026220	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	:STORE ECC WORD
5221	026226	004737	036132			JSR	PC,ECCGEN	:GENERATE EXPECTED ECC PAT
5222	026232	023737	003234	003174		CMP	E.ECPT,T.ECPT	:CHECK ECC PATTERN CORRECT

5223	026240	001401			BEQ	37\$		;YES, CONTINUE
5224	026242	104135			ERROR	135		;ECC PATTERN INCORRECT
5225	026244	005237	003262		37\$: INC	BITCNT		;INCREMENT BIT COUNT
5226	026250	005300			DEC	R0		;CHECK IF FINISHED WITH WORD
5227	026252	001336			BNE	32\$		;NO, CONTINUE
5228	026254	005301			DEC	R1		;CHECK IF FINISHED WITH TEST
5229	026256	001331			BNE	30\$		;NO, CONTINUE
5230								
5231								
5232								
5233								
5234								
5235								
5236								
5237								
5238								
5239								
5240								
5241								
5242								
5243								
5244								
5245	026260	000004						
5246	026262	012737	000012	001200	ST44: SCOPE			
5247					MOV	#10., \$TIMES		;DO 10. ITERATIONS
5248	026270	013702	001270		MOV	\$BASE, R2		;LOAD RK611 BASE
5249	026274	012762	100000	000000	MOV	#CCLR, RKCS1(R2)		;CLEAR RK611
5250	026302	012762	000040	000026	MOV	#DMD, RKMR1(R2)		;PUT RK611 IN DIAGNOSTIC MODE
5251	026310	012762	056216	000004	MOV	#ECCBUF, RKBA(R2)		;LOAD ADDRESS
5252	026316	012762	177400	000002	MOV	#-400, RKWC(R2)		;WORD COUNT=400
5253	026324	012762	000023	000000	MOV	#WRDATA, RKCS1(R2)		;ISSUE WRITE DATA
5254	026332	012700	004612		MOV	#66.*37., R0		;ISSUE ENOUGH CLOCKS UNTIL READY FOR SECTOR PULSE
5255								
5256	026336	012762	000440	000026	1\$: MOV	#DMD!MCLK, RKMR1(R2)		
5257	026344	012762	000040	000026	MOV	#DMD, RKMR1(R2)		
5258	026352	005300			DEC	R0		
5259	026354	001370			BNE	1\$		
5260	026356	012762	000140	000026	MOV	#DMD!MSP, RKMR1(R2)		;SIMULATE SECTOR PULSE
5261	026364	012762	000040	000026	MOV	#DMD, RKMR1(R2)		
5262	026372	005037	003254		CLR	PR.BIT		
5263	026376	005037	003256		CLR	M1.BIT		
5264	026402	012700	000377		MOV	#255., R0		
5265	026406	004737	037026		5\$: JSR	PC, R0BIT		
5266	026412	005300			DEC	R0		;CHECK IF SYNCH FINISHED
5267	026414	001374			BNE	5\$		
5268	026416	012737	000031	003254	MOV	#1, PR.BIT		;SIMULATE SYNCH
5269	026424	004737	037026		JSR	PC, R0BIT		
5270	026430	012703	053064		MOV	#HEAD1, R3		;LOAD HEADER
5271	026434	012701	000003		MOV	#3, R1		;LOAD WORDS PER HEADER
5272	026440	012304			10\$: MOV	(R3)+, R4		;GET NEXT WORD
5273	026442	012700	000020		MOV	#16., R0		;LOAD BITS PER
5274	026446	013737	003254	003256	12\$: MOV	PR.BIT, M1.BIT		;STORE PREVIOUS BIT
5275	026454	006004			ROR	R4		;GET NEXT BIT
5276	026456	103403			BCS	15\$		
5277	026460	005037	003254		CLR	PR.BIT		
5278	026464	000403			BR	16\$		

```
5279
5280 026466 012737 000001 003254 15$: MOV #1,PR.BIT
5281 026474 004737 037026 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5282 026500 005300 DEC RO ;CHECK IF READY FOR NEXT WORD
5283 026502 001361 BNE 12$ ;NO, CONTINUE
5284 026504 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5285 026506 001354 BNE 10$ ;NO, CONTINUE
5286 026510 012700 000101 MOV #65,RO
5287 026514 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5288 026522 005037 003254 CLR PR.BIT
5289 026526 004737 037026 JSR PC,RDBIT
5290 026532 005300 DEC RO ;CHECK IF GAP FINISHED
5291 026534 001367 BNE 20$ ;NO, CONTINUE
5292 026536 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH
5293 026542 005037 003254 CLR PR.BIT
5294 026546 005037 003256 CLR M1.BIT
5295 026552 005037 003260 CLR M2.BIT
5296 026556 012737 050670 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5297 026564 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5298 026570 012700 000400 MOV #256,RO ;LOAD SYNCH COUNT AND WRITE SYNCH
5299 026574 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5300 026602 004737 036300 25$: JSR PC,WRTBIT
5301 026606 104002 ERROR 2
5302 026610 005237 003262 INC BITCNT ;CLEAR BIT COUNT
5303 026614 005300 DEC RO ;CHECK IF SYNCH FINISHED
5304 026616 001371 BNE 25$ ;NO, CONTINUE
5305 026620 012737 000001 003252 MOV #1,P1.BIT ;WRITE SYNCH BIT
5306 026626 004737 036300 JSR PC,WRTBIT
5307 026632 104002 ERROR 2
5308 026634 005037 003266 CLR ECCHI ;INITIALIZE ECC GENERATOR
5309 026640 005037 003270 CLR ECCLO
5310 026644 012737 050742 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5311 026652 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5312 026656 012701 000400 MOV #256,R1 ;LOAD WORDS PER SECTOR
5313 026662 012703 056216 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5314 026666 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5315 026670 012700 000020 MOV #16,RO ;LOAD BITS PER WORD
5316 026674 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5317 026702 013737 003254 003256 MOV PR.BIT,M1.BIT
5318 026710 013737 003252 003254 MOV P1.BIT,PR.BIT
5319 026716 006004 ROR R4 ;DETERMINE NEXT BIT
5320 026720 103403 BCS 34$
5321 026722 005037 003252 CLR P1.BIT
5322 026726 000403 BR 35$
5323
5324 026730 012737 000001 003252 34$: MOV #1,P1.BIT
5325 026736 004737 036300 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5326 026742 104002 ERROR 2
5327 026744 004737 036132 JSR PC,ECCGEN ;GENERATE ECC
5328 026750 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET PATTERN
5329 026756 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5330 026764 001401 BEQ 37$ ;YES, CONTINUE
5331 026766 104135 ERROR 135 ;ECC PATTERN INCORRECT
5332 026770 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5333 026774 022700 000002 CMP #2,RO ;IF THE NEXT BIT IS THE LAST BIT OF
5334 027000 001006 BNE 28$ ;THE LAST DATA WORD, ECCW MUST BE RESET
```

5335	027002	022701	000001			CMP	#1,R1	;IN THE EXPECTED MRI TO INDICATE ECC
5336	027006	001003				BNE	28\$	;BEING WRITTEN
5337	027010	042737	020000	003224		B	#ECCW,E.MR1	
5338	027016	005300			28\$:	[	R0	;CHECK IF THROUGH WITH WORD
5339	027020	001325				BNE	32\$	;NO, CONTINUE
5340	027022	005301				DEC	R1	;CHECK IF AT END OF SECTOR
5341	027024	001320				BNE	30\$	;NO, CONTINUE
5342	027026	012737	051307	001310		MOV	#EM333,EMW	;LOAD ERROR MESSAGE
5343	027034	005037	003262			CLR	BITCNT	;INITIALIZE BIT COUNT
5344	027040	012701	000002			MOV	#2,R1	;LOAD NUMBER OF ECC WORDS
5345	027044	012703	003266			MOV	#ECCHI,R3	;LOAD ADDRESS OF ECC
5346	027050	012304			40\$:	MOV	(R3)+,R4	;GET NEXT ECC WORD
5347	027052	012700	000020			MOV	#16,R0	;LOAD BITS PER WORD
5348	027056	013737	003256	003260	42\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5349	027064	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5350	027072	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5351	027100	006004				ROR	R4	;DETERMINE NEXT BIT
5352	027102	103403				BCS	44\$	
5353	027104	005037	003252			CLR	P1.BIT	
5354	027110	000403				BR	45\$	
5355								
5356	027112	012737	000001	003252	44\$:	MOV	#1,P1.BIT	
5357	027120	004737	036300		45\$:	JSR	PC,WRTBIT	;WRITE NEXT BIT
5358	027124	104002				ERROR	2	
5359	027126	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
5360	027132	022700	000002			CMP	#2,R0	;IF THE LAST BIT OF THE LAST ECC WORD
5361	027136	001006				BNE	46\$	;IS BEING WRITTEN, ECCW MUST BE SET
5362	027140	022701	000001			CMP	#1,R1	;IN EXPECTED MRI TO INDICATE ECC
5363	027144	001003				BNE	46\$	;WRITING IS DONE
5364	027146	052737	020000	003224		BIS	#ECCW,E.MR1	
5365	027154	005300			46\$:	DEC	R0	;CHECK IF THROUGH WITH WORD
5366	027156	001337				BNE	42\$	;NO, CONTINUE
5367	027160	005301				DEC	R1	;CHECK IF FINISH WITH ECC
5368	027162	001332				BNE	40\$	;NO, CONTINUE
5369	027164	012737	051345	001310		MOV	#EM334,EMW	;LOAD ERROR MESSAGE
5370	027172	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
5371	027176	012700	000017			MOV	#15,R0	;LOAD POSTAMBLE BIT COUNT
5372	027202	013737	003256	003260	47\$:	MOV	M1.BIT,M2.BIT	;SHIFT BIT
5373	027210	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5374	027216	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5375	027224	005037	003252			CLR	P1.BIT	
5376	027230	004737	036300			JSR	PC,WRTBIT	;WRITE NEXT BIT
5377	027234	104002				ERROR	2	
5378	027236	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
5379	027242	005300				DEC	R0	;CHECK IF THROUGH WITH POSTAMBLE
5380	027244	001356				BNE	47\$	;NO, CONTINUE
5381	027246	012700	000014			MOV	#3*4,R0	;ISSUE CLOCKS TO COMPLETE COMMAND
5382	027252	012762	000440	000026	50\$:	MOV	#DMD!MCLK,RKMR1(R2)	;ISSUE CLOCK PULSES
5383	027260	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5384	027266	005300				DEC	R0	
5385	027270	001370				BNE	50\$	
5386	027272	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;SAVE CS1
5387	027300	016237	000010	003150		MOV	RKCS2(R2),T.CS2	;SAVE CS2
5388	027306	016237	000014	003154		MOV	RKER(R2),T.ER	;SAVE ERROR REG
5389	027314	012737	000222	003200		MOV	#RDY!WRDATA<T.CS1>,E.CS1	;LOAD EXPECTED CS1
5390	027322	012737	000100	003210		MOV	#IR,E.CS2	;LOAD EXPECTED CS2

5391	027330	005037	003214		CLR	E.ER	;LOAD EXPECTED ERROR REG.
5392	027334	016237	000020	003160	MOV	RKDCYL(R2),T.DCYL	;SAVE CYLINDER ADDR REG
5393	027342	016237	000006	003146	MOV	RKDA(R2),T.DA	;SAVE DISK ADDR REG
5394	027350	016237	000004	003144	MOV	RKBA(R2),T.BA	;SAVE BUS ADDR REG
5395	027356	016237	000002	003142	MOV	RKWC(R2),T.WC	;SAVE WORD COUNT
5396	027364	005037	003220		CLR	E.DCYL	;LOAD EXPECTED CYLINDER AND REG.
5397	027370	012737	000001	003206	MOV	#1,E.DA	;LOAD EXPECTED DISK
5398	027376	012737	057216	003204	MOV	#ECCBUF+(<400*2),E.BA	;LOAD EXPECTED
5399	027404	005037	003202		CLR	E.WC	;LOAD EXPECTED ECC WORD
5400	027410	023737	003200	003140	CMP	E.CS1,T.CS1	;CHECK CS1
5401	027416	001401			BEQ	55\$	
5402	027420	104144			ERROR	144	
5403	027422	023737	003210	003150	55\$: CMP	E.CS2,T.CS2	;CHECK CS2
5404	027430	001401			BEQ	56\$	
5405	027432	104145			ERROR	145	
5406	027434	023737	003214	003154	56\$: CMP	E.ER,T.ER	;CHECK ERROR REG
5407	027442	001401			BEQ	57\$	
5408	027444	104146			ERROR	146	
5409	027446	023737	003204	003144	57\$: CMP	E.BA,T.BA	;CHECK BUS ADDR
5410	027454	001401			BEQ	58\$	
5411	027456	104147			ERROR	147	
5412	027460	023737	003202	003142	58\$: CMP	E.WC,T.WC	;CHECK WORD COUNT
5413	027466	001401			BEQ	59\$	
5414	027470	104150			ERROR	150	
5415	027472	023737	003220	003160	59\$: CMP	E.DCYL,T.DCYL	;CHECK CYLINDER ADDR
5416	027500	001401			BEQ	60\$	
5417	027502	104151			ERROR	151	
5418	027504	023737	003206	003146	60\$: CMP	E.DA,T.DA	;CHECK DISK ADDR.
5419	027512	001401			BEQ	TST45	;YES, GO ON TO NEXT TEST
5420	027514	104152			ERROR	152	
5421							
5422							
5423							
5424							
5425							
5426							
5427							
5428							
5429							
5430							
5431							
5432							
5433							
5434							
5435							
5436							
5437							
5438	027516	000004			TST45: SCOPE		
5439	027520	012737	000012	001200	MOV	#10.,\$TIMES	;DO 10. ITERATIONS
5440							
5441	027526	013702	001270		MOV	\$BASE,R2	;LOAD RK611 BASE
5442	027532	012762	100000	000000	MOV	#CLR,RKCS1(R2)	;CLEAR RK611
5443	027540	012762	000040	000026	MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
5444	027546	012762	056216	000004	MOV	#ECCBUF,RKBA(R2)	;LOAD ADDRESS
5445	027554	012762	177675	000002	MOV	#-103,RKWC(R2)	;WORD COUNT=400
5446	027562	012762	000023	000000	MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA

.SBTTL **PARTIAL WRITE DATA

```

*****
*TEST 45      ZERO FILL ON WRITE DATA
*
*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
*      DATA OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
*      AND A GOOD HEADER.  CLOCK THROUGH ALL 400 WORDS AND
*      THE TWO ECC WORDS.  CHECK THE SECTOR FOR ZERO FILL
*      AND MAKE SURE THE TWO ECC WORDS ARE WRITTEN PROPERLY.
*      CHECK BUS ADDRESS, WORD COUNT, CYLINDER, TRACK, AND SECTOR.
*****

```

```

*****
TST45: SCOPE
MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
MOV      $BASE,R2         ;LOAD RK611 BASE
MOV      #CLR,RKCS1(R2)   ;CLEAR RK611
MOV      #DMD,RKMR1(R2)   ;PUT RK611 IN DIAGNOSTIC MODE
MOV      #ECCBUF,RKBA(R2) ;LOAD ADDRESS
MOV      #-103,RKWC(R2)   ;WORD COUNT=400
MOV      #WRDATA,RKCS1(R2);ISSUE WRITE DATA

```



```
5447 027570 012700 004612      MOV      #66.*37.,R0      ;ISSUE ENOUGH CLOCKS UNTIL
5448                                     ;READY FOR SECTOR PULSE
5449 027574 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
5450 027602 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5451 027610 005300      DEC      R0
5452 027612 001370      BNE      1$
5453 027614 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5454 027622 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5455 027630 005037 003254      CLR      PR.BIT
5456 027634 005037 003256      CLR      M1.BIT
5457 027640 012700 000377      MOV      #255.,R0
5458 027644 004737 037026      JSR      PC,RDBIT
5459 027650 005300      DEC      R0      ;CHECK IF SYNCH FINISHED
5460 027652 001374      BNE      5$
5461 027654 012737 000001 003254      MOV      #1,PR.BIT      ;SIMULATE SYNCH
5462 027662 004737 037026      JSR      PC,RDBIT
5463 027666 012703 053364      MOV      #HEAD1,R3      ;LOAD HEADER
5464 027672 012701 000003      MOV      #3,R1      ;LOAD WORDS PER HEADER
5465 027676 012304      MOV      (R3)+,R4      ;GET NEXT WORD
5466 027700 012700 000020      MOV      #16.,R0      ;LOAD BITS PER
5467 027704 013737 003254 003256 12$: MOV      PR.BIT,M1.BIT      ;STORE PREVIOUS BIT
5468 027712 006004      ROR      R4      ;GET NEXT BIT
5469 027714 103403      BCS      15$
5470 027716 005037 003254      CLR      PR.BIT
5471 027722 000403      BR       16$
5472
5473 027724 012737 000001 003254 15$: MOV      #1,PR.BIT
5474 027732 004737 037026 16$:  JSR      PC,RDBIT      ;SIMULATE NEXT BIT
5475 027736 005300      DEC      R0      ;CHECK IF READY FOR NEXT WORD
5476 027740 001361      BNE      12$      ;NO, CONTINUE
5477 027742 005301      DEC      R1      ;CHECK IF FINISHED WITH HEADER
5478 027744 001354      BNE      10$      ;NO, CONTINUE
5479 027746 012700 000101      MOV      #65.,R0
5480 027752 013737 003254 003256 20$: MOV      PR.BIT,M1.BIT      ;SIMULATE GAP
5481 027760 005037 003254      CLR      PR.BIT
5482 027764 004737 037026      JSR      PC,RDBIT
5483 027770 005300      DEC      R0      ;CHECK IF GAP FINISHED
5484 027772 001367      BNE      20$      ;NO, CONTINUE
5485 027774 005037 003252      CLR      P1.BIT      ;INITIALIZE BITS FOR SYNCH
5486 030000 005037 003254      CLR      PR.BIT
5487 030004 005037 003256      CLR      M1.BIT
5488 030010 005037 003260      CLR      M2.BIT
5489 030014 012737 050670 001310      MOV      #EM327,EMW      ;LOAD ERROR MESSAGE
5490 030022 005037 003262      CLR      BITCNT      ;INITIALIZE BIT COUNT
5491 030026 012700 000400      MOV      #256.,R0      ;LOAD SYNCH COUNT AND WRITE SYNCH
5492 030032 012737 062040 003224      MOV      #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5493 030040 004737 036300      JSR      PC,WRTBIT
5494 030044 104002      ERROR      2
5495 030046 005237 003262      INC      BITCNT      ;CLEAR BIT COUNT
5496 030052 005300      DEC      R0      ;CHECK IF SYNCH FINISHED
5497 030054 001371      BNE      25$      ;NO, CONTINUE
5498 030056 012737 000001 003252      MOV      #1,P1.BIT      ;WRITE SYNCH BIT
5499 030064 004737 036300      JSR      PC,WRTBIT
5500 030070 104002      ERROR      2
5501 030072 005037 003266      CLR      ECCHI      ;INITIALIZE ECC GENERATOR
5502 030076 005037 003270      CLR      ECCLO
```

```

5503 030102 012737 050742 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5504 030110 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5505 030114 012701 000400 MOV #256,R1 ;LOAD WORDS PER SECTOR
5506 030120 012703 056216 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5507 030124 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5508 030126 012700 000020 31$: MOV #16,R0 ;LOAD BITS PER WORD
5509 030132 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5510 030140 013737 003254 003256 MOV PR.BIT,M1.BIT
5511 030146 013737 003252 003254 MOV P1.BIT,PR.BIT
5512 030154 006004 ROR R4 ;DETERMINE NEXT BIT
5513 030156 103403 BCS 34$
5514 030160 005037 003252 CLR P1.BIT
5515 030164 000403 BR 35$
5516
5517 030166 012737 000001 003252 34$: MOV #1,P1.BIT
5518 030174 004737 036300 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5519 030200 104002 ERROR 2
5520 030202 004737 036132 JSR PC,ECCGEN ;GENERATE ECC
5521 030206 016237 000032 003174 MOV R1,ECCPT(R2),T.ECPT ;GET PATTERN
5522 030214 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5523 030222 001401 BEQ 37$ ;YES, CONTINUE
5524 030224 104135 ERROR 135 ;ECC PATTERN INCORRECT
5525 030226 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5526 030232 022700 000002 CMP #2,R0 ;IF THE NEXT BIT IS THE LAST BIT OF
5527 030236 001006 BNE 28$ ;THE LAST DATA WORD, ECCW MUST BE RESET
5528 030240 022701 000001 CMP #1,R1 ;IN THE EXPECTED MR1 TO INDICATE ECC
5529 030244 001003 BNE 28$ ;BEING WRITTEN
5530 030246 042737 020000 003224 BIC #ECCW,E.MR1
5531 030254 005300 28$: DEC R0 ;CHECK IF THROUGH WITH WORD
5532 030256 001325 BNE 32$ ;NO, CONTINUE
5533 030260 005301 DEC R1 ;CHECK IF AT END OF SECTOR
5534 030262 001405 BEQ 39$ ;YES -SKIP
5535 030264 022703 056424 CMP #ECCBUF+(<103*2>),R3 ;CHECK IF 103 WORDS TRANSFERED
5536 030270 003315 BGT 30$ ;NO - GO TO GET NEXT WORD
5537 030272 005004 CLR R4 ;ELSE CLEAR R4 FOR ZEROS
5538 030274 000714 BR 31$ ;GO SIMULATE REST OF SECTOR
5539 030276 012737 051307 001310 39$: MOV #EM333,EMW ;LOAD ERROR MESSAGE
5540 030304 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5541 030310 012701 000002 MOV #2,R1 ;LOAD NUMBER OF ECC WORDS
5542 030314 012703 003266 MOV #ECCI,R3 ;LOAD ADDRESS OF ECC
5543 030320 012304 40$: MOV (R3)+,R4 ;GET NEXT ECC WORD
5544 030322 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
5545 030326 013737 003256 003260 42$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5546 030334 013737 003254 003256 MOV PR.BIT,M1.BIT
5547 030342 013737 003252 003254 MOV P1.BIT,PR.BIT
5548 030350 006004 ROR R4 ;DETERMINE NEXT BIT
5549 030352 103403 BCS 44$
5550 030354 005037 003252 CLR P1.BIT
5551 030360 000403 BR 45$
5552
5553 030362 012737 000001 003252 44$: MOV #1,P1.BIT
5554 030370 004737 036300 45$: JSR PC,WRTBIT ;WRITE NEXT BIT
5555 030374 104002 ERROR 2
5556 030376 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5557 030402 022700 000002 CMP #2,R0 ;IF THE LAST BIT OF THE LAST ECC WORD
5558 030406 001006 BNE 46$ ;IS BEING WRITTEN, ECCW MUST BE SET

```


5559	030410	022701	000001			CMP	#1,R1	; IN EXPECTED MR1 TO INDICATE ECC
5560	030414	001003				BNE	46\$	; WRITING IS DONE
5561	030416	052737	020000	003224		BIS	#ECCW,E.MR1	
5562	030424	005300			46\$:	DEC	R0	; CHECK IF THROUGH WITH WORD
5563	030426	001337				BNE	42\$	; NO CONTINUE
5564	030430	005301				DEC	R1	; CHECK IF FINISH WITH ECC
5565	030432	001332				BNE	40\$	; NO CONTINUE
5566	030434	012737	051345	001310		MOV	#EM334,EMW	; LOAD ERROR MESSAGE
5567	030442	005037	003262			CLR	BITCNT	; CLEAR BIT COUNT
5568	030446	012700	000017			MOV	#15,R0	; LOAD POSTAMBLE BIT COUNT
5569	030452	013737	003256	003260	47\$:	MOV	M1.BIT,M2.BIT	; SHIFT BIT
5570	030460	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5571	030466	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5572	030474	005037	003252			CLR	P1.BIT	
5573	030500	004737	036300			JSR	PC,WRTBIT	; WRITE NEXT BIT
5574	030504	104002				ERROR	2	
5575	030506	005237	003262			INC	BITCNT	; INCREMENT BIT COUNT
5576	030512	005300				DEC	R0	; CHECK IF THROUGH WITH POSTAMBLE
5577	030514	001356				BNE	47\$	; NO CONTINUE
5578	030516	012700	000014			MOV	#3*4,R0	; ISSUE CLOCKS TO COMPLETE COMMAND
5579	030522	012762	000440	000026	50\$:	MOV	#DMD!MCLK,RKMR1(R2)	; ISSUE CLOCK PULSES
5580	030530	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5581	030536	005300				DEC	R0	
5582	030540	001370				BNE	50\$	
5583	030542	016237	000000	003140		MOV	RKCS1(R2),T.CS1	; SAVE CS1
5584	030550	016237	000010	003150		MOV	RKCS2(R2),T.CS2	; SAVE CS2
5585	030556	016237	000014	003154		MOV	RKER(R2),T.ER	; SAVE ERROR REG
5586	030564	012737	000222	003200		MOV	#RDY!WRDATA&<T.C<GO>>,E.CS1	; LOAD EXPECTED CS1
5587	030572	012737	000100	003210		MOV	#IR,E.CS2	; LOAD EXPECTED CS2
5588	030600	005037	003214			CLR	E.ER	; LOAD EXPECTED ERROR REG.
5589	030604	016237	000020	003160		MOV	RKDCYL(R2),T.DCYL	; SAVE CYLINDER ADDR REG
5590	030612	016237	000006	003146		MOV	RKDA(R2),T.DA	; SAVE DISK AND REG
5591	030620	016237	000004	003144		MOV	RKBA(R2),T.BA	; SAVE BUS ADDR REG
5592	030626	016237	000002	003142		MOV	RKWC(R2),T.WC	; SAVE WORD COUNT
5593	030634	005037	003220			CLR	E.DCYL	; LOAD EXPECTED CYLINDER AND REG.
5594	030640	012737	000001	003206		MOV	#1,E.DA	; LOAD EXPECTED DISK
5595	030646	012737	056424	003204		MOV	#ECCBUF+<103*2>,E.BA	; LOAD EXPECTED
5596	030654	005037	003202			CLR	E.WC	; LOAD EXPECTED ECC WORD
5597	030660	023737	003200	003140		CMP	E.CS1,T.CS1	; CHECK CS1
5598	030666	001401				BEQ	55\$	
5599	030670	104154				ERROR	154	
5600	030672	023737	003210	003150	55\$:	CMP	E.CS2,T.CS2	; CHECK CS2
5601	030700	001401				BEQ	56\$	
5602	030702	104155				ERROR	155	
5603	030704	023737	003214	003154	56\$:	CMP	E.ER,T.ER	; CHECK ERROR REG
5604	030712	001401				BEQ	57\$	
5605	030714	104156				ERROR	156	
5606	030716	023737	003204	003144	57\$:	CMP	E.BA,T.BA	; CHECK BUS ADD
5607	030724	001401				BEQ	58\$	
5608	030726	104157				ERROR	157	
5609	030730	023737	003202	003142	58\$:	CMP	E.WC,T.WC	; CHECK WORD COUNT
5610	030736	001401				BEQ	59\$	
5611	030740	104160				ERROR	160	
5612	030742	023737	003220	003160	59\$:	CMP	E.DCYL,T.DCYL	; CHECK CYLINDER ADD
5613	030750	001401				BEQ	60\$	
5614	030752	104161				ERROR	161	

CZR6080 RK611 DSKLS CTRL PRM4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 105
T45 ZERO FILL ON WRITE DATA

SEQ 0105

```

5615 030754 023737 003206 003146 60$: CMP E.DA,T.DA ;CHECK DISK ADDR.
5616 030762 001401 BEQ TST46 ;;;YES, GO ON TO NEXT TEST
5617 030764 104162 ERROR 162
5618 .SBTTL **18 BIT FORMAT WRITES
5619
5620 *****
5621 *TEST 46 18 BIT WRITE DATA (PART 1)
5622 *
5623 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5624 * PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
5625 * OF 400 WORDS TO AN RK06 IN 24 SECTOR FOR:ST,
5626 * CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5627 * AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
5628 * AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777.
5629 * VERIFY THAT TWO ZERO BITS ARE PRESENT FOR EACH WORD.
5630 *****
5631
5632 030766 000004 TST46: SCOPE
5633 030770 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
5634
5635 030776 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5636 031002 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5637 031010 012700 002000 MOV #2000,R0 ;SET A STALL COUNT
5638 031014 005300 5$: DEC R0 ;LOOP UNTIL COUNT 0
5639 031016 001376 BNE 5$
5640
5641 031020 012762 000040 000026 MOV #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5642 031026 012762 055202 000004 MOV #MOD2BF,RKBA(R2) ;LOAD BUFFER ADDRESS
5643 031034 012762 177400 000002 MOV #-400,RKWC(R2) ;--- WORD COUNT
5644 031042 012762 010023 000000 MOV #WRDATA:CFMT,RKCS1(R2) ;--- START COMMAND
5645 031050 012700 005136 MOV #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5646 ; READY FOR SECTOR PULSE
5647
5648 031054 012762 000440 000026 7$: MOV #DMD!MCLK,RKMR1(R2)
5649 031062 012762 000040 000026 MOV #DMD,RKMR1(R2)
5650 031070 005300 DEC R0
5651 031072 001370 BNE 7$
5652
5653 031074 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5654 031102 012762 000040 000026 MOV #DMD,RKMR1(R2)
5655
5656 031110 005037 003254 CLR PR.BIT ;GENERATE SYNC
5657 031114 005037 003256 CLR M1.BIT
5658 031120 012700 000377 MOV #255.,R0
5659
5660 031124 004737 037026 9$: JSR PC,R0BIT ;SIMULATE SYNC 0 BITS
5661 031130 005300 DEC R0
5662 031132 001374 BNE 9$
5663
5664 031134 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNC 1 BIT
5665 031142 004737 037026 JSR PC,R0BIT
5666
5667 031146 012703 053502 MOV #HEAD11,R3 ;SIMULATE HEADER
5668 031152 012701 000003 MOV #3,R1 ;CYL 0 TRK 0 SEC 0
5669 031156 012304 11$: MOV (R3)+,R4 ;GET HEADER WORD
5670 031160 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
5671 031164 013737 003254 13$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT

```

5671	031172	006004				ROR	R4	;GET NEXT BIT
5672	031174	103403				BCS	15\$	
5673	031176	005037	003254			CLR	PR.BIT	
5674	031202	000403				BR	16\$	
5675								
5676	031204	012737	000001	003254	15\$:	MOV	#1,PR.BIT	
5677	031212	004737	037026		16\$:	JSR	PC,RDBIT	;SIMULATE NEXT BIT
5678	031216	005300				DEC	R0	;READY FOR NEXT HEADER WORD?
5679	031220	001361				BNE	13\$	;NO - GET NEXT BIT THIS WORD
5680	031222	005301				DEC	R1	;HEADER DONE?
5681	031224	001354				BNE	11\$	;NO - GET NEXT HEADER WORD
5682	031226	012700	000101			MOV	#65,R0	;SET COUNT FOR GAP.
5683	031232	013737	003254	003256	20\$:	MOV	PR.BIT,M1.BIT	;SIMULATE GAP
5684	031240	005037	003254			CLR	PR.BIT	
5685	031244	004737	037026			JSR	PC,RDBIT	
5686	031250	005300				DEC	R0	
5687	031252	001367				BNE	20\$	
5688	031254	012700	000400			MOV	#256,R0	;SET COUNT FOR WRITE DATA SYNC
5689	031260	012737	050670	001310		MOV	#EM327,EMW	;LOAD ERROR MESSAGE
5690	031266	005037	003252			CLR	P1.BIT	;CLEAR BITS
5691	031272	005037	003254			CLR	PR.BIT	
5692	031276	005037	003256			CLR	M1.BIT	
5693	031302	005037	003260			CLR	M2.BIT	
5694	031306	005037	003262			CLR	BITCNT	;CLEAR BIT COUNTER
5695	031312	012737	062040	003224		MOV	#DMD!ECCW!MEWD!WRTGAT,E.MR1	;SET EXPECTED MR1
5696	031320	004737	036300		22\$:	JSR	PC,WRTBIT	;SIMULATE SYNC 0
5697	031324	104002				ERROR	2	
5698	031326	005237	003262			INC	BITCNT	;BUMP BIT COUNT
5699	031332	005300				DEC	R0	;LOOP UNTIL SYNC 0 WRITTEN
5700	031334	001371				BNE	22\$	
5701	031336	012737	000001	003252		MOV	#1,P1.BIT	;SIMULATE SYNC 1
5702	031344	004737	036300			JSR	PC,WRTBIT	
5703	031350	104002				ERROR	2	
5704	031352	005037	003266			CLR	ECCHI	;INITIALIZE ECC WORDS.
5705	031356	005037	003270			CLR	ECCL0	
5706	031362	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
5707	031366	012703	055202			MOV	#MOD2BF,R3	;SET DATA POINTER
5708	031372	012701	000006			MOV	#6,R1	;SET DATA COUNT
5709	031376	012304			24\$:	MOV	(R3)+,R4	;GET DATA WORD
5710	031400	012737	051613	001310		MOV	#EM338,EMW	;LOAD MESSAGE (18 BIT DATA WRITE)
5711	031406	012700	000022			MOV	#18,R0	;LOAD BITS PER WORD
5712	031412	013737	003256	003260	25\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5713	031420	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5714	031426	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5715	031434	000241				CLC		;CLEAR CARRY & GET NEXT BIT
5716	031436	006004				ROR	R4	
5717	031440	103403				BCS	27\$	
5718	031442	005037	003252			CLR	P1.BIT	
5719	031446	000403				BR	28\$	
5720								
5721	031450	012737	000001	003252	27\$:	MOV	#1,P1.BIT	
5722	031456	004737	036300		28\$:	JSR	PC,WRTBIT	;SIMULATE NEXT BIT
5723	031462	104002				ERROR	2	
5724	031464	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	;GET ECC PATTERN
5725	031472	004737	036132			JSR	PC,ECCGEN	;COMPUTE EXPECTED ECC
5726	031476	023737	003174	003234		CMP	T.ECPT,E.ECPT	;CHECK IF CORRECT

CZR6080 RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 107
 CZR608.P11 02-DEC-77 10:00 T46 18 BIT WRITE DATA (PART 1)

SEQ 0107

```

5727 031504 001401 BEQ 29$ ;YES - SKIP
5728 031506 104164 ERROR 164
5729 031510 005237 003262 29$: INC BITCNT
5730 031514 005300 DEC R0
5731 031516 020027 000002 CMP R0,#2 ;1ST 16 BITS JUST DONE?
5732 031522 001403 BEQ 31$ ;YES - SKIP
5733 031524 005700 TST R0 ;ELSE TEST IF WORD DONE
5734 031526 001331 BNE 25$ ;NO - DO NEXT BIT
5735 031530 000406 BR 32$ ;ELSE DO NEXT WORD
5736 031532 012737 051701 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5737 031540 012704 000000 MOV #0,R4 ;SET UPPER BITS OF WORD
5738 031544 000722 BR 25$ ;GO DO BITS
5739
5740 031546 005301 32$: DEC R1 ;ALL WORDS DONE?
5741 031550 001312 BNE 24$ ;NO - GET NEXT WORD
5742
5743 *****
5744 ;TEST 47 18 BIT WRITE DATA (PART 2)
5745 ;
5746 ; CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5747 ; PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA
5748 ; OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
5749 ; CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5750 ; AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
5751 ; AND A GOOD HEADER. CLOCK THROUGH 6 WORDS OF 177777
5752 ; WITH BAD PARITY SET. VERIFY WRITING OF TWO PARITY
5753 ; BITS ON SIMULATED DISK.
5754 ;
5755 ; NOTE: THIS TEST IS ONLY EXECUTED IF MEMORY
5756 ; PARITY ENABLE IS PRESENT FOR BUFFER
5757 ; LOCATION.
5758 ;
5759 *****
5760 031552 000004 1ST47: SCOPE
5761 031554 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
5762
5763 031562 005737 003274 TST MEMPAR ;CHECK IF MEMORY PARITY AVAIL.
5764 031566 001017 BNE 25$ ;YES - SKIP
5765 031570
5766 031570 012737 000001 001200 1$: MOV #1,$TIMES ;FORCE INTERATION COUNT TO 1
5767 031576 005227 177777 INC #-1 ;ONLY DO ONCE
5768 031602 001007 BNE 64$ ;NO, GO TO NEXT TEST
5769 031604 104401 044041 TYPE TSTBY1 ;TYPE TEST N BYPASSED
5770 031610 013746 001220 MOV $TESTN,-(SP) ;SAVE $TESTN FOR TYPEOUT
5771 031614 104402 TYPEOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
5772 031616 104401 044051 TYPE TSTBY2
5773 031622 000137 032424 64$: JMP 1ST50 ;GO TO NEXT TEST
5774
5775 031626 004237 034102 2$: JSR R2,WRTPAR
5776 031632 000006 6
5777 031634 057216 BADPAR
5778 031636 005700 TST R0
5779 031640 001353 BNE 1$
5780 031642 013737 031650 001110 MOV 3$,$LPERR
5781
5782 031650 3$:

```

```

5783
5784 031650 013702 001270      MOV    $BASE,R2      ;LOAD RK611 BASE
5785 031654 012762 100000 000000 MOV    #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5786 031662 012700 002000      MOV    #2000,R0      ;SET A STALL COUNT
5787 031666 005300      5$: DEC    R0      ;LOOP UNTIL COUNT 0
5788 031670 001376      BNE    5$
5789
5790 031672 012762 000040 000026 MOV    #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5791 031700 012762 057216 000004 MOV    #BADPAR,RKBA(R2) ;LOAD BUFFER ADDRESS
5792 031706 012762 177400 000002 MOV    #-400,RKWC(R2) ; --- WORD COUNT
5793 031714 012762 010023 000000 MOV    #WRDATA:CFMT,RKCS1(R2) ; --- START COMMAND
5794 031722 012700 005136      MOV    #66.*40.+(4.*3.+2),R0 ;ISSUE ENOUGH CLOCKS UNTIL
5795                                     ; READY FOR SECTOR PULSE
5796 031726 012762 000440 000026 7$: MOV    #DMD:MCLK,RKMR1(R2)
5797 031734 012762 000040 000026 MOV    #DMD,RKMR1(R2)
5798 031742 005300      DEC    R0
5799 031744 001370      BNE    7$
5800
5801 031746 012762 000140 000026 MOV    #DMD:MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5802 031754 012762 000040 000026 MOV    #DMD,RKMR1(R2)
5803
5804 031762 005037 003254      CLR    PR.BIT      ;GENERATE SYNC
5805 031766 005037 003256      CLR    M1.BIT
5806 031772 012700 000377      MOV    #255.,R0
5807
5808 031776 004737 037026      9$: JSR    PC,RDBIT      ;SIMULATE SYNC 0 BITS
5809 032002 005300      DEC    R0
5810 032004 001374      BNE    9$
5811
5812 032006 012737 000001 003254 MOV    #1,PR.BIT      ;SIMULATE SYNC 1 BIT
5813 032014 004737 037026      JSR    PC,RDBIT
5814
5815 032020 012703 053502      MOV    #HEAD11,R3      ;SIMULATE HEADER
5816 032024 012701 000003      MOV    #3,R1      ;CYL 0 TRK 0 SEC 0
5817 032030 012304      11$: MOV    (R3)+,R4      ;GET HEADER WORD
5818 032032 012700 000020      MOV    #16.,R0      ;LOAD BITS PER WORD
5819 032036 013737 003254 003256 13$: MOV    PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5820 032044 006004      ROR    R4      ;GET NEXT BIT
5821 032046 103403      BCS    15$
5822 032050 005037 003254      CLR    PR.BIT
5823 032054 000403      BR     16$
5824
5825 032056 012737 000001 003254 15$: MOV    #1,PR.BIT
5826 032064 004737 037026 16$: JSR    PC,RDBIT      ;SIMULATE NEXT BIT
5827 032070 005300      DEC    R0      ;READY FOR NEXT HEADER WORD?
5828 032072 001361      BNE    13$      ;NO - GET NEXT BIT THIS WORD
5829 032074 005301      DEC    R1      ;HEADER DONE?
5830 032076 001354      BNE    11$      ;NO - GET NEXT HEADER WORD
5831 032100 012700 000101      MOV    #65.,R0      ;SET COUNT FOR GAP.
5832 032104 013737 003254 003256 20$: MOV    PR.BIT,M1.BIT ;SIMULATE GAP
5833 032112 005037 003254      CLR    PR.BIT
5834 032116 004737 037026      JSR    PC,RDBIT
5835 032122 005300      DEC    R0
5836 032124 001367      BNE    20$
5837 032126 012700 000400      MOV    #256.,R0      ;SET COUNT FOR WRITE DATA SYNC
5838 032132 012737 050670 001310 MOV    #EM327,EMW      ;LOAD ERROR MESSAGE

```

```

5839 032140 005037 003252 CLR P1.BIT ;CLEAR BITS
5840 032144 005037 003254 CLR PR.BIT
5841 032150 005037 003256 CLR M1.BIT
5842 032154 005037 003260 CLR M2.BIT
5843 032160 005037 003262 CLR BITCNT ;CLEAR BIT COUNTER
5844 032164 012737 062040 003224 22$: MOV #0MD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5845 032172 004737 036300 JSR PC,WRTBIT ;SIMULATE SYNC 0
5846 032176 104002 ERROR 2
5847 032200 005237 003262 INC BITCNT ;BUMP BIT COUNT
5848 032204 005300 DEC R0 ;LOOP UNTIL SYNC 0 WRITTEN
5849 032206 001371 BNE 22$
5850 032210 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNC 1
5851 032216 004737 036300 JSR PC,WRTBIT
5852 032222 104002 ERROR 2
5853 032224 005037 003266 CLR ECCHI ;INITIALIZE ECC WORDS.
5854 032230 005037 003270 CLR ECCLO
5855 032234 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5856 032240 012703 055202 MOV #MOD2BF,R3 ;SET DATA POINTER
5857 032244 012701 000006 MOV #6,R1 ;SET DATA COUNT
5858 032250 012304 24$: MOV (R3)+,R4 ;GET DATA WORD
5859 032252 012737 051613 001310 MOV #EM338,EMW ;LOAD MESSAGE (18 BIT DATA WRITE,
5860 032260 012700 000022 MOV #18,R0 ;LOAD BITS PER WORD
5861 032264 013737 003256 003260 25$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5862 032272 013737 003254 003256 MOV PR.BIT,M1.BIT
5863 032300 013737 003252 003254 MOV P1.BIT,PR.BIT
5864 032306 000241 CLC ;CLEAR CARRY & GET NEXT BIT
5865 032310 006004 ROR R4
5866 032312 103403 BCS 27$
5867 032314 005037 003252 CLR P1.BIT
5868 032320 000403 BR 28$
5869
5870 032322 012737 000001 003252 27$: MOV #1,P1.BIT
5871 032330 004737 036300 28$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5872 032334 104002 ERROR 2
5873 032336 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET ECC PATTERN
5874 032344 004737 036132 JSR PC,ECCGEN ;COMPUTE EXPECTED ECC
5875 032350 023737 003174 003234 CMP T.ECPT,E.ECPT ;CHECK IF CORRECT
5876 032356 001401 BEQ 29$ ;YES - SKIP
5877 032360 104164 ERROR 164
5878 032362 005237 003262 29$: INC BITCNT
5879 032366 005300 DEC R0
5880 032370 020027 000002 CMP R0,#2 ;1ST 16 BITS JUST DONE?
5881 032374 001403 BEQ 31$ ;YES - SKIP
5882 032376 005700 TST R0 ;ELSE TEST IF WORD DONE
5883 032400 001331 BNE 25$ ;NO - DO NEXT BIT
5884 032402 000406 BR 32$ ;ELSE DO NEXT WORD
5885 032404 012737 051701 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5886 032412 012704 000002 MOV #2,R4 ;SET UPPER BITS OF WORD
5887 032416 000722 BR 25$ ;GO DO BITS
5888
5889 032420 005301 32$: DEC R1 ;ALL WORDS DONE?
5890 032422 001312 BNE 24$ ;NO - GET NEXT WORD
5891
5892
5893 ;*****
5894 ;*TEST 50 CLEAR BAD PARITY BUFFER

```


G09

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 110
T50 CLEAR BAD PARITY BUFFER

SEQ 0110

```

5895
5896
5897
5898
5899
5900 032424 000004
5901 032426 012737 000001 001200
5902 032434 012700 000006
5903 032440 012701 057216
5904 032444 012721 177777
5905 032450 005300
5906 032452 001374
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920 032454 000004
5921 032456 012737 000012 001200
5922
5923
5924 032464 013702 001270
5925 032470 012762 100000 000000
5926 032476 012700 002000
5927 032502 005300
5928 032504 001376
5929
5930 032506 012762 000040 000026
5931 032514 012762 056216 000004
5932 032522 012762 177400 000002
5933 032530 012762 010023 000000
5934 032536 012700 005136
5935
5936 032542 012762 000440 000026
5937 032550 012762 000040 000026
5938 032556 005300
5939 032560 001370
5940
5941 032562 012762 000140 000026
5942 032570 012762 000040 000026
5943
5944 032576 005037 003254
5945 032602 005037 003256
5946 032606 012700 000377
5947
5948 032612 004737 037026
5949 032616 005300
5950 032620 001374

;
; THE SOLE PURPOSE OF THIS TEST IS TO CLEAR BADPAR
; BUFFER AND REMOVE THE BAD PARITY
;
; *****
; T50: SCOPE
;      MOV      #1, $TIMES      ;; DO 1 ITERATION
;      MOV      #6, R0
;      MOV      #BADPAR, R1
; 1$:  MOV      #-1, (R1)+
;      DEC      R0
;      BNE      1$
; *****
; TEST S1      18 BIT WRITE DATA (PART 3)
;
; CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
; PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
; OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
; CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
; AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
; AND A GOOD HEADER.  CLOCK THROUGH 400 18 BIT WORDS
; AND THE 32 BIT ECC.  VERIFY THAT THE ECC IS
; WRITTEN CORRECTLY.
; *****
; T51: SCOPE
;      MOV      #10, $TIMES      ;; DO 10. ITERATIONS
;
;      MOV      $BASE, R2      ; LOAD RK611 BASE
;      MOV      #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
;      MOV      #2000, R0      ; SET A STALL COUNT
; 5$:  DEC      R0      ; LOOP UNTIL COUNT 0
;      BNE      5$
;
;      MOV      #DMD, RKMR1(R2) ; SET DIAGNOSTIC MODE
;      MOV      #ECCBUF, RKBA(R2) ; LOAD BUFFER ADDRESS
;      MOV      #-400, RKWC(R2) ; --- WORD COUNT
;      MOV      #WRDATA!CFMT, RKCS1(R2) ; --- START COMMAND
;      MOV      #66.*40.+<4.*3.+2>, R0 ; ISSUE ENOUGH CLOCKS UNTIL
;                                     ; READY FOR SECTOR PULSE
;
; 7$:  MOV      #DMD!MCLK, RKMR1(R2)
;      MOV      #DMD, RKMR1(R2)
;      DEC      R0
;      BNE      7$
;
;      MOV      #DMD!MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
;      MOV      #DMD, RKMR1(R2)
;
;      CLR      PR.BIT      ; GENERATE SYNC
;      CLR      M1.BIT
;      MOV      #255., R0
;
; 9$:  JSR      PC, R0BIT      ; SIMULATE SYNC 0 BITS
;      DEC      R0
;      BNE      9$

```

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	Op 381	Op 382	Op 383	Op 384	Op 385	Op 386	Op 387	Op 388	Op 389	Op 390	Op 391	Op 392	Op 393	Op 394	Op 395	Op 396	Op 397	Op 398	Op 399	Op 400	Op 401	Op 402	Op 403	Op 404	Op 405	Op 406	Op 407	Op 408	Op 409	Op 410	Op 411	Op 412	Op 413	Op 414	Op 415	Op 416	Op 417	Op 418
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

6007	033130	005037	003252		CLR	P1.BIT	
6008	033134	000403			BR	28\$	
6009							
6010	033136	012737	000001	003252	27\$:	MOV	#1,P1.BIT
6011	033144	004737	036300		28\$:	JSR	PC,WRTBIT ;SIMULATE NEXT BIT
6012	033150	104002				ERROR	2
6013	033152	016237	000032	003174		MOV	RKECPT(R2),T.ECPT ;GET ECC PATTERN
6014	033160	004737	036132			JSR	PC,ECCGEN ;COMPUTE EXPECTED ECC
6015	033164	023737	003174	003234		CMP	T.ECPT,E.ECPT ;CHECK IF CORRECT
6016	033172	001401				BEQ	29\$;YES - SKIP
6017	033174	104164				ERROR	164
6018	033176	005237	003262		29\$:	INC	BITCNT
6019	033202	005300				DEC	R0
6020	033204	020027	000002			CMP	R0,#2 ;1ST 16 BITS JUST DONE?
6021	033210	001403				BEQ	31\$;YES - SKIP
6022	033212	005700				TST	R0 ;ELSE TEST IF WORD DONE
6023	033214	001331				BNE	25\$;NO - DO NEXT BIT
6024	033216	000406				BR	32\$;ELSE DO NEXT WORD
6025	033220	012737	051701	001310	31\$:	MOV	#EM339,EMW ;LOAD MESSAGE (BIT 16.17)
6026	033226	012704	000000			MOV	#0,R4 ;SET UPPER BITS OF WORD
6027	033232	000722				BR	25\$;GO DO BITS
6028							
6029	033234	005301			32\$:	DEC	R1 ;ALL WORDS DONE?
6030	033236	001312				BNE	24\$;NO - GET NEXT WORD
6031							
6032	033240	012737	051613	001310		MOV	#EM338,EMW ;SET ERROR MESSAGE
6033	033246	012304				MOV	(R3)+,R4 ;GET LAST WORD
6034	033250	012700	000022			MOV	#18,R0 ;SET BIT COUNT
6035	033254	013737	003256	003260	34\$:	MOV	M1.BIT,M2.BIT ;SHIFT BITS
6036	033262	013737	003254	003256		MOV	PR.BIT,M1.BIT
6037	033270	013737	003252	003254		MOV	P1.BIT,PR.BIT
6038	033276	000241				CLC	;CLEAR CARRY
6039	033300	006004				ROR	R4 ;TEST NEXT BIT
6040	033302	103403				BCS	36\$;SKIP IF A ONE
6041	033304	005037	003252			CLR	P1.BIT ;ELSE SET NEXT BIT FOR ZERO
6042	033310	000403				BR	37\$
6043							
6044	033312	012737	000001	003252	36\$:	MOV	#1,P1.BIT ;SET NEXT BIT FOR 1
6045	033320	004737	036300		37\$:	JSR	PC,WRTBIT ;SIMULATE BIT
6046	033324	104002				ERROR	2
6047	033326	016237	000032	003174		MOV	RKECPT(R2),T.ECPT ;GET PATTERN
6048	033334	004737	036132			JSR	PC,ECCGEN ;GENERATE ECC PATTERN
6049	033340	023737	003174	003234		CMP	T.ECPT,E.ECPT ;TEST IF CORRECT
6050	033346	001401				BEQ	39\$;YES - SKIP
6051	033350	104164				ERROR	164 ;ELSE REPORT
6052	033352	005237	003262		39\$:	INC	BITCNT ;BUMP BIT COUNT
6053	033356	022700	000002			CMP	#2,R0 ;IF NEXT BIT IS THE LAST DATA BIT
6054	033362	001003				BNE	41\$;ECCW MUST BE RESET TO INDICATE
6055	033364	042737	020000	003224		BIC	#ECCW,E.MR1 ;ECC BEING WRITTEN
6056	033372	005300			41\$:	DEC	R0
6057	033374	022700	000002			CMP	#2,R0 ;TEST IF 1ST 16 BITS DONE
6058	033400	001403				BEQ	42\$;YES - SKIP
6059	033402	005700				TST	R0 ;ELSE TEST IF WORD DONE
6060	033404	001323				BNE	34\$;NO - SKIP
6061	033406	000406				BR	45\$;ELSE GO TO ECC PROCESSING
6062							

6063	033410	012737	051701	001310	42\$:	MOV	#EM339,EMW	;SET MESSAGE FOR BITS 16 AND 17
6064	033416	012704	000000			MOV	#0,R4	;SET UP UPPER BITS
6065	033422	000714				BR	34\$	;GO DO LAST TWO BITS
6066	033424	012737	051307	001310	45\$:	MOV	#EM333,EMW	;MESSAGE (ECC WRITE)
6067	033432	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
6068	033436	012703	003266			MOV	#ECCHI,R3	;LOAD POINTER TO ECC WORDS
6069	033442	012701	000002			MOV	#2,R1	;SET FOR 2 WORDS
6070	033446	012304			43\$:	MOV	(R3)+,R4	;GET ECC WORD
6071	033450	012700	000020			MOV	#16,R0	;SET BIT COUNT
6072	033454	013737	003256	003260	44\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6073	033462	013737	003254	003256		MOV	PR.BIT,M1.BIT	
6074	033470	013737	003252	003254		MOV	P1.BIT,PR.BIT	
6075	033476	006004				ROR	R4	;LOAD NEXT BIT
6076	033500	103403				BCS	46\$	
6077	033502	005037	003252			CLR	P1.BIT	
6078	033506	000403				BR	47\$	
6079								
6080	033510	012737	000001	003252	46\$:	MOV	#1,P1.BIT	
6081	033516	004737	036300		47\$:	JSR	PC,WRTBIT	;SIMULATE NEXT BIT
6082	033522	104002				ERROR	2	
6083	033524	022700	000002			CMP	#2,R0	;IF THIS BIT IS THE LAST ECC BIT
6084	033530	001006				BNE	49\$	;ECCW MUST BE SET TO INDICATE
6085	033532	022701	000001			CMP	#1,R1	;ECC IS DONE
6086	033536	001003				BNE	49\$	
6087	033540	052737	020000	003224		BIS	#ECCW,E.MR1	
6088	033546	005237	003262		49\$:	INC	BITCNT	;BUMP BIT COUNT
6089	033552	005300				DEC	R0	;TEST IF LAST BIT THIS WORD IS DONE
6090	033554	001337				BNE	44\$	;NO - LOOP
6091	033556	005301				DEC	R1	;TEST IF BOTH WORDS WRITTEN
6092	033560	001332				BNE	43\$	;NO - LOOP
6093	033562	012737	051345	001310		MOV	#EM334,EMW	;LOAD MESSAGE (POSTAMBLE)
6094	033570	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
6095	033574	012700	000017			MOV	#15,R0	;SET GAP COUNT
6096	033600	013737	003256	003260	51\$:	MOV	M1.BIT,M2.BIT	;SHIFT REST OF BITS
6097	033606	013737	003254	003256		MOV	PR.BIT,M1.BIT	
6098	033614	013737	003252	003254		MOV	P1.BIT,PR.BIT	
6099	033622	005037	003252			CLR	P1.BIT	;CLEAR NEXT BIT
6100	033626	004737	036300			JSR	PC,WRTBIT	;SIMULATE BIT
6101	033632	104002				ERROR	2	
6102	033634	005237	003262			INC	BITCNT	;BUMP BIT COUNT
6103	033640	005300				DEC	R0	;GAP WRITTEN?
6104	033642	001356				BNE	51\$	;NO - LOOP
6105	033644	012737	000001	003234		MOV	#1,E.ECPT	;SET EXPECTED PATTERN
6106	033652	016237	000032	003174		MOV	RKECPT(R2),T.ECPT	;GET '611 PATTERN
6107	033660	023737	003174	003234		CMP	T.ECPT,E.ECPT	;TEST IF EQUAL
6108	033666	001401				BEQ	56\$	;YES - SKIP
6109	033670	104163				ERROR	163	;ELSE REPORT
6110	033672				56\$:			

```

6111
6112
6113
6114
6115
6116
6117
6118
6119
6120 033672
6121 033672 000004
6122 033674 005037 001102
6123 033700 005037 001200
6124 033704 005237 001222
6125 033710 042737 100000 001222
6126 033716 005327
6127 033720 000001
6128 033722 003063
6129 033724 012737
6130 033726 000001
6131 033730 033720
6132 033732 104401 033740
6133 033736 000407
6134
6135 033756
6136 033756 013746 001222
6137
6138 033762 104405
6139 033764 104401 033772
6140 033770 000421
6141
6142 034034
6143 034034 013746 001112
6144
6145 034040 104405
6146 034042 104401 001211
6147 034046 005037 001112
6148 034052 013700 000042
6149 034056 001405
6150 034060 000005
6151 034062 004710
6152 034064 000240
6153 034066 000240
6154 034070 000240
6155 034072
6156 034072 000137
6157 034074 004334
6158 034076 377 377 000
6159 034102
6160
6161
6162
6163
6164
6165
6166

.SBTTL END OF PASS ROUTINE
;*****
;INCREMENT THE PASS NUMBER ($PASS)
;TYPE "END PASS *XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
;WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
;IF THERES A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO TST1

SEOP:
SCOPE
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
DEC (PC)+ ;;LOOP?

SEOPCT: .WORD 1
BGT $DOAGN ;;YES
MOV (PC)+,$(PC)+ ;;RESTORE COUNTER

SENDCT: .WORD 1
SEOPCT
TYPE 65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
65$: .ASCIZ <12><15>/END PASS #/
64$: MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
;;TYPE PASS NUMBER
;;GO TYPE--DECIMAL ASCII WITH SIGN
TYPDS
TYPE 67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$: MOV $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
;;TOTAL NUMBER OF ERRORS
;;GO TYPE--DECIMAL ASCII WITH SIGN
TYPDS
TYPE $CRLF ;;TYPE CARRIAGE RETURN, LINE FEED
CLR $ERTTL ;;CLEAR ERROR TOTAL
$GET42: MOV #42,R0 ;;GET MONITOR ADDRESS
BEQ $DOAGN ;;BRANCH IF NO MONITOR
RESET ;;CLEAR THE WORLD
SENDAD: JSR PC,(R0) ;;GO TO MONITOR
NOP ;;SAVE ROOM
NOP ;;FOR
NOP ;;ACT11

$DOAGN: JMP $(PC)+ ;;RETURN
$RTNAD: .WORD TST1
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
.EVEN

.SBTTL GENERATE BAD PARITY IN MEMORY
;THIS ROUTINE ATTEMPTS TO GENERATE BAD PARITY WORDS IN
;MEMORY. THE NUMBER OF WORDS TO BE GENERATED AND THE AREA
;WHERE THEY ARE TO BE PLACED IS SPECIFIED IN THE WORDS
;FOLLOWING THE CALL. IF ALL THE SPECIFIED WORDS CANNOT
;BE GENERATED, THE BUFFER IS CLEARED OF BAD PARITY WORDS.

```

```
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179 034102 011200
6180 034104 016201 000002
6181 034110 012703 172100
6182 034114 013704 003274
6183 034120 012705 000020
6184 034124 006004
6185 034126 103002
6186 034130 012713 000005
6187 034134 062703 000002
6188 034140 005305
6189 034142 001370
6190 034144 012721 177777
6191 034150 005300
6192 034152 001374
6193 034154 012703 172100
6194 034160 013704 003274
6195 034164 012705 000020
6196 034170 006004
6197 034172 103002
6198 034174 012713 000001
6199 034200 062703 000002
6200 034204 005305
6201 034206 001370
6202 034210 012737 034360 000114
6203 034216 011200
6204 034220 016201 000002
6205 034224 010004
6206 034226 005721
6207 034230 000240
6208 034232 000240
6209 034234 005300
6210 034236 001373
6211 034240 005704
6212 034242 001444
6213 034244 011200
6214 034246 016201 000002
6215 034252 012721 000000
6216 034256 005300
6217 034260 001374
6218 034262 012700 177777
6219 034266 012703 172100
6220 034272 013704 003274
6221 034276 012705 000020
6222 034302 006004

; *
; * RO IS USED AS A FLAG TO INDICATE IF BAD PARITY WAS
; * GENERATED. IF RO IS 0 WHEN THE ROUTINE RETURNS TO CALLER
; * PARITY WAS MADE BAD IN ALL THE DESIRED WORDS. IF RO IS ALL
; * ONES IT WAS NOT SUCCESSFUL.
; *
; * CALL: JSR R2,WRTPAR
; * POINTER TO AREA TO BE ALTERED
; * NUMBER OF WORDS
; * RETURN: RTS R2
; * WITH RO AS SUCCESS FLAG
; *
WRTPAR: MOV (R2),RO ;GET BUFFER POINTER
MOV 2(R2),R1 ;GET NUMBER OF WORDS
MOV #MEMBAS,R3 ;GET BASE OF VECTOR AREA
MOV MEMPAR,R4 ;STORE FLAGS
MOV #16.,R5 ;GET NUMBER OF REGISTERS
5$: ROR R4 ;CHECK IF PARITY ENABLE ON THIS BANK
BCC 7$ ;NO, TRY NEXT BANK
MOV #WR.PAR,(R3) ;ALLOW SETTING OF BAD PARITY
7$: ADD #2,R3 ;CALCULATE NEXT ADDRESS
DEC R5 ;CHECK IF FINISHED
5$ BNE 5$ ;NO, GET NEXT PARITY MEMORY ADDRESS
8$: MOV #-1,(R1)+ ;WRITE BAD PARITY
DEC RO ;LOOP UNTIL ALL LOCATIONS WRITTEN
BNE 8$
MOV #MEMBAS,R3 ;LOAD BASE OF CONTROL REGS
MOV MEMPAR,R4 ;LOAD FLAGS
MOV #16.,R5 ;GET NUMBER OF REGISTERS
15$: ROR R4 ;CHECK IF PARITY ENABLE ON THIS BANK
BCC 17$
MOV #PAR.EN,(R3) ;ALLOW PARITY DETECTION
17$: ADD #2,R3 ;CHECK IF FINISHED
DEC R5 ;CHECK IF FINISHED
15$ BNE 15$ ;NO, GET NEXT PARITY MEMORY ADDRESS
MOV #30$,MEMVEC ;SET PARITY TRAP VECTOR
MOV (R2),RO ;NUMBER OF WORDS
MOV 2(R2),R1 ;BUFFER POINTER
MOV RO,R4 ;STORE NUMBER
10$: TST (R1)+ ;TEST FIRST BUFFER WORD - SHOULD GET
NOP ;TRAP WHICH DECREMENTS R4
NOP
11$: DEC RO ;DEC NUMBER OF WORDS COUNTER
BNE 10$ ;ALL CHECKED? NO - SKIP
R4 ;TEST IF ALL LOCATIONS BAD
BEQ 20$ ;YES - SKIP
MOV (R2),RO ;GET NUMBER OF WORDS
MOV 2(R2),R1 ;BUFFER POINTER
12$: MOV #0,(R1)+ ;CLEAR ALL OF AREA THAT WAS MADE BAD
DEC RO
BNE 12$
MOV #-1,RO ;SET FLAG THAT COULDN'T SET BAD PARITY
13$: MOV #MEMBAS,R3 ;LOAD BASE OF CONTROL REGISTERS
MOV MEMPAR,R4 ;NOW SET UP TO CLEAR PARITY
MOV #16.,R5 ;CONTROL REGISTERS AND SET
14$: ROR R4 ;ENABLE WHERE APPLICABLE
```

6223	034304	103002			BCC	18\$	
6224	034306	012713	000001		MOV	#PAR.EN, (R3)	
6225	034312	062703	000002		18\$: ADD	#2, R3	
6226	034316	005305			DEC	R5	
6227	034320	001370			BNE	14\$	
6228	034322	022222			CMP	(P2)+, (R2)+	; SET POINTER FOR RETURN
6229	034324	012737	034526	000114	MOV	#MEMEARR, MEMVEC	; RESET TRAP VECTOR
6230	034332	012737	000340	000116	MOV	#PR7, MEMVEC+2	
6231	034340	012746	000000		MOV	#PRO, -(SP)	; RESET PRIOTITY TO 0
6232	034344	012746	034352		MOV	#19\$, -(SP)	
6233	034350	000002			RTI		
6234							
6235	034352	000202			19\$: RTS	R2	
6236							
6237	034354	005000			20\$: CLR	R0	; CLEAR FLAG - BAD PARITY SET UP
6238	034356	000743			BR	13\$	
6239							
6240	034360	005304			30\$: DEC	R4	; DEC THE COUNTER
6241	034362	022626			CMP	(SP)+, (SP)+	; CLEAN STACK
6242	034364	000723			BR	11\$	; GO CHECK NEXT LOCATION
6243							
6244					.SBTTL	CHECK FOR MEMORY CHECK ENABLE	
6245					;*	THIS ROUTINE CHECKS ALL AVAILABLE MEMORY FOR PARITY OPTIC::	
6246					;*	IF OPTION IS PRESENT, PARITY DETECT IS ENABLED ON THAT BANK.	
6247					;*		
6248					;*	CALL: JSR PC, PARCHK	
6249					;*	RETURN: RTS PC	
6250							
6251	034366	012737	034464	000004	PARCHK: MOV	#20\$, ERRVEC	; SET VECTOR FOR MEMORY PARITY CHECK
6252	034374	012737	000340	000006	MOV	#PR7, ERRVEC+2	
6253	034402	005037	003274		CLR	MEMPAR	; CLEAR FLAG
6254	034406	012703	172100		MOV	#MEMBAS, R3	; LOAD REGISTER TO DETERMINE IF
6255							; MEMORY CHECK ENABLE AVAILABLE
6256	034412	012704	000001		MOV	#1, R4	; INITIALIZE MASK
6257	034416	012713	000001		16\$: MOV	#PAR.EN, (R3)	; ENABLE MEMORY CHECK
6258	034422	005713			TST	(R3)	
6259	034424	050437	003274		BIS	R4, MEMPAR	; SET FLAG
6260	034430	052713	007740		1\$: BIS	#7740, (R3)	; SET BIT 5 TO BIT 11
6261	034434	032713	007740		BIT	#7740, (R3)	; ARE THEY SETED ?
6262	034440	001003			BNE	2\$	
6263	034442	005037	003274		CLR	MEMPAR	
6264	034446	000411			BR	22\$	
6265	034450				2\$:		
6266							; THE ABOVE 6 LINES ADDED 24-OCT-77
6267	034450	062703	000002		ADD	#2, R3	; BUMP TO NEXT CSR
6268	034454	000241			CLC		
6269	034456	006104			ROL	R4	; CHECK IF FINISHED
6270	034460	001356			BNE	16\$	; NO, SET UP NEXT MEMORY PARITY MODULE
6271	034462	000403			BR	22\$	; RESTORE TRAP VECTOR
6272							
6273	034464	022626			20\$: CMP	(SP)+, (SP)+	; ADJUST STACK
6274	034466	005037	003274		CLR	MEMPAR	; 24-OCT-77
6275					ADD	#2, R3	
6276					CLC		
6277					ROL	R4	; CHECK IF FINISHED
6278					BNE	16\$	; NO, GET NEXT LOCATION

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 117
CHECK FOR MEMORY CHECK ENABLE

SEQ 0117

```

6279 034472 012737 000006 000004 22$: MOV #ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
6280 034500 005037 000006 CLR ERRVEC+2
6281 034504 005737 003274 TST MEMPAR ;CHECK IF MEMORY CHECK ENABLE AVAILIABLE
6282 034510 001005 BNE 25$ ;YES, RETURN
6283 034512 012737 000116 000114 MOV #MEMVEC+2,MEMVEC ;RESTORE TRAP CATCHER
6284 034520 005037 000116 CLR MEMVEC+2
6285 034524 000207 25$: RTS PC ;RETURN
6286 .SBTTL MEMORY CHECK ENABLE TRAP
6287 ;* THIS ROUTINE HANDLES ANY PARITY TRAP. THE LOCATION WHERE
6288 ;* PARITY WAS BAD IS REPORTED.
6289
6290 034526 012737 034542 001202 MEMERR: MOV #10$, $ESCAPE ;LOAD ESCAPE
6291 034534 011637 003244 MOV (SP),TRAPPC ;STORE PC
6292 034540 104001 ERROR 1 ;REPORT MEM PARITY ERROR
6293 034542 005037 001202 10$: CLR $ESCAPE ;CLEAR ESCAPE
6294 034546 032777 00100C 144364 BIT #SW9,$SWR ;CHECK IF LOOP ON ERROR
6295 034554 001001 BNE 15$ ;YES, FORCE STACK AND TRY AGAIN
6296 034556 000002 RTI ;NO, RETURN
6297
6298 034560 012706 001100 15$: MOV #STACK,SP ;INITIALIZE STACK
6299 034564 000177 144320 JMP $SLPERR ;LOOP ON ERROR
6300
6301 .SBTTL CONTROLLED HALT ROUTINE
6302 ;* THIS ROUTINE IS ENTERED WHEN A ^C IS ENTERED IN THE KEYBOARD.
6303 ;* IF NO MONITOR IS PRESENT THE PROGRAM HALTS ELSE THE PROGRAM
6304 ;* IS FORCED TO LAST PASS AND JUMPS TO END OF PASS.
6305
6306 034570 013702 001270 000000 CTRHLT: MOV $BASE,R2 ;SET '611 BASE
6307 034574 012762 100000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
6308 034602 104401 043631 TYPE OPRODS ;TYPE HALT MESSAGE
6309 034606 012706 001100 MOV #STACK,SP ;CLEAR STACK
6310 034612 005037 001202 CLR $ESCAPE ;CLEAR ESCAPE
6311 034616 105037 001103 CLRB $ERFLG ;CLEAR ERROR FLAG
6312 034622 005737 000042 TST 42 ;TEST IF STAND ALONE
6313 034626 001404 BEQ 1$ ;YES - SKIP
6314 034630 005037 033720 CLR $EOPCT ;ZERO PASS COUNT
6315 034634 000137 033672 JMP $EOP ;GO TO END OF PASS
6316
6317 034640 000000 1$: HALT ;HALT PROGRAM
6318 034642 000137 003340 JMP START1 ;GO TO RESTART IF CONTINUE
6319
6320 .SBTTL SCOPE HANDLER ROUTINE
6321
6322 ;*****
6323 ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
6324 ;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
6325 ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
6326 ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6327 ;*SW14=1 LOOP ON TEST
6328 ;*SW11=1 INHIBIT ITERATIONS
6329 ;*SW09=1 LOOP ON ERROR
6330 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
6331 ;*CALL
6332 ;* SCOPE ;;SCOPE=IOT
6333
6334 034646 $SCOPE:

```


6335	034646	104407				CKSWR		TEST FOR CHANGE IN SOFT-SWR
6336	034650	032777	040000	144262	1\$:	BIT	#BIT14,2SWR	LOOP ON PRESENT TEST?
6337	034656	001131				BNE	\$OVER	YES IF SW14=1
6338								TESTER*****
6339	034660	000416				*****START OF CODE FOR THE XOR		IF RUNNING ON THE "XOR" TESTER CHANGE
6340						\$XTSTR: BR	6\$	THIS INSTRUCTION TO A "NOP" (NOP=240)
6341	034662	013746	000004			MOV	2\$ERRVEC, -(SP)	SAVE THE CONTENTS OF THE ERROR VECTOR
6342	034666	012737	034706	000004		MOV	5\$2\$ERRVEC	SET FOR TIMEOUT
6343	034674	005737	177060			TST	2\$177060	TIME OUT ON XOR?
6344	034700	012637	000004			MOV	(SP)+, 2\$ERRVEC	RESTORE THE ERROR VECTOR
6345	034704	000500				BR	\$SVLAD	GO TO THE NEXT TEST
6346	034706	022626			5\$:	CMP	(SP)+, (SP)+	CLEAR THE STACK AFTER A TIME OUT
6347	034710	012637	000004			MOV	(SP)+, 2\$ERRVEC	RESTORE THE ERROR VECTOR
6348	034714	000440				BR	7\$	LOOP ON THE PRESENT TEST
6349	034716							TESTER*****
6350	034716	032777	000400	144214	6\$:	*****END OF CODE FOR THE XOR		LOOP ON SPEC. TEST?
6351	034724	001421				BIT	#BIT08,2SWR	BR IF NO
6352	034726	005046				BEQ	2\$	CLEAR A TEMP. LOCATION
6353	034730	117716	144204			CLR	-(SP)	PICKUP THE DESIRED TEST NUMBER
6354	034734	001414				MOVB	2\$SWR, (SP)	BRANCH IF BAD TEST NUMBER IN SWR
6355	034736	022716	000051			BEQ	8\$	CHECK THE NUMBER IN THE SWR
6356	034742	002411				CMP	5\$1, (SP)	BRANCH IF TEST NUMBER IS OUT OF RANGE
6357	034744	011637	001102			BLT	8\$	UPDATE THE TEST NUMBER
6358	034750	005316				MOV	(SP), \$TSTNM	BACKUP BY ONE
6359	034752	006316				DEC	(SP)	SCALE THE TEST NUMBER AS AN INDEX
6360	034754	062716	035160			ASL	(SP)	FORM THE ADDRESS OF TEST POINTER
6361	034760	013637	001106			ADD	5\$SW08TBL, (SP)	SET LOOP ADDRESS TO DESIRED TEST
6362	034764	000466				MOV	2(SP)+, \$LPADR	GO LOOP ON THE TEST
6363	034766	005726				BR	\$OVER	CLEAN THE BAD TEST NUMBER OFF OF THE STACK
6364	034770	105737	001103		8\$:	TST	(SP)+	HAS AN ERROR OCCURRED?
6365	034774	001421			2\$:	TSTB	\$ERFLG	BR IF NO
6366	034776	123737	001115	001103		BEQ	3\$	MAX. ERRORS FOR THIS TEST OCCURRED?
6367	035004	101015				CMPB	\$ERMAX, \$ERFLG	BR IF NO
6368	035006	032777	001000	144124		BHI	3\$	LOOP ON ERROR?
6369	035014	001404				BIT	#BIT09,2SWR	BR IF NO
6370	035016	013737	001110	001106	7\$:	BEQ	4\$	SET LOOP ADDRESS TO LAST SCOPE
6371	035024	000446				MOV	\$LPERR, \$LPADR	
6372	035026	105037	001103			BR	\$OVER	
6373	035032	005037	001200		4\$:	CLRB	\$ERFLG	ZERO THE ERROR FLAG
6374	035036	000415				CLR	\$TIMES	CLEAR THE NUMBER OF ITERATIONS TO MAKE
6375	035040	032777	004000	144072	3\$:	BR	1\$	ESCAPE TO THE NEXT TEST
6376	035046	001011				BIT	#BIT11,2SWR	INHIBIT ITERATIONS?
6377	035050	005737	001222			BNE	1\$	BR IF YES
6378	035054	001406				TST	\$PASS	IF FIRST PASS OF PROGRAM
6379	035056	005237	001104			BEQ	1\$	INHIBIT ITERATIONS
6380	035062	023737	001200	001104		INC	\$ICNT	INCREMENT ITERATION COUNT
6381	035070	002024				CMP	\$TIMES, \$ICNT	CHECK THE NUMBER OF ITERATIONS MADE
6382	035072	012737	000001	001104	1\$:	BGE	\$OVER	BR IF MORE ITERATION REQUIRED
6383	035100	013737	035156	001200		MOV	#1, \$ICNT	REINITIALIZE THE ITERATION COUNTER
6384	035106	105237	001102			MOV	\$MXCNT, \$TIMES	SET NUMBER OF ITERATIONS TO DO
6385	035112	113737	001102	001220	\$SVLAD:	INCB	\$TSTNM	COUNT TEST NUMBERS
6386	035120	011637	001106			MOVB	\$TSTNM, \$TESTN	SET TEST NUMBER IN APT MAILBOX
6387	035124	011637	001110			MOV	(SP), \$LPADR	SAVE SCOPE LOOP ADDRESS
6388	035130	005037	001202			MOV	(SP), \$LPERR	SAVE ERROR LOOP ADDRESS
6389	035134	112737	000001	001115		CLR	\$ESCAPE	CLEAR THE ESCAPE FROM ERROR ADDRESS
6390	035142	013777	001102	143772	\$OVER:	MOVB	#1, \$ERMAX	ONLY ALLOW ONE(1) ERROR ON NEXT TEST
						MOV	\$TSTNM, 2\$DISPLAY	DISPLAY TEST NUMBER

```

6391 035150 013716 001106      MOV      $LPADR, (SP)      ;: FUDGE RETURN ADDRESS
6392 035154 000002      RTI          ;: FIXES PS
6393 035156 003720      $MXCNT: 2000.      ;: MAX. NUMBER OF ITERATIONS
6394 035160      $SW08TBL:
6395 035160 004336      .WORD      TST1+2      ;: STARTING ADDRESS OF TEST 1
6396 035162 004570      .WORD      TST2+2      ;: STARTING ADDRESS OF TEST 2
6397 035164 005022      .WORD      TST3+2      ;: STARTING ADDRESS OF TEST 3
6398 035166 005254      .WORD      TST4+2      ;: STARTING ADDRESS OF TEST 4
6399 035170 005504      .WORD      TST5+2      ;: STARTING ADDRESS OF TEST 5
6400 035172 005734      .WORD      TST6+2      ;: STARTING ADDRESS OF TEST 6
6401 035174 006164      .WORD      TST7+2      ;: STARTING ADDRESS OF TEST 7
6402 035176 006450      .WORD      TST10+2     ;: STARTING ADDRESS OF TEST 10
6403 035200 006720      .WORD      TST11+2     ;: STARTING ADDRESS OF TEST 11
6404 035202 007170      .WORD      TST12+2     ;: STARTING ADDRESS OF TEST 12
6405 035204 007444      .WORD      TST13+2     ;: STARTING ADDRESS OF TEST 13
6406 035206 010114      .WORD      TST14+2     ;: STARTING ADDRESS OF TEST 14
6407 035210 010540      .WORD      TST15+2     ;: STARTING ADDRESS OF TEST 15
6408 035212 011100      .WORD      TST16+2     ;: STARTING ADDRESS OF TEST 16
6409 035214 011472      .WORD      TST17+2     ;: STARTING ADDRESS OF TEST 17
6410 035216 012066      .WORD      TST20+2     ;: STARTING ADDRESS OF TEST 20
6411 035220 012462      .WORD      TST21+2     ;: STARTING ADDRESS OF TEST 21
6412 035222 013056      .WORD      TST22+2     ;: STARTING ADDRESS OF TEST 22
6413 035224 013452      .WORD      TST23+2     ;: STARTING ADDRESS OF TEST 23
6414 035226 014240      .WORD      TST24+2     ;: STARTING ADDRESS OF TEST 24
6415 035230 014730      .WORD      TST25+2     ;: STARTING ADDRESS OF TEST 25
6416 035232 015324      .WORD      TST26+2     ;: STARTING ADDRESS OF TEST 26
6417 035234 015776      .WORD      TST27+2     ;: STARTING ADDRESS OF TEST 27
6418 035236 016450      .WORD      TST30+2     ;: STARTING ADDRESS OF TEST 30
6419 035240 017122      .WORD      TST31+2     ;: STARTING ADDRESS OF TEST 31
6420 035242 017574      .WORD      TST32+2     ;: STARTING ADDRESS OF TEST 32
6421 035244 020362      .WORD      TST33+2     ;: STARTING ADDRESS OF TEST 33
6422 035246 021150      .WORD      TST34+2     ;: STARTING ADDRESS OF TEST 34
6423 035250 021736      .WORD      TST35+2     ;: STARTING ADDRESS OF TEST 35
6424 035252 022426      .WORD      TST36+2     ;: STARTING ADDRESS OF TEST 36
6425 035254 023116      .WORD      TST37+2     ;: STARTING ADDRESS OF TEST 37
6426 035256 023605      .WORD      TST40+2     ;: STARTING ADDRESS OF TEST 40
6427 035260 024276      .WORD      TST41+2     ;: STARTING ADDRESS OF TEST 41
6428 035262 025012      .WORD      TST42+2     ;: STARTING ADDRESS OF TEST 42
6429 035264 025536      .WORD      TST43+2     ;: STARTING ADDRESS OF TEST 43
6430 035266 026262      .WORD      TST44+2     ;: STARTING ADDRESS OF TEST 44
6431 035270 027520      .WORD      TST45+2     ;: STARTING ADDRESS OF TEST 45
6432 035272 030770      .WORD      TST46+2     ;: STARTING ADDRESS OF TEST 46
6433 035274 031554      .WORD      TST47+2     ;: STARTING ADDRESS OF TEST 47
6434 035276 032426      .WORD      TST50+2     ;: STARTING ADDRESS OF TEST 50
6435 035300 032456      .WORD      TST51+2     ;: STARTING ADDRESS OF TEST 51
6436 ;:*****
6437 ;:SBTTL LOOP ON INTERNAL ERROR
6438
6439 035302 032777 001000 143630 SCOP1$: BIT      $SW9, $SWR      ;: CHECK IF LOOP ON ERROR
6440 035310 001405      BEQ          SS          ;: NO RETURN
6441 035312 105737 001103      TSTB      $ERFLG      ;: CHECK IF ERROR OCCURED
6442 035316 001402      BEQ          SS          ;: NO, RETURN
6443 035320 013716 001110      MOV      $LPERR, (SP)      ;: GO BACK TO BEGINNING OF LOOP
6444 035324 000002      RTI          ;: RETURN
6445
6446 ;:*****

```



```

6447          .SBTTL TYPE ERROR ROUTINE
6448          .*ENTRY JSR PC,TERR
6449          .*RETURN RTS PC
6450          .*
6451          .*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
6452          .*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
6453          .*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
6454          .*THE ERROR.
6455          .*****
6456          TYPERR: SAVREG
6457          MOVW $STSTNM,$STSTN      ;GET TEST NUMBER
6458          BIC #177400,$STSTN      ;CLEAR UNUSED BITS
6459          MOVW $ITEMB,R0           ;ENTER ERROR NUMBER
6460          BIC #177400,R0          ;CLEAR UNUSED BITS
6461          DEC R0                  ;FORM INDEX FOR ERROR TABLE
6462          ASL R0
6463          ASL R0
6464          ASL R0
6465          1$: ADD #ERRTB,R0        ;FORM ADDRESS OF ERROR ENTRY
6466          MOV (R0)+,2$           ;GET EM POINTER
6467          BEQ 3$                 ;BRANCH IF THERE ISN'T ONE
6468          TYPE ,SCRLF            ;TYPE CARRIAGE RETURN LINE FEED
6469          TYPE                   ;TYPE ERROR MESSAGE (EM)
6470          2$: .WORD 0             ;EM POINTER GOES HERE
6471          3$: MOV (R0)+,4$       ;GET DH POINTER
6472          BEQ 5$                 ;BRANCH IF THERE ISN'T ONE
6473          TYPE ,SCRLF            ;TYPE CR-LF
6474          TYPE                   ;TYPE DATA HEADER
6475          4$: .WORD 0             ;DH POINTER GOES HERE
6476          5$: MOV (R0)+,R1        ;GET DT POINTER
6477          BEQ 20$                ;BRANCH IF THERE ARE NONE
6478          CLR R4                 ;RESET INDENT SWITCH
6479          MOV (R0)+,R0           ;GET DF POINTER
6480          MOV (R0)+,R2           ;STORE NUMBER OF DH'S
6481          TYPE ,SCRLF            ;
6482          10$: MOVW (R0)+,R3      ;GET & STORE NUMBER OF DATA WORDS
6483          TSTB (R0)+            ;BUMP PAST FORMAT WORD
6484          R3                    ;TEST IF ANY DATA FOR THIS HEADER
6485          BEQ 14$                ;NO - SKIP DATA PRINT
6486          TST R4                 ;CHECK IF INDENT WORDS
6487          BNE 12$                ;YES, GO INDENT
6488          11$: MOV 2(R1)+,-(SP)  ;PUT FIRST DATA WORD ON STACK
6489          TYPOC                  ;TYPE IT
6490          DEC R3                 ;MORE DATA WORDS
6491          BEQ 13$                ;NO-BRANCH
6492          TYPE SPACE2            ;TYPE SEPARATORS
6493          BR 11$                 ;LOOP
6494          13$: TYPE ,SCRLF        ;TYPE <CR><LF>
6495          TST (R0)               ;CHECK IF NEXT HEADER AVAILABLE
6496          BEQ 14$                ;NO, DO NOT CHANGE INDENT
6497          COM R4                 ;CHANGE INDENT
6498          14$: DEC R2            ;MORE DH'S?
6499          BLE 20$                ;NO-BRANCH
6500          15$: MOV (R0)+,18$     ;GET NEXT DH POINTER
6501          BEQ 10$                ;IF NO HEADER GET DATA
6502          TST R4                 ;INDENT?

```

```

6503 035522 001402      BEQ      17$      ;NO-BRANCH
6504 035524 104401 043751 TYPE      ,SPACE2  ;INDENT
6505 035530 104401      17$:      TYPE      ;TYPE DH
6506 035532 000000      18$:      .WORD      0      ;DH POINTER GOES HERE
6507 035534 104401 001211 TYPE      ,SCRLF
6508 035540 000740      BR      10$      ;LOOP
6509 035542 104414      20$:      RESREG
6510 035544 005237 003250 INC      ERRCNT      ;INCREMENT ERROR COUNT
6511 035550 032777 010000 143362 BIT      #SW12,DSWR ;CHECK IF ABORT AFTER 20 ERRORS
6512 035556 001421      BEQ      25$      ;NO RETURN
6513 035560 022737 000024 003250 CMP      #20,ERRCNT ;CHECK IF EROR THRESHOLD EXCEEDED
6514 035566 103015      BHIS     25$      ;NO RETURN
6515 035570 104401 043754 TYPE      ,ABORT  ;TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
6516                                ;ERROR THRESHOLD EXCEEDED"
6517 035574 005737 000042      TST      42      ;CHECK IF IN CHAIN MODE
6518 035600 001407      BEQ      30$      ;NO HALT
6519 035602 012737 000001 033720 MOV      #1,SEOPCT ;FORCE END OF PASS COUNT TO ONE FOR ABORT
6520 035610 012706 001100      MOV      #STACK,SP ;INITIALIZE STACK
6521 035614 000137 033672      JMP      SEOP   ;BRING IN NEXT PROGRAM IN CHAIN
6522 035620 000000      30$:      HALT
6523 035622 000207      25$:      RTS      PC
6524
6525      .SBTTL  GENERATE HEADERS FOR SECTOR, TRACK, AND CYLINDER INCREMENTS
6526
6527 035624 013737 003300 003302 INCHDR: MOV      CYLN,HEADER ;LOAD HEADER WORD 1
6528 035632 005037 003304      CLR      HEADER+2 ;CALCULATE HEADER WORD 2
6529 035636 113737 003277 003305      MOV      TRACK,HEADER+3
6530 035644 006237 003304      ASR      HEADER+2
6531 035650 006237 003304      ASR      HEADER+2
6532 035654 006237 003304      ASR      HEADER+2
6533 035660 153737 003276 003304      BISB     SECTOR,HEADER+2
6534 035666 052737 140000 003304      BIS      #140000,HEADER+2
6535 035674 013746 003302      MOV      HEADER, -(SP) ;GENERATE XOR
6536 035700 013737 003304 003306      MOV      HEADER+2,HEADER+4
6537 035706 043737 003302 003306      BIC      HEADER,HEADER+4
6538 035714 043716 003304      BIC      HEADER+2, (SP)
6539 035720 052637 003306      BIS      (SP)+,HEADER+4
6540 035724 013737 003300 003220      MOV      CYLN,E.DCYL ;INCREMENT DISK ADDRESS
6541 035732 013737 003276 003206      MOV      SECTOR,E.DA
6542 035740 105237 003206      INCB     E.DA
6543 035744 122737 000026 003206      CMPB     #26,E.DA
6544 035752 001014      BNE      10$
6545 035754 105037 003206      CLRB     E.DA
6546 035760 105237 003207      INCB     E.DA+1
6547 035764 122737 000003 003207      CMPB     #3,E.DA+1
6548 035772 001004      BNE      10$
6549 035774 005037 003206      CLR      E.DA
6550 036000 005237 003220      INC      E.DCYL
6551 036004 000207      10$:      RTS      PC      ;RETURN
6552
6553      .SBTTL  CALCULATE EXPECTED HEADER
6554
6555 036006 005003      CALHDR:  CLR      R3      ;CLEAR EXPECTED FIRST WORD
6556 036010 013701 003220      MOV      E.DCYL,R1 ;STORE CYLINDER
6557 036014 001406      BEQ      5$      ;CHECK IF ZERO
6558 036016 012700 000020      MOV      #16,,R0 ;LOAD SHIFT COUNT

```

```

6559 036022 006101          1$: ROL R1          ;CALCULATE FIRST WORD
6560 036024 006003          ROR R3
6561 036026 005300          DEC R0
6562 036030 001374          BNE 1$
6563 036032 010337 003226  5$: MOV R3,E.MR2      ;LOAD FIRST WORD
6564 036036 013746 003206  E.DA,-(SP)      ;GET TRACK AND SECTOR
6565 036042 001410          BEQ 7$          ;CHECK IF TRACK = 0 AND SECTOR = 0
6566 036044 042716 000377  BIC #377,(SP)    ;CLEAR SECTOR BITS
6567 036050 001403          BEQ 6$          ;CHECK TRACK = 0
6568 036052 006216          ASR (SP)        ;SHIFT HEAD 3 BITS RIGHT
6569 036054 006216          ASR (SP)
6570 036056 006216          ASR (SP)
6571 036060 153716 003206  6$: BISB E.DA,(SP)    ;OR IN SECTOR BITS
6572 036064 012601          7$: MOV (SP)+,R1
6573 036066 032737 010000 003200  BIT #CFMT,E.CS1  ;CHECK FORMAT = 24 SECTOR
6574 036074 001402          BEQ 8$          ;NO, CALCULATE SECOND WORD
6575 036076 052701 001000  BIS #BIT9,R1      ;SET FORMAT BIT
6576 036102 005003          CLR R3          ;CLEAR EXPECTED SECOND WORD
6577 036104 005701          TST R1          ;CHECK IF WORD = 0
6578 036106 001406          BEQ 15$         ;YES, RETURN
6579 036110 012700 000020  MOV #16.,R0      ;LOAD SHIFT COUNT
6580 036114 006101          10$: ROL R1
6581 036116 006003          ROR R3
6582 036120 005300          DEC R0
6583 036122 001374          BNE 10$
6584 036124 010337 003230  15$: MOV R3,E.MR3    ;LOAD SECOND WORD
6585 036130 000207          RTS PC          ;RETURN
6586
6587 ;*****
6588 ;SBTTL GENERATE ECC WORD
6589
6590 036132 005737 003252  ECCGEN: TST P1.BIT    ;CHECK IF 1
6591 036136 001410          BEQ 3$          ;NO, CHECK IF BIT 31 = 1
6592 036140 032737 000001 003266  BIT #1,ECCHI  ;NO, CHECK IF BIT 31 = 1
6593 036146 001410          BEQ 5$          ;NO, SET ECC XORED BIT
6594 036150 005037 003272  1$: CLR ECCXOR    ;CLEAR ECC XORED BIT
6595 036154 000241          CLC          ;CLEAR CARRY FLOP
6596 036156 000410          BR 7$          ;SHIFT IN ECC BIT
6597
6598 036160 032737 000001 003266  3$: BIT #1,ECCHI  ;CHECK IF BIT 31 = 1
6599 036166 001770          BEQ 1$          ;NO, CLEAR ECC XORED BIT
6600 036170 012737 000001 003272  5$: MOV #1,ECCXOR  ;SET ECC XOR
6601 036176 000261          SEC          ;SET CARRY FLOP
6602 036200 006037 003270  7$: ROR ECCL0
6603 036204 006037 003266  ROR ECCHI
6604 036210 005737 003272  TST ECCXOR      ;CHECK IF XOR NEEDED
6605 036214 001422          BEQ 10$         ;NO, RETURN
6606 036216 012746 020020  MOV #BIT13:BIT4,-(SP) ;DO XOR OF ECC BITS
6607 036222 043716 003270  BIC ECCL0,(SP)    ; 0, 2, 11, 21, 23
6608 036226 042737 020020 003270  BIC #BIT13:BIT4,ECCL0
6609 036234 052637 003270  BIS (SP)+,ECCL0
6610 036240 012746 002400  MOV #BIT10:BIT8,-(SP)
6611 036244 043716 003266  BIC ECCHI,(SP)
6612 036250 042737 002400 003266  BIC #BIT10:BIT8,ECCHI
6613 036256 052637 003266  BIS (SP)+,ECCHI
6614 036262 013737 003266 003234 10$: MOV ECCHI,E.ECPT ;STORE ECC PATTERN

```

```

6615 036270 042737 174000 003234 BIC #174000,E.ECPT ;MASK UNSEEN BITS
6616 036276 000207 RTS PC ;RETURN
6617 ;*****
6618 ;SBTTL SIMULATE ONE BIT WRITE IN MAINTENCE MODE
6619
6620 036300 052737 042040 003224 WRTBIT: BIS #DMD!MEWD!WRTGAT,E.MR1 ;INITIALIZE
6621 036306 042737 114000 003224 BIC #RDGATE!PCA!PCD,E.MR1 ;EXPECTED MR1
6622 036314 005737 003254 TST PR.BIT ;CHECK IF ONE
6623 036320 001122 BNE 20$ ;YES, SIMULATE A ONE
6624 036322 005737 003256 TST M1.BIT ;CHECK IF PREVIOUS ONE
6625 036326 001023 BNE 10$ ;YES, NO TRANSITION
6626 036330 042737 002000 003224 BIC #MEWD,E.MR1 ;INDICATE TRANSITION
6627 036336 005737 003252 TST P1.BIT ;CHECK IF NEXT BIT = 1
6628 036342 001007 BNE 5$ ;YES, CHECK FOR PRECOMP ADVANCE
6629 036344 005737 003260 TST M2.BIT ;CHECK FOR PRECOMP. DELAY
6630 036350 001412 BEQ 10$ ;NO, CLOCK IN ZERO
6631 036352 052737 010000 003224 BIS #PCD,E.MR1 ;SET PRECOMP. DELAY
6632 036360 000406 BR 10$ ;CLOCK IN ZERO
6633
6634 036362 005737 003260 5$: TST M2.BIT ;CHECK FOR PRECOMP. ADVANCE
6635 036366 001003 BNE 10$ ;CLOCK IN ZERO
6636 036370 052737 004000 003224 BIS #PCA,E.MR1 ;SET PRECOMPENSATION ADVANCE
6637 036376 012762 000440 000026 10$: MOV #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6638 036404 012762 000040 000026 MOV #DMD,RKMR1(R2)
6639 036412 016237 000026 003164 MOV RKMR1(R2),T.MR1 ;STORE MR1
6640 036420 023737 003164 003224 CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6641 036426 001416 BEQ 15$ ;YES, CLOCK IN REST OF BIT
6642 036430 012737 036450 001202 MOV #13$, $ESCAPE ;LOAD ESCAPE
6643 036436 012737 052773 001312 MOV #EMW2,EMW+2 ;LOAD ERROR MESSAGE
6644 036444 011646 MOV (SP),-(SP) ;STORE RETURN
6645 036446 000207 RTS PC ;REPORT ERROR
6646 036450 032777 001000 142462 13$: BIT #SW9, $SWR ;CHECK IF LOOP ON ERROR
6647 036456 001402 BEQ 15$ ;NO, CONTINUE
6648 036460 000137 037012 JMP 63$ ;GO LOOP ON ERROR
6649
6650 036464 052737 002000 003224 15$: BIS #MEWD,E.MR1 ;RESET TRANSITION INDICATION
6651 036472 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6652 036500 012762 000040 000026 MOV #DMD,RKMR1(R2)
6653 036506 016237 000026 003164 MOV RKMR1(R2),T.MR1 ;STORE MR1
6654 036514 023737 003164 003224 CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6655 036522 001414 BEQ 18$ ;YES, RETURN
6656 036524 012737 036544 001202 MOV #17$, $ESCAPE ;LOAD ESCAPE
6657 036532 012737 053063 001312 MOV #EMW4,EMW+2 ;LOAD ERROR MESSAGE
6658 036540 011646 MOV (SP),-(SP) ;SAVE RETURN
6659 036542 000207 RTS PC ;REPORT ERROR
6660
6661 036544 032777 001000 142366 17$: BIT #SW9, $SWR ;CHECK IF LOOP ON ERROR
6662 036552 001117 BNE 63$
6663 036554 005037 001202 18$: CLR $ESCAPE ;CLEAR ESCAPE
6664 036560 062716 000002 ADD #2,(SP) ;ADJUST ESCAPE
6665 036564 000207 RTS PC ;RETURN
6666
6667
6668 036566 005737 003252 20$: TST P1.BIT ;CHECK IF NEXT BIT A ONE
6669 036572 001007 BNE 30$ ;YES, CHECK FOR PRECOMP. DELAY
6670 036574 005737 003256 TST M1.BIT ;CHECK FOR PRECOMP. ADVANCE

```

H10

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 124
SIMULATE ONE BIT WRITE IN MAINTENANCE MODE

SEQ 0124

```

6671 036600 001412          BEQ      40$      ;NO, CHECK MR1
6672 036602 052737 004000 003224  BIS      #PCA,E.MR1    ;SET PRECOMP ADVANCE
6673 036610 000406          BR       40$      ;CHECK MR1
6674
6675 036612 005737 003256      30$:  TST      M1.BIT      ;CHECK FOR PRECOMP. DELAY
6676 036616 001003          BNE      40$      ;NO, CHECK MR1
6677 036620 052737 010000 003224  BIS      #PCD,E.MR1    ;SET PRECOMP. DELAY
6678 036626 012762 000440 000026  40$:  MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6679 036634 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6680 036642 016237 000026 003164  MOV      RKMR1(R2),T.MR1 ;STORE MR1
6681 036650 023737 003164 003224  CMP      T.MR1,E.MR1    ;CHECK IF MR1 CORRECT
6682 036656 001414          BEQ      45$      ;YES, CLOCK IN REST OF BIT
6683 036660 012737 036700 001202  MOV      #42$, $ESCAPE ;LOAD ESCAPE
6684 036666 012737 052773 001312  MOV      #EMW2,EMW+2    ;LOAD ERROR MESSAGE
6685 036674 011646          MOV      (SP),-(SP) ;STORE RETURN
6686 036676 000207          RTS      PC         ;REPORT ERROR
6687
6688 036700 032777 001000 142232  42$:  BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
6689 036706 001041          BNE      63$      ;YES, LOOP ERROR
6690 036710 042737 002000 003224  45$:  BIC      #MEWD,E.MR1    ;INDICATE A TRANSITION
6691 036716 012762 000440 000026  MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK TRANSITION
6692 036724 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6693 036732 016237 000026 003164  MOV      RKMR1(R2),T.MR1 ;STORE MR1
6694 036740 023737 003164 003224  CMP      T.MR1,E.MR1    ;CHECK IF MR1 CORRECT
6695 036746 001414          BEQ      50$      ;YES, RETURN
6696 036750 012737 036770 001202  MOV      #47$, $ESCAPE ;LOAD ESCAPE
6697 036756 012737 053063 001312  MOV      #EMW4,EMW+2    ;LOAD ERROR MESSAGE
6698 036764 011646          MOV      (SP),-(SP) ;SAVE RETURN
6699 036766 000207          RTS      PC         ;REPORT ERROR
6700
6701 036770 032777 001000 142142  47$:  BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
6702 036776 001005          BNE      63$      ;YES, REPORT ERROR
6703 037000 005037 001202          50$:  CLR      $ESCAPE    ;LOAD ESCAPE
6704 037004 062716 000002          ADD      #2,(SP)      ;ADJUST RETURN
6705 037010 000207          RTS      PC         ;RETURN
6706
6707 037012 005037 001202          63$:  CLR      $ESCAPE    ;CLEAR ESCAPE
6708 037016 012706 001100          MOV      #STACK,SP    ;FORCE STACK
6709 037022 000177 142062          JMP      $SLPERR    ;JUMP TO LOOP ADDRESS
6710
6711 ;*****
6712 ;SBTTL SIMULATE ONE BIT OF READ DATA IN MAINTENANCE MODE
6713
6714 037026 105737 003254      RDBIT: TSTB     PR.BIT      ;CHECK IF ONE
6715 037032 001024          BNE      10$      ;YES, SIMULATE ONE
6716 037034 105737 003256      TSTB     M1.BIT      ;CHECK IF PREVIOUS ONE
6717 037040 001404          BEQ      4$       ;INSERT TRANSITION
6718 037042 012762 000440 000026  MOV      #DMD!MCLK,RKMR1(R2) ;DO NOT INSERT TRANSITION
6719 037050 000403          BR       5$       ;CLOCK IN ZERO
6720
6721 037052 012762 001440 000026  4$:  MOV      #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
6722 037060 012762 000040 000026  5$:  MOV      #DMD,RKMR1(R2)    ;CLOCK IN ZERO
6723 037066 012762 000440 000026  MOV      #DMD!MCLK,RKMR1(R2)
6724 037074 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6725 037102 000207          RTS      PC         ;RETURN
6726

```

```

6727 037104 012762 000440 000026 10$: MOV #DMD!MCLK,RKMR1(R2);CLOCK IN ONE
6728 037112 012762 000040 000026 MOV #DMD,RKMR1(R2)
6729 037120 012762 001440 000026 MOV #DMD!MCLK!MERD,RKMR1(R2)
6730 037126 012762 000040 000026 MOV #DMD,RKMR1(R2)
6731 037134 000207 RTS PC;RETURN
6732 .SBTTL APT COMMUNICATIONS ROUTINE
6733
6734 :*****
6735 037136 112737 000001 037402 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
6736 037144 112737 000001 037400 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
6737 037152 000403 BR $ATYC
6738 037154 112737 000001 037402 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
6739 037162 $ATYC:
6740 037162 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
6741 037164 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
6742 037166 105737 037400 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
6743 037172 001450 BEQ 5$ ;;IF NOT: BR
6744 037174 122737 000001 001234 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
6745 037202 001031 BNE 3$ ;;IF NOT: BR
6746 037204 132737 000100 001235 BITB #APTSPool,$ENVM ;;SHOULD SPOOL MESSAGES?
6747 037212 001425 BEQ 3$ ;;IF NOT: BR
6748 037214 017600 000004 MOV #4(SP),R0 ;;GET MESSAGE ADDR.
6749 037220 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
6750 037226 005737 001214 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
6751 037232 001375 BNE 1$ ;;IF NOT: WAIT
6752 037234 010037 001230 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
6753 037240 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
6754 037242 001376 BNE 2$
6755 037244 163700 001230 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
6756 037250 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
6757 037252 010037 001232 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
6758 037256 012737 000004 001214 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
6759 037264 000413 BR 5$
6760 037266 017637 000004 037312 3$: MOV #4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
6761 037274 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
6762 037302 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
6763 037306 004737 037604 JSR PC,$TYPE ;;CALL TYPE MACRO
6764 037312 000000 4$: .WORD 0
6765 037314 5$:
6766 037314 105737 037402 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
6767 037320 001416 BEQ 12$ ;;IF NOT: BR
6768 037322 005737 001234 TST $ENV ;;RUNNING UNDER APT?
6769 037326 001413 BEQ 12$ ;;IF NOT: BR
6770 037330 005737 001214 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
6771 037334 001375 BNE 11$ ;;IF NOT: WAIT
6772 037336 017637 000004 001216 MOV #4(SP),$FATAL ;;GET ERROR #
6773 037344 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
6774 037352 005237 001214 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
6775 037356 105037 037402 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
6776 037362 105037 037401 CLRB $LFLG ;;CLEAR LOG FLAG
6777 037366 105037 037400 CLRB $MFLG ;;CLEAR MESSAGE FLAG
6778 037372 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
6779 037374 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
6780 037376 000207 RTS PC;RETURN
6781 037400 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
6782 037401 000 $LFLG: .BYTE 0 ;;LOG FLAG

```


6783 037402 000
6784 037404
6785 000200
6786 000001
6787 000100
6788 00004C
6789
6790
6791
6792
6793
6794
6795
6796
6797
6798
6799
6800
6801
6802

\$FFLG: .BYTE 0 ;;FATAL FLAG
EVEN
APTSIZE=200
APTENV=001
APTSP00L=100
APTC SUP=040
.SBTTL ERROR HANDLER ROUTINE

;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
;AND GO TO TYPERR ON ERROR
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW15=1 HALT ON ERROR
;SW13=1 INHIBIT ERROR TYPEOUTS
;SW10=1 BELL ON ERROR
;SW09=1 LOOP ON ERROR
;CALL
;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

6803 037404
6804 037404 104407
6805 037406 105237 001103
6806 037412 001775
6807 037414 013777 001102 141520
6808 037422 032777 002000 141510
6809 037430 001402
6810 037432 104401 001204
6811 037436 005237 001112
6812 037442 011637 001116
6813 037446 162737 000002 001116
6814 037454 117737 141436 001114
6815 037462 032777 020000 141450
6816 037470 001004
6817 037472 004737 035326
6818 037476 104401 001211
6819 037502
6820 037502 122737 000001 001234
6821 037510 001007
6822 037512 113737 001114 037524
6823 037520 004737 037154
6824 037524 000
6825 037525 000
6826 037526 000777
6827 037530 005777 141404
6828 037534 100002
6829 037536 000000
6830 037540 104407
6831 037542 032777 001000 141370
6832 037550 001402
6833 037552 013716 001110
6834 037556 005737 001202
6835 037562 001402
6836 037564 013716 001202
6837 037570
6838 037570 022737 034062 000042

\$ERROR:
7\$: CKSWR
INCB
BEQ 7\$
MOV \$TSTNM, @DISPLAY
BIT #BIT10, @SWR
BEQ 1\$
TYPE
1\$: INC \$ERTTL
MOV (SP), \$ERRPC
SUB #2, \$ERRPC
MOVB @ \$ERRPC, \$ITEMB
BIT #BIT13, @SWR
BNE 20\$
JSR PC, TYPERR
TYPE
20\$: CMPB #APTENV, \$ENV
BNE 2\$
MOVB \$ITEMB, 21\$
JSR PC, SATY4
21\$: .BYTE 0
.BYTE 0
22\$: BR 22\$
TST @SWR
2\$: BPL 3\$
HALT
CKSWR
3\$: BIT #BIT09, @SWR
BEQ 4\$
MOV \$LPERR, (SP)
4\$: TST \$ESCAPE
BEQ 5\$
MOV \$ESCAPE, (SP)
5\$: CMP # \$ENDAD, @#42
;;TEST FOR CHANGE IN SOFT-SWR
;;SET THE ERROR FLAG
;;DON'T LET THE FLAG GO TO ZERO
;;DISPLAY TEST NUMBER AND ERROR FLAG
;;BELL ON ERROR?
;;NO - SKIP
;;RING BELL
;;COUNT THE NUMBER OF ERRORS
;;GET ADDRESS OF ERROR INSTRUCTION
;;STRIP AND SAVE THE ERROR ITEM CODE
;;SKIP TYPEOUT IF SET
;;SKIP TYPEOUTS
;;GO TO USER ERROR ROUTINE
;;RUNNING IN APT MODE
;;NO SKIP APT ERROR REPORT
;;SET ITEM NUMBER AS ERROR NUMBER
;;REPORT FATAL ERROR TO APT
;;APT ERROR LOOP
;;HALT ON ERROR
;;SKIP IF CONTINUE
;;HALT ON ERROR!
;;TEST FOR CHANGE IN SOFT-SWR
;;LOOP ON ERROR SWITCH SET?
;;BR IF NO
;;FUDGE RETURN FOR LOOPING
;;CHECK FOR AN ESCAPE ADDRESS
;;BR IF NONE
;;FUDGE RETURN ADDRESS FOR ESCAPE
;;ACT-11 AUTO-ACCEPT"

6839 037576 001001
6840 037600 000000
6841 037602
6842 037602 000002
6843
6844
6845
6846
6847
6848
6849
6850
6851
6852
6853
6854
6855
6856
6857
6858
6859
6860

6861 037604 105737 001157
6862 037610 100002
6863 037612 000000
6864 037614 000430
6865 037616 010046
6866 037620 017600 000002
6867 037624 122737 000001 001234
6868 037632 001011
6869 037634 132737 000100 001235
6870 037642 001405
6871 037644 010037 037654
6872 037650 004737 037144
6873 037654 000000
6874 037656 132737 000040 001235
6875 037664 001003
6876 037666 112046
6877 037670 001005
6878 037672 005726
6879 037674 012600
6880 037676 062716 000002
6881 037702 000002
6882 037704 122716 000011
6883 037710 001430
6884 037712 122716 000200
6885 037716 001006
6886 037720 005726
6887 037722 104401
6888 037724 001211
6889 037726 105037 040062
6890 037732 000755
6891 037734 004737 040016
6892 037740 123726 001155
6893 037744 001350
6894 037746 013746 001154

```

        BNE      6$          ;;BRANCH IF NO
        HALT
6$:      RTI                ;;RETURN

.SBTTL  TYPE ROUTINE

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
*      TYPE
*      MESADR
*
$TYPE:   TSTB      $TPFLG          ;;IS THERE A TERMINAL?
        BPL      1$              ;;BR IF YES
        HALT      HERE IF NO TERMINAL
        BR      3$              LEAVE
1$:      MOV      RO, -(SP)        SAVE RO
        MOV      22(SP), RO      GET ADDRESS OF ASCIZ STRING
        CMPB     #APTENV, $ENV    RUNNING IN APT MODE
        BNE     62$              NO, GO CHECK FOR APT CONSOLE
        BITB     #APTPOOL, $ENVM  SPOOL MESSAGE TO APT
        BEQ      62$              NO, GO CHECK FOR CONSOLE
        MOV      RO, 61$          SETUP MESSAGE ADDRESS FOR APT
        JSR      PC, $ATY3        SPOOL MESSAGE TO APT
        .WORD    0               MESSAGE ADDRESS
62$:     BITB     #APTCSUP, $ENVM  APT CONSOLE SUPPRESSED
        BNE     60$              YES, SKIP TYPE OUT
        MOV      (RO)+, -(SP)    PUSH CHARACTER TO BE TYPED ONTO STACK
        BNE     4$              BR IF IT ISN'T THE TERMINATOR
        TST      (SP)+           IF TERMINATOR POP IT OFF THE STACK
        MOV      (SP)+, RO       RESTORE RO
60$:     ADD      #2, (SP)        ADJUST RETURN PC
3$:      RTI                    RETURN
4$:      CMPB     #HT, (SP)       BRANCH IF <HT>
        BEQ      8$              ;;BRANCH IF NOT <CRLF>
        CMPB     #CRLF, (SP)
        BNE     5$              ;;POP <CR><LF> EQUIV
        TST      (SP)+           ;;TYPE A CR AND LF
        TYPE
        $CRLF
        CLRB
        BR      2$              CLEAR CHARACTER COUNT
5$:      JSR      PC, $TYPEC      GET NEXT CHARACTER
6$:      CMPB     $FILLC, (SP)+  GO TYPE THIS CHARACTER
        BNE     2$              IS IT TIME FOR FILLER CHARS.?
        MOV      $NULL, -(SP)    IF NO GO GET NEXT CHAR.
        ;;GET # OF FILLER CHARS. NEEDED

```



```
6895
6896 037752 105366 000001      7$:   DECB    1(SP)      ;; AND THE NULL CHAR.
6897 037756 002770              BLT     6$          ;; DOES A NULL NEED TO BE TYPED?
6898 037760 004737 040016      JSR     PC,$TYPEC    ;; BR IF NO--GO POP THE NULL OFF OF STACK
6899 037764 105337 040062      DECB    $CHARCNT    ;; GO TYPE A NULL
6900 037770 000770              BR      7$          ;; DO NOT COUNT AS A COUNT
6901                                ;; LOOP
6902
6903      ;HORIZONTAL TAB PROCESSOR
6904 037772 112716 000040      8$:   MOVB    #' (SP)      ;; REPLACE TAB WITH SPACE
6905 037776 004737 040016      9$:   JSR     PC,$TYPEC    ;; TYPE A SPACE
6906 040002 132737 000007 040062  BITB    #7,$CHARCNT    ;; BRANCH IF NOT AT
6907 040010 001372              BNE     9$          ;; TAB STOP
6908 040012 005726              TST     (SP)+        ;; POP SPACE OFF STACK
6909 040014 000724              BR      2$          ;; GET NEXT CHARACTER
6910 040016 105777 141126      $TYPEC: TSTB    @STPS      ;; WAIT UNTIL PRINTER IS READY
6911 040022 100375              BPL     $TYPEC
6912 040024 116677 000002 141120  MOVB    2(SP),@STPB    ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6913 040032 122766 000015 000002  CMPB    #CR,2(SP)    ;; IS CHARACTER A CARRIAGE RETURN?
6914 040040 001003              BNE     1$          ;; BRANCH IF NO
6915 040042 105037 040062      CLRB    $CHARCNT    ;; YES--CLEAR CHARACTER COUNT
6916 040046 000406              BR      $TYPEX
6917 040050 122766 000012 000002 1$:   CMPB    #LF,2(SP)    ;; IS CHARACTER A LINE FEED?
6918 040056 001402              BEQ     $TYPEX      ;; BRANCH IF YES
6919 040060 105227              INCB    (PC)+        ;; COUNT THE CHARACTER
6920 040062 000000      $CHARCNT: .WORD 0          ;; CHARACTER COUNT STORAGE
6921 040064 000207      $TYPEX: RTS     PC
6922
6923      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6924
6925      ;*****
6926      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6927      ;OCTAL (ASCII) NUMBER AND TYPE IT.
6928      ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6929      ;CALL:
6930      ;   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
6931      ;   TYPOS      ;; CALL FOR TYPEOUT
6932      ;   .BYTE    N          ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6933      ;   .BYTE    M          ;; M=1 OR 0
6934      ;   ;; 1=TYPE LEADING ZEROS
6935      ;   ;; 0=SUPPRESS LEADING ZEROS
6936      ;
6937      ;$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
6938      ;$TYPOS OR $TYPOC
6939      ;CALL:
6940      ;   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
6941      ;   TYPON      ;; CALL FOR TYPEOUT
6942      ;
6943      ;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6944      ;CALL:
6945      ;   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
6946      ;   TYPOC      ;; CALL FOR TYPEOUT
6947      ;
6948 040066 017646 000000      $TYPOS: MOV     @2(SP),-(SP)    ;; PICKUP THE MODE
6949 040072 116637 000001 040311  MOVB    1(SP),$OFILL    ;; LOAD ZERO FILL SWITCH
6950 040100 112637 040313      MOVB    (SP)+,$OMODE+1    ;; NUMBER OF DIGITS TO TYPE
```

```

6951 040104 062716 000002 ADD #2,(SP) ;ADJUST RETURN ADDRESS
6952 040110 000406 BR $TYPON
6953 040112 112737 000001 040311 $TYPON: MOV #1,$OFILL ;SET THE ZERO FILL SWITCH
6954 040120 112737 000006 040313 MOV #6,$OMODE+1 ;SET FOR SIX(6) DIGITS
6955 040126 112737 000005 040310 $TYPON: MOV #5,$OCNT ;SET THE ITERATION COUNT
6956 040134 010346 MOV R3,-(SP) ;SAVE R3
6957 040136 010446 MOV R4,-(SP) ;SAVE R4
6958 040140 010546 MOV R5,-(SP) ;SAVE R5
6959 040142 113704 040313 MOV $OMODE+1,R4 ;GET THE NUMBER OF DIGITS TO TYPE
6960 040146 005404 NEG R4
6961 040150 062704 000006 ADD #6,R4 ;SUBTRACT IT FOR MAX. ALLOWED
6962 040154 110437 040312 MOV R4,$OMODE ;SAVE IT FOR USE
6963 040160 113704 040311 MOV $OFILL,R4 ;GET THE ZERO FILL SWITCH
6964 040164 016605 000012 MOV 12(SP),R5 ;PICKUP THE INPUT NUMBER
6965 040170 005003 CLR R3 ;CLEAR THE OUTPUT WORD
6966 040172 006105 1$: ROL R5 ;ROTATE MSB INTO "C"
6967 040174 000404 BR 3$ ;GO DO MSB
6968 040176 006105 2$: ROL R5 ;FORM THIS DIGIT
6969 040200 006105 ROL R5
6970 040202 006105 ROL R5
6971 040204 010503 MOV R5,R3
6972 040206 006103 3$: ROL R3 ;GET LSB OF THIS DIGIT
6973 040210 105337 040312 DECB $OMODE ;TYPE THIS DIGIT?
6974 040214 100016 BPL 7$ ;BR IF NO
6975 040216 042703 177770 BIC #177770,R3 ;GET RID OF JUNK
6976 040222 001002 BNE 4$ ;TEST FOR 0
6977 040224 005704 TST R4 ;SUPPRESS THIS 0?
6978 040226 001403 BEQ 5$ ;BR IF YES
6979 040230 005204 4$: INC R4 ;DON'T SUPPRESS ANYMORE 0'S
6980 040232 052703 000060 BIS #'0,R3 ;MAKE THIS DIGIT ASCII
6981 040236 052703 000040 5$: BIS #'1,R3 ;MAKE ASCII IF NOT ALREADY
6982 040242 110337 040306 MOV R3,$$ ;SAVE FOR TYPING
6983 040246 104401 040306 TYPE 8$ ;GO TYPE THIS DIGIT
6984 040252 105337 040310 7$: DECB $OCNT ;COUNT BY 1
6985 040256 003347 BGT 2$ ;BR IF MORE TO DO
6986 040260 002402 BLT 6$ ;BR IF DONE
6987 040262 005204 INC R4 ;INSURE LAST DIGIT ISN'T A BLANK
6988 040264 000744 BR 2$ ;GO DO THE LAST DIGIT
6989 040266 012605 6$: MOV (SP)+,R5 ;RESTORE R5
6990 040270 012604 MOV (SP)+,R4 ;RESTORE R4
6991 040272 012603 MOV (SP)+,R3 ;RESTORE R3
6992 040274 016666 000002 000004 MOV 2(SP),4(SP) ;SET THE STACK FOR RETURNING
6993 040302 012616 MOV (SP)+,(SP)
6994 040304 000002 RTI ;RETURN
6995 040306 000 8$: .BYTE 0 ;STORAGE FOR ASCII DIGIT
6996 040307 000 .BYTE 0 ;TERMINATOR FOR TYPE ROUTINE
6997 040310 000 $OCNT: .BYTE 0 ;OCTAL DIGIT COUNTER
6998 040311 000 $OFILL: .BYTE 0 ;ZERO FILL SWITCH
6999 040312 000000 $OMODE: .WORD 0 ;NUMBER OF DIGITS TO TYPE
7000 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7001
7002 ;*****
7003 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7004 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7005 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7006 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE

```

N10

```
7007 ;*REPLACED WITH SPACES.
7008 ;*CALL:
7009 ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
7010 ;*      TYPDS      ;;GO TO THE ROUTINE
7011
7012 $TYPDS:
7013      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
7014      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7015      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7016      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7017      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7018      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
7019      MOV      20(SP),R5     ;;GET THE INPUT NUMBER
7020      BPL      1$           ;;BR IF INPUT IS POS.
7021      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
7022      MOVB     #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
7023      CLR      R0           ;;ZERO THE CONSTANTS INDEX
7024      MOV      #SDBLK,R3    ;;SETUP THE OUTPUT POINTER
7025      MOVB     #'',(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
7026      CLR      R2           ;;CLEAR THE BCD NUMBER
7027      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
7028      SUB      R1,R5        ;;FORM THIS BCD DIGIT
7029      BLT      4$           ;;BR IF DONE
7030      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
7031      BR      3$
7032      ADD      R1,R5        ;;ADD BACK THE CONSTANT
7033      TST      R2           ;;CHECK IF BCD DIGIT=0
7034      BNE      5$           ;;FALL THROUGH IF 0
7035      TSTB     (SP)         ;;STILL DOING LEADING 0'S?
7036      BMI      7$           ;;BR IF YES
7037      ASLB     (SP)         ;;MSD?
7038      BCC      6$           ;;BR IF NO
7039      MOVB     1(SP),-1(R3)  ;;YES--SET THE SIGN
7040      BIS      #'0,R2       ;;MAKE THE BCD DIGIT ASCII
7041      BIS      #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7042      MOVB     R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7043      TST      (R0)+        ;;JUST INCREMENTING
7044      CMP      R0,#10       ;;CHECK THE TABLE INDEX
7045      BLT      2$           ;;GO DO THE NEXT DIGIT
7046      BGT      8$           ;;GO TO EXIT
7047      MOV      R5,R2        ;;GET THE LSD
7048      BR      6$           ;;GO CHANGE TO ASCII
7049      TSTB     (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
7050      BPL      9$           ;;BR IF NO
7051      MOVB     -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
7052      CLRB     (R3)         ;;SET THE TERMINATOR
7053      MOV      (SP)+,R5     ;;POP STACK INTO R5
7054      MOV      (SP)+,R3     ;;POP STACK INTO R3
7055      MOV      (SP)+,R2     ;;POP STACK INTO R2
7056      MOV      (SP)+,R1     ;;POP STACK INTO R1
7057      MOV      (SP)+,R0     ;;POP STACK INTO R0
7058      TYPE     $SDBLK      ;;NOW TYPE THE NUMBER
7059      MOV      2(SP),4(SP)  ;;ADJUST THE STACK
7060      MOV      (SP)+,(SP)
7061      RTI
7062 $DTBL: 10000.
```

7063	040522	001750			1000.	
7064	040524	000144			100.	
7065	040526	000012			10.	
7066	040530	000004			\$DBLK: .BLKW 4	
7067					.SBTTL TTY INPUT ROUTINE	
7068					*****	
7069					ENABL LSB	
7070					\$TKCNT: .WORD 0	;; NUMBER OF ITEMS IN QUEUE
7071	040540	000000			\$TKQIN: .WORD 0	;; INPUT POINTER
7072	040542	000000			\$TKQOUT: .WORD 0	;; OUTPUT POINTER
7073	040544	000000			\$TKQSRT: .BLKB 1	;; TTY KEYBOARD QUEUE
7074	040546	000001			\$TKQEND=.	
7075		040547			.EVEN	
7076		040550				
7077						
7078					*TK INITIALIZE ROUTINE	
7079					*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE	
7080					*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT	
7081						
7082					*CALL:	
7083					* JSR PC,\$TKINT	
7084					* RETURN	
7085						
7086	040550	005037	040540		\$TKINT: CLR \$TKCNT	;; CLEAR COUNT OF ITEMS IN QUEUE
7087	040554	012737	040546	040542	MOV \$TKQSRT,\$TKQIN	;; MOVE THE STARTING ADDRESS OF THE
7088	040562	013737	040542	040544	MOV \$TKQIN,\$TKQOUT	;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
7089	040570	012737	040620	000060	MOV \$TKSRV,\$TKVEC	;; INITIALIZE THE KEYBOARD VECTOR
7090	040576	012737	000200	000062	MOV #200,\$TKVEC+2	;; "BR" LEVEL 4
7091	040604	005777	140336		TST \$TKB	;; CLEAR DONE FLAG
7092	040610	012777	000100	140326	MOV #100,\$TKS	;; ENABLE TTY KEYBOARD INTERRUPT
7093	040616	000207			RTS PC	;; RETURN TO CALLER
7094						
7095					*TK SERVICE ROUTINE	
7096					*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT	
7097					*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING	
7098					*IT IN THE QUEUE.	
7099					*IF THE CHARACTER IS A "CONTROL-C" (↑C) \$TKINT IS CALLED AND	
7100					*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)	
7101						
7102	040620	117746	140322		\$TKSRV: MOVB \$TKB, -(SP)	;; PICKUP THE CHARACTER
7103	040624	042716	177600		BIC #↑C177, (SP)	;; STRIP THE JUNK
7104	040630	021627	000003		CMP (SP), #3	;; IS IT A CONTROL C?
7105	040634	001007			BNE 1\$	;; BRANCH IF NO
7106	040636	104401	041734		TYPE \$CNTLC	;; TYPE A CONTROL-C (↑C)
7107	040642	004737	040550		JSR PC,\$TKINT	;; INIT THE KEYBOARD
7108	040646	005726			TST (SP)+	;; CLEAN UP STACK
7109	040650	000137	034570		JMP CTRHLT	;; CONTROL C RESTART
7110	040654	021627	000007		1\$: CMP (SP), #7	;; IS IT A CONTROL G?
7111	040660	001004			BNE 2\$	;; BRANCH IF NO
7112	040662	022737	000176	001140	CMP \$SWREG, SWR	;; IS SOFT-SWR SELECTED?
7113	040670	001500			BEQ 6\$	;; GO TO SWR CHANGE
7114						
7115	040672				2\$: CMP #1,\$TKCNT	;; IS THE QUEUE FULL?
7116	040672	022737	000001	040540	BNE 3\$	;; BRANCH IF NO
7117	040700	001004			TYPE \$BELL	;; RING THE TTY BELL
7118	040702	104401	001204			

```

7119 040706 005726          TST      (SP)+      ;; CLEAN CHARACTER OFF OF STACK
7120 040710 000451          BR        5$      ;; EXIT
7121 040712 021627 000023  3$:  CMP      (SP),#22  ;; IS IT A CONTROL-S?
7122 040716 001021          BNE      32$      ;; BRANCH IF NO
7123 040720 005077 140220          CLR      @STKS  ;; DISABLE TTY KEYBOARD INTERRUPTS
7124 040724 005726          TST      (SP)+      ;; CLEAN CHAR OFF STACK
7125 040726 105777 140212  31$:  TSTB     @STKS  ;; WAIT FOR A CHAR
7126 040732 100375          BPL      31$      ;; LOOP UNTIL ITS THERE
7127 040734 117746 140206          MOVB     @STKB,-(SP)  ;; GET THE CHARACTER
7128 040740 042716 177600          BIC      #1C177,(SP)  ;; MAKE IT 7-BIT ASCII
7129 040744 022627 000021          CMP      (SP)+,#21  ;; IS IT A CONTROL-Q?
7130 040750 001366          BNE      31$      ;; BRANCH IF NO
7131 040752 012777 000100 140164  MOV      #100,@STKS  ;; REENABLE TTY KEYBOARD INTERRUPTS
7132 040760 000002          RTI             ;; RETURN
7133 040762 005237 040540  32$:  INC      STKCNT  ;; COUNT THIS CHARACTER
7134 040766 021627 000140          CMP      (SP),#140  ;; IS IT UPPER CASE?
7135 040772 002405          BLT      4$      ;; BRANCH IF YES
7136 040774 021627 000175          CMP      (SP),#175  ;; IS IT A SPECIAL CHAR?
7137 041000 003002          BGT      4$      ;; BRANCH IF YES
7138 041002 042716 000040          BIC      #40,(SP)  ;; MAKE IT UPPER CASE
7139 041006 112677 177530  4$:  MOVB     (SP)+,@STKQIN  ;; AND PUT IT IN QUEUE
7140 041012 005237 040542          INC      STKQIN  ;; UPDATE THE POINTER
7141 041016 023727 040542 040547  CMP      STKQIN,#STKQEND  ;; GO OFF THE END?
7142 041024 001003          BNE      5$      ;; BRANCH IF NO
7143 041026 012737 040546 040542  MOV      #STKQSRST,STKQIN  ;; RESET THE POINTER
7144 041034 000002  5$:  RTI             ;; RETURN
7145
7146  ;; *****
7147  ;; *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7148  ;; *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7149  ;; *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
7150  ;; *CALL WHEN OPERATING IN TTY INTERRUPT MODE.
7151 041036 022737 000176 001140  $CKSWR:  CMP      #SWREG,SWR  ;; IS THE SOFT-SWR SELECTED
7152 041044 001124          BNE      15$      ;; EXIT IF NOT
7153 041046 105777 140072          TSTB     @STKS  ;; IS A CHAR WAITING?
7154 041052 100121          BPL      15$      ;; IF NOT, EXIT
7155 041054 117746 140066          MOVB     @STKB,-(SP)  ;; YES
7156 041060 042716 177600          BIC      #1C177,(SP)  ;; MAKE IT 7-BIT ASCII
7157 041064 021627 000007          CMP      (SP),#7  ;; IS IT A CONTROL-G?
7158 041070 001300          BNE      2$      ;; IF NOT, PUT IT IN THE TTY QUEUE
7159
7160  ;; *****
7161  ;; *CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7162  ;; *ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7163  ;; *CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
7164 041072 123727 001134 000001  6$:  CMPB     $AUTOB,#1  ;; ARE WE RUNNING IN AUTO-MODE?
7165 041100 001674          BEQ      2$      ;; BRANCH IF YES
7166 041102 005726          TST      (SP)+      ;; CLEAR CONTROL-G OFF STACK
7167 041104 004737 040550          JSR      PC,STKINT  ;; FLUSH THE TTY INPUT QUEUE
7168 041110 005077 140030          CLR      @STKS  ;; DISABLE TTY KEYBOARD INTERRUPTS
7169 041114 112737 000001 001135  MOVB     #1,$INTAG  ;; SET INTERRUPT MODE INDICATOR
7170
7171 041122 104401 041746          $GTSWR:  TYPE     ,SCNTLG  ;; ECHO THE CONTROL-G (!G)
7172 041126 104401 041753          TYPE     $MSWR  ;; TYPE CURRENT CONTENTS
7173 041132 013746 000176          MOV      SWREG,-(SP)  ;; SAVE SWREG FOR TYPEOUT

```

7175	041136	104402			TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
7176	041140	104401	041764		TYPE	, \$MNEW	::PROMPT FOR NEW SWR
7177	041144	005046		19\$:	CLR	-(SP)	::CLEAR COUNTER
7178	041146	005046			CLR	-(SP)	::THE NEW SWR
7179	041150	105777	137770	7\$:	TSTB	2\$TKS	::CHAR THERE?
7180	041154	100375			BPL	7\$	::IF NOT TRY AGAIN
7181							
7182	041156	117746	137764		MOVB	2\$TKB, -(SP)	::PICK UP CHAR
7183	041162	042716	177600		BIC	#1C177, (SP)	::MAKE IT 7-BIT ASCII
7184							
7185	041166	021627	000003		CMP	(SP), #3	::IS IT A CONTROL-C?
7186	041172	001015			BNE	9\$	::BRANCH IF NOT
7187	041174	104401	041734		TYPE	, \$CNTLC	::YES, ECHO CONTROL-C (↑C)
7188	041200	062706	000006		ADD	#6, SP	::CLEAN UP STACK
7189	041204	123727	001135	000001	CMPB	\$INTAG, #1	::REENABLE TTY KEYBOARD INTERRUPTS?
7190	041212	001003			BNE	8\$	::BRANCH IF NO
7191	041214	012777	000100	137722	MOV	#100, 2\$TKS	::ALLOW TTY KEYBOARD INTERRUPTS
7192	041222	000137	034570	8\$:	JMP	CIRHLT	::CONTROL-C RESTART
7193							
7194							
7195	041226	021627	000025		CMP	(SP), #25	::IS IT A CONTROL-U?
7196	041232	001005			BNE	10\$	::BRANCH IF NOT
7197	041234	104401	041741		TYPE	, \$CNTLU	::YES, ECHO CONTROL-U (↑U)
7198	041240	062706	000006	20\$:	ADD	#6, SP	::IGNORE PREVIOUS INPUT
7199	041244	000737			BR	19\$	::LET'S TRY IT AGAIN
7200							
7201							
7202	041246	021627	000015	10\$:	CMP	(SP), #15	::IS IT A <CR>?
7203	041252	001022			BNE	16\$	::BRANCH IF NO
7204	041254	005766	000004		TST	4(SP)	::YES, IS IT THE FIRST CHAR?
7205	041260	001403			BEQ	11\$	::BRANCH IF YES
7206	041262	016677	000002	137650	MOV	2(SP), 2\$SWR	::SAVE NEW SWR
7207	041270	062706	000006	11\$:	ADD	#6, SP	::CLEAN UP STACK
7208	041274	104401	001211	14\$:	TYPE	, \$CRLF	::ECHO <CR> AND <LF>
7209	041300	123727	001135	000001	CMPB	\$INTAG, #1	::RE-ENABLE TTY KBD INTERRUPTS?
7210	041306	001003			BNE	15\$	::BRANCH IF NOT
7211	041310	012777	000100	137626	MOV	#100, 2\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
7212	041316	000002		15\$:	RTI		::RETURN
7213	041320	004737	040016	16\$:	JSR	PC, \$TYPEC	::ECHO CHAR
7214	041324	021627	000060		CMP	(SP), #60	::CHAR < 0?
7215	041330	002420			BLT	18\$	::BRANCH IF YES
7216	041332	021627	000067		CMP	(SP), #67	::CHAR > 7?
7217	041336	003015			BGT	18\$	::BRANCH IF YES
7218	041340	042726	000060		BIC	#60, (SP)+	::STRIP-OFF ASCII
7219	041344	005766	000002		TST	2(SP)	::IS THIS THE FIRST CHAR
7220	041350	001403			BEQ	17\$	::BRANCH IF YES
7221	041352	006316			ASL	(SP)	::NO, SHIFT PRESENT
7222	041354	006316			ASL	(SP)	::CHAR OVER TO MAKE
7223	041356	006316			ASL	(SP)	::ROOM FOR NEW ONE.
7224	041360	005266	000002	17\$:	INC	2(SP)	::KEEP COUNT OF CHAR
7225	041364	056616	177776		BIS	-2(SP), (SP)	::SET IN NEW CHAR
7226	041370	000667			BR	7\$	::GET THE NEXT ONE
7227	041372	104401	001210	18\$:	TYPE	, \$QUES	::TYPE ?<CR><LF>
7228	041376	000720			BR	20\$	::SIMULATE CONTROL-U
7229							
7230					.DSABL	LSB	


```

7231
7232
7233
7234
7235
7236
7237
7238
7239
7240 041400 011646
7241 041402 016666 000004 000002
7242 041410 005066 000004
7243 041414 005046
7244 041416 012746 041424
7245 041422 000002
7246 041424
7247 041424 005737 040540
7248 041430 001775
7249 041432 005337 040540
7250 041436 117766 177102 000004
7251 041444 005237 040544
7252 041450 023727 040544 040547
7253 041456 001003
7254 041460 012737 040546 040544
7255 041466 000002
7256
7257
7258
7259
7260
7261
7262
7263 041470 010346
7264 041472 005046
7265 041474 012703 041724
7266 041500 022703 041734
7267 041504 101456
7268 041506 104410
7269 041510 112613
7270 041512 122713 000177
7271 041516 001022
7272 041520 005716
7273 041522 001007
7274 041524 112737 000134 041722
7275 041532 104401 041722
7276 041536 012716 177777
7277 041542 005303
7278 041544 020327 041724
7279 041550 103434
7280 041552 111337 041722
7281 041556 104401 041722
7282 041562 000746
7283 041564 005716
7284 041566 001406
7285 041570 112737 000134 041722
7286 041576 104401 041722

*****
;THIS ROUTINE WILL INPLT A SINGLE CHARACTER FROM THE TTY
;CALL:
;      RDCHR
;      RETURN HEPE
;      ;GET A CHARACTER FROM THE QUEUE
;      ;CHARACTER IS ON THE STACK
;      ;WITH PARITY BIT STRIPPED OFF
;
SRDCHR: MOV      (SP), -(SP)      ;PUSH DOWN THE PC AND
        MOV      4(SP), 2(SP)    ;THE PS
        CLR      4(SP)           ;GET READY FOR A CHARACTER
        CLR      -(SP)           ;PUT NEW PS ON STACK
        MOV      #64$, -(SP)     ;PUT NEW PC ON STACK
        RTI                    ;POP NEW PC AND PS
64$:
1$:      TST      STKCNT          ;WAIT ON A CHARACTER
        BEQ      1$
        DEC      STKCNT          ;DECREMENT THE COUNTER
        MOVB     @STKQOUT, 4(SP) ;GET ONE CHARACTER
        INC      STKQOUT         ;UPDATE THE POINTER
        CMP      STKQOUT, #STKQEND ;DID IT GO OFF OF THE END?
        BNE     2$              ;BRANCH IF NO
        MOV      #STKQSRRT, STKQOUT ;RESET THE POINTER
        RTI                    ;RETURN
2$:
*****
;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
;CALL:
;      RDLIN
;      RETURN HERE
;      ;INPUT A STRING FROM THE TTY
;      ;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
;      ;TERMINATOR WILL BE A BYTE OF ALL 0'S
;
SRDLIN: MOV      R3, (SP)        ;SAVE R3
        CLR      -(SP)          ;CLEAR THE RUBOUT KEY
        MOV      #STTYIN, R3    ;GET ADDRESS
        CMP      #STTYIN+8, R3  ;BUFFER FULL?
        BLOS     4$             ;BR IF YES
        RDCHR    (SP)+, (R3)    ;GO READ ONE CHARACTER FROM THE TTY
        MOVB     #177, (R3)     ;GET CHARACTER
        CMPB     5$             ;IS IT A RUBOUT
        BNE     5$             ;BR IF NO
        TST      (SP)           ;IS THIS THE FIRST RUBOUT?
        BNE     6$             ;BR IF NO
        MOVB     #' \, 9$       ;TYPE A BACK SLASH
        TYPE     9$
        MOV      #-1, (SP)      ;SET THE RUBOUT KEY
        DEC      R3             ;BACKUP BY ONE
        CMP      R3, #STTYIN    ;STACK EMPTY?
        BLOS     4$             ;BR IF YES
        MOVB     (R3), 9$       ;SETUP TO TYPEOUT THE DELETED CHAR.
        TYPE     9$
        BR       2$            ;GO TYPE
        TST      (SP)           ;GO READ ANOTHER CHAR.
        BEQ      7$             ;RUBOUT KEY SET?
        MOVB     #' \, 9$       ;BR IF NO
        TYPE     9$             ;TYPE A BACK SLASH

```

```

7287 041602 005016
7288 041604 122713 000025
7289 041610 001003
7290 041612 104401 041741
7291 041616 000726
7292 041620 122713 000022
7293 041624 001011
7294 041626 105013
7295 041630 104401 001211
7296 041634 104401 041724
7297 041640 000717
7298 041642 104401 001210
7299 041646 000712
7300 041650 111337 041722
7301 041654 104401 041722
7302 041660 122723 000015
7303 041664 001305
7304 041666 105063 177777
7305 041672 104401 001212
7306 041676 005726
7307 041700 012603
7308 041702 011646
7309 041704 016666 000004 000002
7310 041712 012766 041724 000004
7311 041720 000002
7312 041722 000
7313 041723 000
7314 041724 000010
7315 041734 041536 005015 000
7316 041741 136 006525 000012
7317 041746 043536 005015 000
7318 041753 015 051412 051127
7319 041760 036440 000040
7320 041764 020040 042516 020127
7321 041772 020075 000
7322 041776
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337 041776 011646
7338 042000 016666 000004 000002
7339 042006 010046
7340 042010 010146
7341 042012 010246
7342 042014 104411

CLR (SP)
7$: CMPB #25,(R3)
BNE 8$
TYPE $CNTLU
BR 1$
8$: CMPB #22,(R3)
BNE 3$
CLRB (R3)
TYPE $CRLF
TYPE $TTYIN
BR 2$
4$: TYPE $QUES
BR 1$
3$: MOVB (R3),9$
TYPE 9$
CMPB #15,(R3)+
BNE 2$
CLRB -1(R3)
TYPE $LF
TST (SP)+
MOV (SP)+,R3
MOV (SP)-,(SP)
MOV 4(SP),2(SP)
MOV $TTYIN,4(SP)
RTI
9$: .BYTE 0
.BYTE 0
$TTYIN: .BLKB 8.
$CNTLC: .ASCIZ /TC/15/12/
$CNTLU: .ASCIZ /TU/15/12/
$CNTLG: .ASCIZ /TG/15/12/
$MSWR: .ASCIZ <15><12>/SWR = /
$MNEW: .ASCIZ / NEW = /

.EVEN
.SBttl READ AN OCTAL NUMBER FROM THE TTY

*****
*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY.
*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
*CALL:
*
* RDOCT
* RETURN HERE
*
;; READ AN OCTAL NUMBER
;; LOW ORDER BITS ARE ON TOP OF THE STACK
;; HIGH ORDER BITS ARE IN $HIOCT

$RDOCT: MOV (SP)-,(SP)
MOV 4(SP),2(SP)
MOV R0,-(SP)
MOV R1,-(SP)
MOV R2,-(SP)
1$: RDLIN

;; CLEAR THE RUBOUT KEY
;; IS CHARACTER A CTRL U?
;; BR IF NO
;; TYPE A CONTROL "U"
;; GO START OVER
;; IS CHARACTER A "TR"?
;; BRANCH IF NO
;; CLEAR THE CHARACTER
;; TYPE A "CR" & "LF"
;; TYPE THE INPUT STRING
;; GO PICKUP ANOTHER CHACTER
;; TYPE A '?'
;; CLEAR THE BUFFER AND LOOP
;; ECHO THE CHARACTER

;; CHECK FOR RETURN
;; LOOP IF NOT RETURN
;; CLEAR RETURN (THE 15)
;; TYPE A LINE FEED
;; CLEAN RUBOUT KEY FROM THE STACK
;; RESTORE R3
;; ADJUST THE STACK AND PUT ADDRESS OF THE
;; FIRST ASCII CHARACTER ON IT

;; STORAGE FOR ASCII CHAR. TO TYPE
;; TERMINATOR
;; RESERVE 8 BYTES FOR TTY INPUT
;; CONTROL "C"
;; CONTROL "U"
;; CONTROL "G"

;; PROVIDE SPACE FOR THE
;; INPUT NUMBER
;; PUSH R0 ON STACK
;; PUSH R1 ON STACK
;; PUSH R2 ON STACK
;; READ AN ASCIZ LINE

```


7343	042016	012600		MOV	(SP)+,R0	:: GET ADDRESS OF 1ST CHARACTER
7344	042020	010037	042124	MOV	R0,5\$	:: AND SAVE IT
7345	042024	005001		CLR	R1	:: CLEAR DATA WORD
7346	042026	005002		CLR	R2	
7347	042030	112046		2\$: MOV	(R0)+,-(SP)	:: PICKUP THIS CHARACTER
7348	042032	001420		BEG	3\$	:: IF ZERO GET OUT
7349	042034	122716	000060	CMPB	#'0,(SP)	:: MAKE SLRE THIS CHARACTER
7350	042040	003026		BGT	4\$	:: IS AN OCTAL DIGIT
7351	042042	122716	000067	CMPB	#'7,(SP)	
7352	042046	002423		BLT	4\$	
7353	042050	006301		ASL	R1	:: *2
7354	042052	006102		ROL	R2	
7355	042054	006301		ASL	R1	:: *4
7356	042056	006102		ROL	R2	
7357	042060	006301		ASL	R1	:: *8
7358	042062	006102		ROL	R2	
7359	042064	042716	177770	BIC	#1C7,(SP)	:: STRIP THE ASCII JUNK
7360	042070	062601		ADD	(SP)+,R1	:: ADD IN THIS DIGIT
7361	042072	000756		BR	2\$	:: LOOP
7362	042074	005726		3\$: TST	(SP)+	:: CLEAN TERMINATOR FROM STACK
7363	042076	010166	000012	MOV	R1,12(SP)	:: SAVE THE RESULT
7364	042102	010237	042134	MOV	R2,\$SHIOCT	
7365	042106	012602		MOV	(SP)+,R2	:: POP STACK INTO R2
7366	042110	012601		MOV	(SP)+,R1	:: POP STACK INTO R1
7367	042112	012600		MOV	(SP)+,R0	:: POP STACK INTO R0
7368	042114	000002		RTI		:: RETURN
7369	042116	005726		4\$: TST	(SP)+	:: CLEAN PARTIAL FROM STACK
7370	042120	105010		CLRB	(R0)	:: SET A TERMINATOR
7371	042122	104401		TYPE		:: TYPE UP THRU THE BAD CHAR.
7372	042124	000000		5\$: .WORD	0	
7373	042126	104401	001210	TYPE	\$QUES	:: "?" "CR" & "LF"
7374	042132	000730		BR	1\$	:: TRY AGAIN
7375	042134	000000		\$SHIOCT: .WORD	0	:: HIGH ORDER BITS GO HERE
7376				.SBTTL	SAVE AND RESTORE R0-R5 ROUTINES	
7377						
7378						
7379						
7380						
7381						
7382						
7383						
7384						
7385						
7386						
7387						
7388						
7389						
7390						
7391						
7392						
7393	042136			\$SAVREG:		
7394	042136	010046		MOV	R0,-(SP)	:: PUSH R0 ON STACK
7395	042140	010146		MOV	R1,-(SP)	:: PUSH R1 ON STACK
7396	042142	010246		MOV	R2,-(SP)	:: PUSH R2 ON STACK
7397	042144	010346		MOV	R3,-(SP)	:: PUSH R3 ON STACK
7398	042146	010446		MOV	R4,-(SP)	:: PUSH R4 ON STACK

```

7399 042150 010546          MOV      R5,-(SP)          ;; PUSH R5 ON STACK
7400 042152 016646 000022    MOV      22(SP),-(SP)      ;; SAVE PS OF MAIN FLOW
7401 042156 016646 000022    MOV      22(SP),-(SP)      ;; SAVE PC OF MAIN FLOW
7402 042162 016646 000022    MOV      22(SP),-(SP)      ;; SAVE PS OF CALL
7403 042166 016646 000022    MOV      22(SP),-(SP)      ;; SAVE PC OF CALL
7404 042172 000002          RTI
7405
7406          ;*RESTORE RO-R5
7407          ;*CALL:
7408          ;*
7409          $RESREG:
7410 042174          MOV      (SP)+,22(SP)          ;; RESTORE PC OF CALL
7411 042200 012666 000022    MOV      (SP)+,22(SP)      ;; RESTORE PS OF CALL
7412 042204 012666 000022    MOV      (SP)+,22(SP)      ;; RESTORE PC OF MAIN FLOW
7413 042210 012666 000022    MOV      (SP)+,22(SP)      ;; RESTORE PS OF MAIN FLOW
7414 042214 012605          MOV      (SP)+,R5          ;; POP STACK INTO R5
7415 042216 012604          MOV      (SP)+,R4          ;; POP STACK INTO R4
7416 042220 012603          MOV      (SP)+,R3          ;; POP STACK INTO R3
7417 042222 012602          MOV      (SP)+,R2          ;; POP STACK INTO R2
7418 042224 012601          MOV      (SP)+,R1          ;; POP STACK INTO R1
7419 042226 012600          MOV      (SP)+,R0          ;; POP STACK INTO R0
7420 042230 000002          RTI
7421
7422          SBTTL  POWER DOWN AND UP ROUTINE
7423
7424          ;;*****
7425
7426          ;POWER DOWN ROUTINE
7427 042232 017737 136702 003312 $PWRDN: MOV      @SWR,SAVSWR      ;SAVE SWITCH REGISTER
7428 042240 012737 042260 000024    MOV      @PWRUP,PWRVEC      ;SET UP VECTOR
7429 042246 012737 000340 000026    MOV      @PR7,PWAVEC+2
7430 042254 000000          HALT
7431 042256 000776          BR      .-2          ;HANG UP
7432
7433          ;;*****
7434
7435          ;POWER UP ROUTINE
7436 042260 005037 042354          $PWRUP: CLR      $PWRCT          ;LOAD WAIT COUNT
7437 042264 012737 000144 042356    MOV      #100,$PWRCT+2
7438 042272 005237 042354          1$: INC      $PWRCT          ;WAIT FOR TELETYPE
7439 042276 001375          BNE      1$
7440 042300 005337 042356          DEC      $PWRCT+2
7441 042304 001372          BNE      1$
7442 042306 012737 042232 000024    MOV      @PWRDN,PWRVEC      ;SET UP FOR POWER DOWN VECTOR
7443 042314 012737 000340 000026    MOV      @PR7,PWAVEC+2
7444 042322 012706 001100          MOV      @STACK,SP          ;FORCE STACK
7445 042326 104401 042360          TYPE      $POWER          ;TYPE POWER
7446 042332 004737 034366          JSR      PC,PARCHK      ;REINITIALIZE MEMORY CHECK ENABLE
7447 042336 013777 003312 136574    MOV      SAVSWR,@SWR      ;RESTORE SWITCH REGISTER
7448 042344 013702 001270          MOV      $BASE,R2      ;REINITIALISE R2 FOR '611 BASE
7449 042350 000177 136532          JMP      @SLPADR ;GO BACK TO LAST TEST
7450
7451 042354 000000 000000          $PWRCT: .WORD 0,0          ;TELETYPE TIME OUT
7452 042360 047520 042527 000122    $POWER: .ASCIZ /POWER/
7453          .EVEN
7454          .SBTTL  TRAP DECODER

```

```
7455
7456
7457
7458
7459
7460
7461
7462 042366 010046
7463 042370 016600 000002
7464 042374 005740
7465 042376 111000
7466 042400 006300
7467 042402 016000 042422
7468 042406 000200
7469
7470
7471
7472
7473 042410 011646
7474 042412 016666 000004 000002
7475 042420 000002
7476
7477
7478
7479
7480
7481
7482
7483
7484 042422 042410
7485 042424 037604
7486 042426 040112
7487 042430 040066
7488 042432 040126
7489 042434 040314
7490
7491 042436 041126
7492
7493 042440 041036
7494 042442 041400
7495 042444 041470
7496 042446 041776
7497 042450 042136
7498 042452 042174
7499 042454 035302

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

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

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

$TRAP2: MOV      (SP), -(SP)       ;;MOVE THE PC DOWN
        MOV      4(SP), 2(SP)     ;;MOVE THE PSW DOWN
        RTI                      ;;RESTORE THE PSW

.SBTTL  TRAP TABLE

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

; ROUTINE
;-----
$TRPAD: .WORD     $TRAP2          TRAP+1(104401)  TTY TYPEOUT ROUTINE
        $TYPE     ;;CALL=TYPE    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOC    ;;CALL=TYPOC   TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPOS    ;;CALL=TYPOS   TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPON    ;;CALL=TYPON   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
        $TYPDS    ;;CALL=TYPDS
        $GTSWR    ;;CALL=GTSWR   TRAP+6(104406)  GET SOFT-SWR SETTING
        $CKSWR    ;;CALL=CKSWR   TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR    ;;CALL=RDCHR   TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN    ;;CALL=RDLIN   TRAP+11(104411) TTY TYPEIN STRING ROUTINE
        $RDOCT    ;;CALL=RDOCT   TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
        $SAVREG   ;;CALL=SAVREG  TRAP+13(104413) SAVE RO-R5 ROUTINE
        $RESREG   ;;CALL=RESREG  TRAP+14(104414) RESTORE RO-R5 ROUTINE
        $SCOPI$   ;;CALL=SCOPI   TRAP+15(104415) INTERNAL LOOP ON ERROR
```

				.SBTTL DATA TABLE FOR PRINT OUT	
7500					
7501					
7502	042456	001220	003244	DT000:	.WORD \$TESTN,TRAPPC
7503	042462	001220	001116	DT002:	.WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT
7504	042470	003164	003252		
7505	042476	003256	003260		
7506	042504	001220	001116	DT003:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3
7507	042512	003140	003226		
7508	042520	003230	003170		
7509	042524	001220	001116	DT041:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,E.BA,T.BA,E.WC,T.WC
7510	042532	003140	003210		
7511	042540	003214	003154		
7512	042546	003144	003202		
7513	042554	001220	001116	DT046:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7514	042562	003140	003210		
7515	042570	003214	003154		
7516	042574	001220	001116	DT051:	.WORD \$TESTN,\$ERRPC,E.DB,T.DB,WRDCNT
7517	042602	003162	003264		
7518	042606	001220	001116	DT052:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,T.CS2,T.ER
7519	042614	003140	003150		
7520	042622	001220	001116	DT053:	.WORD \$TESTN,\$ERRPC,E.MR1,T.MR1
7521	042630	003164			
7522	042632	001220	001116	DT054:	.WORD \$TESTN,\$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
7523	042640	003160	003206		
7524	042646	001220	001116	DT071:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7525	042654	003140	003210		
7526	042662	003214	003154		
7527	042666	055174	055176		
7528	042674	001220	001116	DT074:	.WORD VRCHDR,VRCHDR+2,VRCHDR+4 .WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,HDRCNT
7529	042702	003140	003210		
7530	042710	003214	003154		
7531	042716	001220	001116	DT077:	.WORD \$TESTN,\$ERRPC,E.MR1,T.MR1,HDRCNT
7532	042724	003164	003310		
7533	042730	001220	001116	DT134:	.WORD \$TESTN,\$ERRPC,E.ECPT,T.ECPT,BITCNT
7534	042736	003174	003262		
7535	042742	001220	001116	DT144:	.WORD \$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7536	042750	003140	003210		
7537	042756	003214	003154		
7538	042762	003204	003144		
7539	042770	003142	003220		
7540	042776	003206	003146		
7541	043002	001220	001116	DT151:	.WORD \$TESTN,\$ERRPC,E.ECPT,T.ECPT
7542	043010	003174			

7543
7544
7545 043012 000001
7546 043014 002 000
7547 043016 000005
7548 043020 000 000
7549 043022 044065
7550 043024 000 000
7551 043026 044103
7552 043030 002 000
7553 043032 044147
7554 043034 000 000
7555 043036 044233
7556 043040 007 000
7557 043042 000005
7558 043044 000 000
7559 043046 044065
7560 043050 000 000
7561 043052 044103
7562 043054 002 000
7563 043056 044321
7564 043060 000 000
7565 043062 044400
7566 043064 006 000
7567 043066 000005
7568 043070 000 000
7569 043072 044065
7570 043074 000 000
7571 043076 044103
7572 043100 002 000
7573 043102 044457
7574 043104 000 000
7575 043106 044536
7576 043110 006 000
7577 043112 000007
7578 043114 000 000
7579 043116 044065
7580 043120 000 000
7581 043122 044103
7582 043124 002 000
7583 043126 044616
7584 043130 000 000
7585 043132 044675
7586 043134 006 000
7587 043136 044752
7588 043140 000 000
7589 043142 045011
7590 043144 004 000
7591 043146 000005
7592 043150 000 000
7593 043152 044065
7594 043154 000 000
7595 043156 044103
7596 043160 002 000
7597 043162 044616
7598 043164 000 000

.SBTTL DATA FORMAT FOR PRINT OUT

DF000: .WORD 1
.BYTE 2,0
DF002: .WORD 5
.BYTE 0,0
.WORD DH0C0A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH002A
.BYTE 0,0
.WORD DH002B
.BYTE 7,0
DF003: .WORD 5 ;ERROR 3-24
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH003A
.BYTE 0,0
.WORD DH003B
.BYTE 6,0
DF025: .WORD 5 ;ERROR 25-40
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH025A
.BYTE 0,0
.WORD DH025B
.BYTE 6,0
DF041: .WORD 7
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH041A
.BYTE 0,0
.WORD DH041B
.BYTE 6,0
.WORD DH041C
.BYTE 0,0
.WORD DH041D
.BYTE 4,0
DF046: .WORD 5
.BYTE 0,0
.WORD DH000A
.BYTE 0,0
.WORD DH000B
.BYTE 2,0
.WORD DH041A
.BYTE 0,0

7599	043166	044675		.WORD	DH041B	
7600	043170	006	000	.BYTE	6,0	
7601	043172	000005		DF051: .WORD	5,0	
7602	043174	000	000	.BYTE	0,0	
7603	043176	044065		.WORD	DH000A	
7604	043200	000	000	.BYTE	0,0	
7605	043202	044103		.WORD	DH0C0B	
7606	043204	002	000	.BYTE	2,0	
7607	043206	045046		.WORD	DH051A	
7608	043210	000	000	.BYTE	0,0	
7609	043212	045073		.WORD	DH051B	
7610	043214	003	000	.BYTE	3,0	
7611	2216	000005		DF052: .WORD	5,0	
7612	043220	000	000	.BYTE	0,0	
7613	043222	044065		.WORD	DH000A	
7614	043224	000	000	.BYTE	0,0	
7615	043226	044103		.WORD	DH000B	
7616	043230	002	000	.BYTE	2,0	
7617	043232	045121		.WORD	DH052A	
7618	043234	000	000	.BYTE	0,0	
7619	043236	045140		.WORD	DH052B	
7620	043240	004	000	.BYTE	4,0	
7621	043242	000005		DF053: .WORD	5,0	
7622	043244	000	000	.BYTE	0,0	
7623	043246	044065		.WORD	DH000A	
7624	043250	000	000	.BYTE	0,0	
7625	043252	044103		.WORD	DH000B	
7626	043254	002	000	.BYTE	2,0	
7627	043256	045175		.WORD	DH053A	
7628	043260	000	000	.BYTE	0,0	
7629	043262	045214		.WORD	DH053B	
7630	043264	002	000	.BYTE	2,0	
7631	043266	000005		DF054: .WORD	5,0	; ERROR-54-61
7632	043270	000	000	.BYTE	0,0	
7633	043272	044065		.WORD	DH000A	
7634	043274	000	000	.BYTE	0,0	
7635	043276	044103		.WORD	DH000B	
7636	043300	002	000	.BYTE	2,0	
7637	043302	045232		.WORD	DH054A	
7638	043304	000	000	.BYTE	0,0	
7639	043306	045271		.WORD	DH054B	
7640	043310	004	000	.BYTE	4,0	
7641	043312	000007		DF071: .WORD	7,0	; ERRORS 71-73
7642	043314	000	000	.BYTE	0,0	
7643	043316	044065		.WORD	DH000A	
7644	043320	000	000	.BYTE	0,0	
7645	043322	044103		.WORD	DH000B	
7646	043324	002	000	.BYTE	2,0	
7647	043326	044616		.WORD	DH041A	
7648	043330	000	000	.BYTE	0,0	
7649	043332	044675		.WORD	DH041B	
7650	043334	006	000	.BYTE	6,0	
7651	043336	045326		.WORD	DH071A	
7652	043340	000	000	.BYTE	0,0	
7653	043342	045347		.WORD	DH071B	
7654	043344	003	000	.BYTE	3,0	

7655	043346	000005		DF074:	.WORD	5
7656	043350	000	000		.BYTE	0,0
7657	043352	044065			.WORD	0H000A
7658	043354	000	000		.BYTE	0,0
7659	043356	044103			.WORD	0H000B
7660	043360	002	000		.BYTE	2,0
7661	043362	045376			.WORD	0H074A
7662	043364	000	000		.BYTE	0,0
7663	043366	045465			.WORD	0H074B
7664	043370	007	000		.BYTE	7,0
7665	043372	000005		DF077:	.WORD	5
7666	043374	000	000		.BYTE	0,0
7667	043376	044065			.WORD	0H000A
7668	043400	000	000		.BYTE	0,0
7669	043402	044103			.WORD	0H000B
7670	043404	002	000		.BYTE	2,0
7671	043406	045551			.WORD	0H077A
7672	043410	000	000		.BYTE	0,0
7673	043412	045600			.WORD	0H077B
7674	043414	003	000		.BYTE	3,0
7675	043416	000005		DF134:	.WORD	5
7676	043420	000	000		.BYTE	0,0
7677	043422	044065			.WORD	0H000A
7678	043424	000	000		.BYTE	0,0
7679	043426	044103			.WORD	0H000B
7680	043430	002	000		.BYTE	2,0
7681	043432	045624			.WORD	0H134A
7682	043434	000	000		.BYTE	0,0
7683	043436	045650			.WORD	0H134B
7684	043440	003	000		.BYTE	3,0
7685	043442	000007		DF144:	.WORD	7
7686	043444	000	000		.BYTE	0,0
7687	043446	044065			.WORD	0H000A
7688	043450	000	000		.BYTE	0,0
7689	043452	044103			.WORD	0H000B
7690	043454	002	000		.BYTE	2,0
7691	043456	044616			.WORD	0H041A
7692	043460	000	000		.BYTE	0,0
7693	043462	044675			.WORD	0H041B
7694	043464	006	000		.BYTE	6,0
7695	043466	045676			.WORD	0H144A
7696	043470	000	000		.BYTE	0,0
7697	043472	045775			.WORD	0H144B
7698	043474	010	000		.BYTE	10,0
7699	043476	000005		DF151:	.WORD	5
7700	043500	000	000		.BYTE	0,0
7701	043502	044065			.WORD	0H000A
7702	043504	000	000		.BYTE	0,0
7703	043506	044103			.WORD	0H000B
7704	043510	002	000		.BYTE	2,0
7705	043512	045121			.WORD	0H052A
7706	043514	000	000		.BYTE	0,0
7707	043516	046072			.WORD	0H151A
7708	043520	002	000		.BYTE	2,0

N11

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 143
ASCII MESSAGES

SEQ 0143

```
7709 .SBTTL ASCII MESSAGES
7710
7711 043522 005015 045522 030466 OPR001: .ASCIZ <15><12>/RK611 BUS ADDRESS ( /
7712 043530 020061 052502 020123
7713 043536 042101 051104 051505
7714 043544 020123 020050 000
7715 043551 040 020051 020075 OPR002: .ASCIZ / ) = /
7716 043556 000
7717 043557 122 033113 030461 OPR003: .ASCIZ /RK611 VECTOR ADDRESS ( /
7718 043564 053040 041505 047524
7719 043572 020122 042101 051104
7720 043600 051505 020123 020050
7721 043606 000
7722 043607 122 033113 030461 OPR004: .ASCIZ /RK611 PRIORITY ( /
7723 043614 050040 044522 051117
7724 043622 052111 020131 020050
7725 043630 000
7726 043631 015 025012 025052 OPR005: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
7727 043636 025052 020040 051120
7728 043644 043517 040522 020115
7729 043652 040510 052114 042105
7730 043660 025040 025052 025052
7731 043666 005015 000
7732 043671 015 051412 041505 OPR006: .ASCIZ <15><12>/SECOND PASS RUN TIME IS APPROX 3:15 MINUTES/<15><12>
7733 043676 047117 020104 040520
7734 043704 051523 051040 047125
7735 043712 052040 046511 020105
7736 043720 051511 040440 050120
7737 043726 047522 020130 035063
7738 043734 032461 046440 047111
7739 043742 052125 051505 005015
7740 043750 000
7741 043751 040 000040
7742 043754 005015 051120 043517 SPACE2: .ASCIZ / /
7743 043762 040522 020115 041101 ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
7744 043770 051117 042524 020104
7745 043776 042502 040503 051525
7746 044004 020105 051105 047522
7747 044012 020122 044124 042522
7748 044020 044123 046117 020104
7749 044026 054105 042503 042105
7750 044034 042105 005015 000
7751 044041 015 052012 051505 TSTBY1: .ASCIZ <15><12>/TEST /
7752 044046 020124 000
7753 044051 040 054502 040520 TSTBY2: .ASCIZ / BYPASSED/<15><12>
7754 044056 051523 042105 005015
7755 044064 000
```


.SBTTL DATA HEADERS			
7756			
7757			
7758	044065	124	051505 020124
7759	044072	020040	042440 051122
7760	044100	051117	000
7761	044103	116	046525 020040
7762	044110	020040	050040 000103
7763	044116	042524	052123 020040
7764	044124	020040	051124 050101
7765	044132	005015	
7766	044134	052516	020115 020040
7767	044142	020040	041520 000
7768	044147	105	050130 041505
7769	044154	020124	040440 052103
7770	044162	040525	020114 050040
7771	044170	042522	042523 052116
7772	044176	050040	042522 042523
7773	044204	052116	050040 042522
7774	044212	042523	052116 050040
7775	044220	042522	042523 052116
7776	044226	041040	052111 000
7777	044233	122	046513 030522
7778	044240	020040	051040 046513
7779	044246	030522	020040 041040
7780	044254	052111	030453 020040
7781	044262	041040	052111 020040
7782	044270	020040	041040 052111
7783	044276	030455	020040 041040
7784	044304	052111	031055 020040
7785	044312	041440	052517 052116
7786	044320	000	
7787	044321	105	050130 041505
7788	044326	020124	040440 052103
7789	044334	040525	020114 042440
7790	044342	050130	041505 020124
7791	044350	040440	052103 040525
7792	044356	020114	042440 050130
7793	044364	041505	020124 040440
7794	044372	052103	040525 000114
7795	044400	045522	051503 020061
7796	044406	020040	045522 051503
7797	044414	020061	020040 042515
7798	044422	051523	040440 020040
7799	044430	042515	051523 040440
7800	044436	020040	042515 051523
7801	044444	041040	020040 042515
7802	044452	051523	041040 000
7803	044457	105	050130 041505
7804	044464	020124	040440 052103
7805	044472	040525	020114 042440
7806	044500	050130	041505 020124
7807	044506	040440	052103 040525
7808	044514	020114	042440 050130
7809	044522	041505	020124 040440
7810	044530	052103	040525 000114
7811	044536	045522	051503 020061

.SBTTL DATA HEADERS			
DH000A:	.ASCIZ	/TEST	ERROR/
DH000B:	.ASCIZ	/NUM	PC/
DH000C:	.ASCII	/TEST	TRAP/(<15>(<12>
DH002A:	.ASCIZ	/EXPECT	ACTUAL PRESENT PRESENT PRESENT PRESENT BIT/
DH002B:	.ASCIZ	/RKMR1	RKMR1 BIT+1 BIT BIT-1 BIT-2 COUNT/
DH003A:	.ASCIZ	/EXPECT	ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
DH003B:	.ASCIZ	/RKCS1	RKCS1 MESS A MESS A MESS B MESS B/
DH025A:	.ASCIZ	/EXPECT	ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
DH025B:	.ASCIZ	/RKCS1	RKCS1 HD WD 1 HD WD 1 HD WD 2 HD WD 2

7812	044544	020040	045522	051503							
7813	044552	020061	020040	042110							
7814	044560	053440	020104	020061							
7815	044566	042110	053440	020104							
7816	044574	020061	042110	053440							
7817	044602	020104	020062	042110							
7818	044610	053440	020104	000062							
7819	044616	054105	042520	052103	DH041A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT ACTUAL/
7820	044624	020040	041501	052524							
7821	044632	046101	020040	054105							
7822	044640	042520	052103	020040							
7823	044646	041501	052524	046101							
7824	044654	020040	054105	042520							
7825	044662	052103	020040	041501							
7826	044670	052524	046101	000							
7827	044675	122	041513	030523	DH041B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2	RKER RKER/
7828	044702	020040	051040	041513							
7829	044710	030523	020040	051040							
7830	044716	041513	031123	020040							
7831	044724	051040	041513	031123							
7832	044732	020040	051040	042513							
7833	044740	020122	020040	051040							
7834	044746	042513	000122								
7835	044752	054105	042520	052103	DH041C:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/	
7836	044760	020040	041501	052524							
7837	044766	046101	020040	054105							
7838	044774	042520	052103	020040							
7839	045002	041501	052524	046101							
7840	045010	000									
7841	045011	122	041113	020101	DH041D:	.ASCIZ	/RKBA	RKBA	RKWC	RKWC/	
7842	045016	020040	051040	041113							
7843	045024	020101	020040	051040							
7844	045032	053513	020103	020040							
7845	045040	051040	053513	000103							
7846	045046	040504	040524	020040	DH051A:	.ASCIZ	/DATA	DATA	WORD/		
7847	045054	020040	040504	040524							
7848	045062	020040	020040	047527							
7849	045070	042122	000								
7850	045073	105	050130	041505	DH051B:	.ASCIZ	/EXPECT	READ	COUNT/		
7851	045100	020124	051040	040505							
7852	045106	020104	020040	041440							
7853	045114	052517	052116	000							
7854	045121	105	050130	041505	DH052A:	.ASCIZ	/EXPECT	ACTUAL/			
7855	045126	020124	040440	052103							
7856	045134	040525	000114								
7857	045140	045522	051503	020061	DH052B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKER/	
7858	045146										

7868	045232	054105	042520	052103													
7869	045240	020040	041501	052524	DH054A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/							
7870	045246	046101	020040	054105													
7871	045254	042520	052103	020040													
7872	045262	041501	052524	046101													
7873	045270	000															
7874	045271	122	042113	054503	DH054B:	.ASCIZ	/RKDCYL	RKDCYL	RKDA	RKDA/							
7875	045276	020114	051040	042113													
7876	045304	054503	020114	051040													
7877	045312	042113	020101	020040													
7878	045320	051040	042113	000101													
7879	045326	042510	042101	051105	DH071A:	.ASCIZ	/HEADER	SIMULATED/									
7880	045334	051440	046511	046125													
7881	045342	052101	042105	000													
7882	045347	127	051117	020104	DH071B:	.ASCIZ	/WORD 1	WORD 2	WORD 3/								
7883	045354	020061	053440	051117													
7884	045362	020104	020062	053440													
7885	045370	051117	020104	000063													
7886	045376	054105	042520	052103	DH074A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT	ACTUAL	HEADER/				
7887	045404	020040	041501	052524													
7888	045412	046101	020040	054105													
7889	045420	042520	052103	020040													
7890	045426	041501	052524	046101													
7891	045434	020040	054105	042520													
7892	045442	052103	020040	041501													
7893	045450	052524	046101	020040													
7894	045456	042510	042101	051105													
7895	045464	000															
7896	045465	122	041513	030523	DH074B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2	RKER	RKER	NUM/				
7897	045472	020040	051040	041513													
7898	045500	030523	020040	051040													
7899	045506	041513	031123	020040													
7900	045514	051040	041513	031123													
7901	045522	020040	051040	042513													
7902	045530	020122	020040	051040													
7903	045536	042513	020122	020040													
7904	045544	047040															

SEO 0147

CZR6DBD PK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 148
ERROR MESSAGES

SEQ 0148

.SBTTL ERROR MESSAGES

7946					
7947					
7948	046111	125	042516	050130	EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
7949	046116	041505	042524	020104	
7950	046124	042515	047515	054522	
7951	046132	050040	051101	052111	
7952	046140	020131	047105	041101	
7953	046146	042514	052040	040522	
7954	046154	000120			
7955	046156	052101	042524	050115	EM300: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM READ DATA/
7956	046164	044524	043516	052040	
7957	046172	020117	044103	041505	
7958	046200	020113	042523	045505	
7959	046206	046440	051505	020123	
7960	046214	051106	046517	051040	
7961	046222	040505	020104	040504	
7962	046230	040524	000		
7963	046233	101	052124	046505	EM301: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE DATA/
7964	046240	052120	047111	020107	
7965	046246	047524	041440	042510	
7966	046254	045503	051440	042505	
7967	046262	020113	042515	051523	
7968	046270	043040	047522	020115	
7969	046276	051127	052111	020105	
7970	046304	040504	040524	000	
7971	046311	101	052124	046505	EM302: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE CHECK/
7972	046316	052120	047111	020107	
7973	046324	047524	041440	042510	
7974	046332	045503	051440	042505	
7975	046340	020113	042515	051523	
7976	046346	043040	047522	020115	
7977	046354	051127	052111	020105	
7978	046362	044103	041505	000113	
7979	046370	052101	042524	050115	EM303: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM READ DATA/
7980	046376	044524	043516	052040	
7981	046404	020117	044103	041505	
7982	046412	020113	046103	040505	
7983	046420	020122	042515	051523	
7984	046426	043040	047522	020115	
7985	046434	042522	042101	042040	
7986	046442	052101	000101		
7987	046446	052101	042524	050115	EM304: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE DATA/
7988	046454	044524	043516	052040	
7989	046462	020117	044103	041505	
7990	046470	020113	046103	040505	
7991	046476	020122	042515	051523	
7992	046504	043040	047522	020115	
7993	046512	051127	052111	020105	
7994	046520	040504	040524	000	
7995	046525	101	052124	046505	EM305: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE CHECK/
7996	046532	052120	047111	020107	
7997	046540	047524	041440	042510	
7998	046546	045503	041440	042514	
7999	046554	051101	046440	051505	
8000	046562	020123	051106	046517	
8001	046570	053440	044522	042524	

8002	046576	041440	042510	045503	
8003	046604	000			
8004	046605	101	052124	046505	EM306: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/(<15><12>
8005	046612	052120	047111	020107	
8006	046620	047524	041440	042510	
8007	046626	045503	044040	040505	
8008	046634	020104	042507	042516	
8009	046642	040522	044524	047117	
8010	046650	005015			
8011	046652	044527	044124	053040	.ASCIZ /WITH VARIOUS CYLINDER VALUES/
8012	046660	051101	047511	051525	
8013	046666	041440	046131	047111	
8014	046674	042504	020122	040526	
8015	046702	052514	051505	000	EM307: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/(<15><12>
8016	046707	101	052124	046505	
8017	046714	052120	047111	020107	
8018	046722	047524	041440	042510	
8019	046730	045503	044040	040505	
8020	046736	020104	042507	042516	
8021	046744	040522	044524	047117	
8022	046752	005015			
8023	046754	044527	044124	053040	.ASCIZ /WITH VARIOUS TRACK VALUES/
8024	046762	051101	047511	051525	
8025	046770	052040	040522	045503	
8026	046776	053040	046101	042525	
8027	047004	000123			
8028	047006	052101	042524	050115	EM308: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/(<15><12>
8029	047014	044524	043516	052040	
8030	047022	020117	044103	041505	
8031	047030	020113	042510	042101	
8032	047036	043440	047105	051105	
8033	047044	052101	047511	006516	
8034	047052	012			
8035	047053	127	052111	020110	.ASCIZ /WITH VARIOUS SECTOR VALUES/
8036	047060	040526	044522	052517	
8037	047066	020123	042523	052103	
8038	047074	051117	053040	046101	
8039	047102	042525	000123		
8040	047106	052101	042524	050115	EM309: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/(<15><12>
8041	047114	044524	043516	052040	
8042	047122	020117	044103	041505	
8043	047130	020113	042510	042101	
8044	047136	043440	047105	051105	
8045	047144	052101	047511	006516	
8046	047152	012			
8047	047153	127	052111	020110	.ASCIZ /WITH VARIOUS FORMAT VALUES/
8048	047160	040526	044522	052517	
8049	047166	020123	047506	046522	
8050	047174	052101	053040	046101	
8051	047202	042525	000123		
8052	047206	052101	042524	050115	EM310: .ASCIZ /ATTEMPTING TO CHECK NPR DATA TRANSFER FOR WRITE DATA/
8053	047214	044524	043516	052040	
8054	047222	020117	044103	041505	
8055	047230	020113	050116	020122	
8056	047236	040504	040524	052040	
8057	047244	040522	051516	042506	

8058	047252	020122	047506	020122	
8059	047250	051127	052111	020105	
8060	047266	040504	040524	000	
8061	047273	101	052124	046505	EM311: .ASCIZ /ATTEMPTING TO CHECK HEADER RECOGNITION/
8062	047300	052120	047111	020107	
8063	047306	047524	041440	042510	
8064	047314	045503	044040	040505	
8065	047322	042504	020122	042522	
8066	047330	047503	047107	052111	
8067	047336	047511	000116		
8068	047342	052101	042524	050115	EM312: .ASCIZ /ATTEMPTING TO CHECK SECTOR INCREMENT/
8069	047350	044524	043516	052040	
8070	047356	020117	044103	041505	
8071	047364	020113	042523	052103	
8072	047372	051117	044440	041516	
8073	047400	042522	042515	052116	
8074	047406	000			
8075	047407	101	052124	046505	EM313: .ASCIZ /ATTEMPTING TO CHECK TRACK INCREMENT/
8076	047414	052120	047111	020107	
8077	047422	047524	041440	042510	
8078	047430	045503	052040	040522	
8079	047436	045503	044440	041516	
8080	047444	042522	042515	052116	
8081	047452	000			
8082	047453	101	052124	046505	EM314: .ASCIZ /ATTEMPTING TO CHECK CYLINDER INCREMENT/
8083	047460	052120	047111	020107	
8084	047466	047524	041440	042510	
8085	047474	045503	041440	046131	
8086	047502	047111	042504	020122	
8087	047510	047111	051103	046505	
8088	047516	047105	000124		
8089	047522	052101	042524	050115	EM315: .ASCII /ATTEMPTING TO CHECK SECTOR PULSE DETECTION/<15><12>
8090	047530	044524	043516	052040	
8091	047536	020117	044103	041505	
8092	047544	020113	042523	052103	
8093	047552	051117	050040	046125	
8094	047560	042523	042040	052105	
8095	047566	041505	044524	047117	
8096	047574	005015			
8097	047576	044527	044124	053440	.ASCIZ /WITH WRITE DATA/
8098	047604	044522	042524	042040	
8099	047612	052101	000101		
8100	047616	052101	042524	050115	EM316: .ASCIZ /ATTEMPTING TO FORCE BAD SECTOR ERROR/
8101	047624	044524	043516	052040	
8102	047632	020117	047506	041522	
8103	047640	020105	040502	020104	
8104	047646	042523	052103	051117	
8105	047654	042440	051122	051117	
8106	047662	000			
8107	047663	101	052124	046505	EM317: .ASCII /ATTEMPTING TO FORCE HEADER VRC ERROR/<15><12>
8108	047670	052120	047111	020107	
8109	047676	047524	043040	051117	
8110	047704	042503	044040	040505	
8111	047712	042504	020122	051126	
8112	047720	020103	051105	047522	
8113	047726	006522	012		

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 151
ERROR MESSAGES

SEQ 0151

8114	047731	127	052111	020110	.ASCIZ /WITH BAD SECTOR PRESENT/
8115	047736	040502	020104	042523	
8116	047744	052103	051117	050040	
8117	047752	042522	042523	052116	
8118	047760	000			
8119	047761	101	052124	046505	EM318: .ASCIZ /ATTEMPTING TO FORCE HVRC ERROR/
8120	047766	052120	047111	020107	
8121	047774	047524	043040	051117	
8122	050002	042503	044040	051126	
8123	050010	020103	051105	047522	
8124	050016	000122			
8125	050020	052101	042524	050115	EM319: .ASCIZ /ATTEMPTING TO FORCE OPERATION INCOMPLETE/
8126	050026	044524	043516	052040	
8127	050034	020117	047506	041522	
8128	050042	020105	050117	051105	
8129	050050	052101	047511	020116	
8130	050056	047111	047503	050115	
8131	050064	042514	042524	000	
8132	050071	101	052124	046505	EM320: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37/
8133	050076	044520	043516	052040	
8134	050104	020117	047506	041522	
8135	050112	020105	050117	020111	
8136	050120	044527	044124	044040	
8137	050126	051126	020103	051105	
8138	050134	047522	020122	047117	
8139	050142	044040	040505	042504	
8140	050150	020122	033463	000	
8141	050155	101	052124	046505	EM321: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36/
8142	050162	044520	043516	052040	
8143	050170	020117	047506	041522	
8144	050176	020105	050117	020111	
8145	050204	044527	044124	044040	
8146	050212	051126	020103	051105	
8147	050220	047522	020122	047117	
8148	050226	044040	040505	042504	
8149	050234	020122	033063	000	
8150	050241	101	052124	046505	EM322: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0/
8151	050246	044520	043516	052040	
8152	050254	020117	047506	041522	
8153	050262	020105	050117	020111	
8154	050270	044527	044124	044040	
8155	050276	051126	020103	051105	
8156	050304	047522	020122	047117	
8157	050312	044040	040505	042504	
8158	050320	020122	000060		
8159	050324	044103	041505	044513	EM323: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS BAD SECTOR ERROR/
8160	050332	043516	044040	040505	
8161	050340	042504	020122	042522	
8162	050346	047503	047107	052111	
8163	050354	047511	020116	044527	
8164	050362	044124	050040	042522	
8165	050370	044526	052517	020123	
8166	050376	040502	020104	042523	
8167	050404	052103	051117	042440	
8168	050412	051122	051117	000	
8169	050417	103	042510	045503	EM324: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER VRC ERROR/

8170	050424	047111	020107	042510
8171	050432	042101	051105	051040
8172	050440	041505	043517	044516
8173	050446	044524	047117	053440
8174	050454	052111	020110	051120
8175	050462	053105	047511	051525
8176	050470	044040	040505	042504
8177	050476	020122	051126	020103
8178	050504	051105	047522	000122
8179	050512	047506	041522	047111
8180	050520	020107	040502	020104
8181	050526	042523	052103	051117
8182	050534	042440	051122	051117
8183	050542	053440	052111	020110
8184	050550	051120	053105	047511
8185	050556	051525	044040	040505
8186	050564	042504	020122	051126
8187	050572	020103	051105	047522
8188	050600	000122		
8189	050602	047506	041522	047111
8190	050610	020107	042510	042101
8191	050616	053040	041522	042440
8192	050624	051122	051117	053440
8193	050632	052111	020110	051120
8194	050640	053105	047511	051525
8195	050646	041040	042101	051440
8196	050654	041505	047524	020122
8197	050662	051105	047522	000122
8198	050670	052101	042524	050115
8199	050676	044524	043516	052040
8200	050704	020117	051127	052111
8201	050712	020105	054523	041516
8202	050720	020110	044527	044124
8203	050726	053440	044522	042524
8204	050734	042040	052101	000101
8205	050742	052101	042524	050115
8206	050750	044524	043516	052040
8207	050756	020117	051127	052111
8208	050764	020105	040504	040524
8209	050772	043040	042511	042114
8210	051000	053440	052111	020110
8211	051006	051127	052111	020105
8212	051014	040504	040524	000
8213	051021	101	052124	046505
8214	051026	052120	047111	020107
8215	051034	047524	041440	042510
8216	051042	045503	042440	041503
8217	051050	050040	052101	042524
8218	051056	047122	041440	042514
8219	051064	051101	047111	000107
8220	051072	052101	042524	050115
8221	051100	044524	043516	052040
8222	051106	020117	044103	041505
8223	051114	020113	041505	020103
8224	051122	042507	042516	040522
8225	051130	044524	047117	000

EM325: .ASCIZ /FORCING BAD SECTOR ERROR WITH PREVIOUS HEADER VRC ERROR/

EM326: .ASCIZ /FORCING HEAD VRC ERROR WITH PREVIOUS BAD SECTOR ERROR/

EM327: .ASCIZ /ATTEMPTING TO WRITE SYNCH WITH WRITE DATA/

EM328: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD WITH WRITE DATA/

EM329: .ASCIZ /ATTEMPTING TO CHECK ECC PATTERN CLEARING/

EM330: .ASCIZ /ATTEMPTING TO CHECK ECC GENERATION/

8226	051135	101	052124	046505	EM331: .ASCIZ /ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR/
8227	051142	052120	047111	020107	
8228	051150	040502	020104	042523	
8229	051156	052103	051117	042440	
8230	051164	051122	051117	051440	
8231	051172	052105	044524	043516	
8232	051200	041440	047117	051124	
8233	051206	046117	042514	020122	
8234	051214	051105	047522	000122	
8235	051222	052101	042524	050115	EM332: .ASCIZ /ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR/
8236	051230	044524	043516	044040	
8237	051236	040505	042504	020122	
8238	051244	051126	020103	051105	
8239	051252	047522	020122	042523	
8240	051260	052124	047111	020107	
8241	051266	047503	052116	047522	
8242	051274	046114	051105	042440	
8243	051302	051122	051117	000	
8244	051307	101	052124	046505	EM333: .ASCIZ /ATTEMPTING TO WRITE ECC WORDS/
8245	051314	052120	047111	020107	
8246	051322	047524	053440	044522	
8247	051330	042524	042440	041503	
8248	051336	053440	051117	051504	
8249	051344	000			
8250	051345	101	052124	046505	EM334: .ASCIZ /ATTEMPTING TO WRITE POSTAMBLE/
8251	051352	052120	047111	020107	
8252	051360	047524	053440	044522	
8253	051366	042524	050040	051517	
8254	051374	040524	041115	042514	
8255	051402	000			
8256	051403	101	052124	046505	EM335: .ASCIZ /ATTEMPTING COMPLETE EXECUTION OF WRITE DATA/
8257	051410	052120	047111	020107	
8258	051416	047503	050115	042514	
8259	051424	042524	042440	042530	
8260	051432	052503	044524	047117	
8261	051440	047440	020106	051127	
8262	051446	052111	020105	040504	
8263	051454	040524	000		
8264	051457	101	052124	046505	EM336: .ASCIZ /ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR/
8265	051464	052120	047111	020107	
8266	051472	051105	047522	020122	
8267	051500	046103	040505	020122	
8268	051506	044527	044124	041440	
8269	051514	047117	051124	046117	
8270	051522	042514	020122	046103	
8271	051530	040505	000122		
8272	051534	052101	042524	050115	EM337: .ASCIZ /ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL/
8273	051542	044524	043516	050040	
8274	051550	051101	044524	046101	
8275	051556	051440	041505	047524	
8276	051564	020122	051127	052111	
8277	051572	020105	044527	044124	
8278	051600	055040	051105	020117	
8279	051606	044506	046114	000	
8280	051613	101	052124	046505	EM338: .ASCIZ /ATTEMPTING TO WRITE 18 BIT DATA FIELD WITH WRITE DATA/
8281	051620	052120	047111	020107	

8282	051626	047524	053440	044522	
8283	051634	042524	030440	020070	
8284	051642	044502	020124	040504	
8285	051650	040524	043040	042511	
8286	051656	042114	053440	052111	
8287	051664	020110	051127	052111	
8288	051672	020105	040504	040524	
8289	051700	000			
8290	051701	101	052124	046505	EM339: .ASCII /ATTEMPTING TO WRITE BIT 16-17 OF 18 BIT/
8291	051706	052120	047111	020107	
8292	051714	047524	053440	044522	
8293	051722	042524	041040	052111	
8294	051730	030440	026466	033461	
8295	051736	047440	020106	034061	
8296	051744	041040	052111		
8297	051750	005015	040504	040524	.ASCIZ <15><12>/DATA FIELD WITH WRITE DATA/
8298	051756	043040	042511	042114	
8299	051764	053440	052111	020110	
8300	051772	051127	052111	020105	
8301	052000	040504	040524	000	
8302	052005	101	052124	046505	EM340: .ASCIZ /ATTEMPTING WRITE DATA IN 18 BIT MODE/
8303	052012	052120	047111	020107	
8304	052020	051127	052111	020105	
8305	052026	040504	040524	044440	
8306	052034	020116	034061	041040	
8307	052042	052111	046440	042117	
8308	052050	000105			
8309	052052	051503	020061	047111	EM4000: .ASCIZ /CS1 INCORRECT/
8310	052060	047503	051122	041505	
8311	052066	000124			
8312	052070	042515	051523	040440	EM4001: .ASCIZ /MESS A INCORRECT/
8313	052076	044440	041516	051117	
8314	052104	042522	052103	000	
8315	052111	115	051505	020123	EM4002: .ASCIZ /MESS B INCORRECT/
8316	052116	020102	047111	047503	
8317	052124	051122	041505	000124	
8318	052132	042510	042101	051105	EM4003: .ASCIZ /HEADER WORD 1 INCORRECT/
8319	052140	053440	051117	020104	
8320	052146	020061	047111	047503	
8321	052154	051122	041505	000124	
8322	052162	042510	042101	051105	EM4004: .ASCIZ /HEADER WORD 2 INCORRECT/
8323	052170	053440	051117	020104	
8324	052176	020062	047111	047503	
8325	052204	051122	041505	000124	
8326	052212	051503	020062	047111	EM4005: .ASCIZ /CS2 INCORRECT/
8327	052220	047503	051122	041505	
8328	052226	000124			
8329	052230	051105	047522	020122	EM4006: .ASCIZ /ERROR REG INCORRECT/
8330	052236	042522	020107	047111	
8331	052244	047503	051122	041505	
8332	052252	000124			
8333	052254	052502	020123	042101	EM4007: .ASCIZ /BUS ADDRESS INCORRECT/
8334	052262	051104	051505	020123	
8335	052270	047111	047503	051122	
8336	052276	041505	000124		
8337	052302	047527	042122	041440	EM4008: .ASCIZ /WORD COUNT INCORRECT/

8338	052310	052517	052116	044440	
8339	052316	041516	051117	042522	
8340	052324	052103	000		
8341	052327	103	030523	044440	EM4009: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
8342	052334	041516	051117	042522	
8343	052342	052103	040440	052106	
8344	052350	051105	051040	040505	
8345	052356	044504	043516	042040	
8346	052364	052101	020101	052502	
8347	052372	043106	051105	000	
8348	052377	103	031123	044440	EM4010: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
8349	052404	041516	051117	042522	
8350	052412	052103	040440	052106	
8351	052420	051105	051040	040505	
8352	052426	044504	043516	042040	
8353	052434	052101	020101	052502	
8354	052442	043106	051105	000	
8355	052447	105	031122	051117	EM4011: .ASCIZ /ERROR REG INCORRECT AFTER READING DATA BUFFER/
8356	052454	051040	043505	044440	
8357	052462	041516	051117	042522	
8358	052470	052103	040440	052106	
8359	052476	051105	051040	040505	
8360	052504	044504	043516	042040	
8361	052512	052101	020101	052502	
8362	052520	043106	051105	000	
8363	052525	104	052101	020101	EM4012: .ASCIZ /DATA READ FROM MEMORY INCORRECT/
8364	052532	042522	042101	043040	
8365	052540	047522	020115	042515	
8366	052546	047515	054522	044440	
8367	052554	041516	051117	042522	
8368	052562	052103	000		
8369	052565	115	030522	044440	EM4013: .ASCIZ /MR1 INCORRECT AFTER GAP IN WRITE DATA/
8370	052572	041516	051117	042522	
8371	052600	052103	040440	052106	
8372	052606	051105	043440	050101	
8373	052614	044440	020116	051127	
8374	052622	052111	020105	040504	
8375	052630	040524	000		
8376	052633	104	051511	020113	EM4014: .ASCIZ /DISK ADDRESS REG. INCORRECT/
8377	052640	042101	051104	051505	
8378	052646	020123	042522	027107	
8379	052654	044440	041516	051117	
8380	052662	042522	052103	000	
8381	052667	103	046131	047111	EM4015: .ASCIZ /CYLINDER ADDRESS REG. INCORRECT/
8382	052674	042504	020122	042101	
8383	052702	051104	051505	020123	
8384	052710	042522	027107	044440	
8385	052716	041516	051117	042522	
8386	052724	052103	000		
8387	052727	115	030522	044440	EM4016: .ASCIZ /MR1 INCORRECT/
8388	052734	041516	051117	042522	
8389	052742	052103	000		
8390	052745	105	041503	050040	EM4017: .ASCIZ /ECC PAT REG INCORRECT/
8391	052752	052101	051040	043505	
8392	052760	044440	041516	051117	
8393	052766	042522	052103	000	

N12

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 156
ERROR MESSAGES

SEQ 0156

8394	052773	115	030522	044440
8395	053000	041516	051117	042522
8396	053006	052103	047440	020116
8397	053014	051461	020124	047504
8398	053022	047127	040527	042122
8399	053030	052040	040522	051516
8400	053036	051511	047511	020116
8401	053044	043117	046440	044501
8402	053052	052116	041440	047514
8403	053060	045503	000	
8404	053063	115	030522	044440
8405	053070	041516	051117	042522
8406	053076	052103	047440	020116
8407	053104	047062	020104	047504
8408	053112	047127	040527	042122
8409	053120	052040	040522	051516
8410	053126	052111	047511	020116
8411	053134	043117	046440	044501
8412	053142	052116	041440	047514
8413	053150	045503	000	

EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/

EMW4: .ASCIZ /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/

.SBTTL DATA BUFFERS

.EVEN
NPRBUF:

0414				
0415				
0416	053154			
0417	053154	000	000	.BYTE 0,0
0418	053154	001	001	.BYTE 1,1
0419	053156	002	002	.BYTE 2,2
0420	053160	003	003	.BYTE 3,3
0421	053162	004	004	.BYTE 4,4
0422	053164	005	005	.BYTE 5,5
0423	053166	006	006	.BYTE 6,6
0424	053170	007	007	.BYTE 7,7
0425	053172	010	010	.BYTE 10,10
0426	053174	011	011	.BYTE 11,11
0427	053176	012	012	.BYTE 12,12
0428	053200	013	013	.BYTE 13,13
0429	053202	014	014	.BYTE 14,14
0430	053204	015	015	.BYTE 15,15
0431	053206	016	016	.BYTE 16,16
0432	053210	017	017	.BYTE 17,17
0433	053212	020	020	.BYTE 20,20
0434	053214	021	021	.BYTE 21,21
0435	053216	022	022	.BYTE 22,22
0436	053220	023	023	.BYTE 23,23
0437	053222	024	024	.BYTE 24,24
0438	053224	025	025	.BYTE 25,25
0439	053226	026	026	.BYTE 26,26
0440	053230	027	027	.BYTE 27,27
0441	053232	030	030	.BYTE 30,30
0442	053234	031	031	.BYTE 31,31
0443	053236	032	032	.BYTE 32,32
0444	053240	033	033	.BYTE 33,33
0445	053242	034	034	.BYTE 34,34
0446	053244	035	035	.BYTE 35,35
0447	053246	036	036	.BYTE 36,36
0448	053250	037	037	.BYTE 37,37
0449	053252	040	040	.BYTE 40,40
0450	053254	041	041	.BYTE 41,41
0451	053256	042	042	.BYTE 42,42
0452	053260	043	043	.BYTE 43,43
0453	053262	044	044	.BYTE 44,44
0454	053264	045	045	.BYTE 45,45
0455	053266	046	046	.BYTE 46,46
0456	053270	047	047	.BYTE 47,47
0457	053272	050	050	.BYTE 50,50
0458	053274	051	051	.BYTE 51,51
0459	053276	052	052	.BYTE 52,52
0460	053300	053	053	.BYTE 53,53
0461	053302	054	054	.BYTE 54,54
0462	053304	055	055	.BYTE 55,55
0463	053306	056	056	.BYTE 56,56
0464	053310	057	057	.BYTE 57,57
0465	053312	060	060	.BYTE 60,60
0466	053314	061	061	.BYTE 61,61
0467	053316	062	062	.BYTE 62,62
0468	053320	063	063	.BYTE 63,63
0469	053322			

8470	053324	064	064	.BYTE	64,64
8471	053326	065	065	.BYTE	65,65
8472	053330	066	066	.BYTE	66,66
8473	053332	067	067	.BYTE	67,67
8474	053334	070	070	.BYTE	70,70
8475	053336	071	071	.BYTE	71,71
8476	053340	072	072	.BYTE	72,72
8477	053342	073	073	.BYTE	73,73
8478	053344	074	074	.BYTE	74,74
8479	053346	075	075	.BYTE	75,75
8480	053350	076	076	.BYTE	76,76
8481	053352	077	077	.BYTE	77,77
8482	053354	100	100	.BYTE	100,100
8483	053356	101	101	.BYTE	101,101
8484	053360	102	102	.BYTE	102,102
8485	053362	103	103	.BYTE	103,103
8486	053364	000000		HEAD1: .WORD	000000
8487	053366	140000		.WORD	140000
8488	053370	140000		.WORD	140000
8489	053372	000000		HEAD2: .WORD	000000
8490	053374	040000		.WORD	040000
8491	053376	040000		.WORD	040000
8492	053400	000000		HEAD3: .WORD	000000
8493	053402	100000		.WORD	100000
8494	053404	100000		.WORD	100000
8495	053406	000300		HEAD4: .WORD	000300
8496	053410	040057		.WORD	040057
8497	053412	040356		.WORD	040356
8498	053414	000100		HEAD5: .WORD	000100
8499	053416	040000		.WORD	040000
8500	053420	040100		.WORD	040100
8501	053422	000100		.WORD	000100
8502	053424	140001		.WORD	140001
8503	053426	140101		.WORD	140101
8504	053430	000200		HEAD6: .WORD	000200
8505	053432	140000		.WORD	140000
8506	053434	140000		.WORD	140000
8507	053436	000200		.WORD	000200
8508	053440	140001		.WORD	140001
8509	053442	140201		.WORD	140201
8510	053444	000400		HEAD7: .WORD	000400
8511	053446	140000		.WORD	140000
8512	053450	140000		.WORD	140000
8513	053452	000400		.WORD	000400
8514	053454	040001		.WORD	040001
8515	053456	040401		.WORD	040401
8516	053460	000140		HEAD8: .WORD	000140
8517	053462	040000		.WORD	040000
8518	053464	040140		.WORD	040140
8519	053466	000140		.WORD	000140
8520	053470	140001		.WORD	140001
8521	053472	140101		.WORD	140101
8522	053474	000300		HEAD10: .WORD	000300
8523	053476	140000		.WORD	140000
8524	053500	140000		.WORD	140000
8525	053502	000000		HEAD11: .WORD	000000

CZR6DB0 RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 159
DATA BUFFERS

SEQ 0159

8526	053504	141000				.WORD	141000
8527	053506	141000				.WORD	141000
8528	053510	000000	000000	000000	ECC1:	.WORD	0,0,0,0,0,0,0,0
8529	053516	000000	000000	000000			
8530	053524	000000	000000				
8531	053530	005001			ECC2:	.WORD	005001
8532	053532	040040				.WORD	040040
8533	053534	020004				.WORD	020004
8534	053536	000064				.WORD	000064
8535	053540	000000				.WORD	000000
8536	053542	000000				.WORD	000000
8537	053544	000000				.WORD	000000
8538	053546	000000				.WORD	000000
8539	053550	177777			ECC3:	.WORD	177777
8540	053552	177777				.WORD	177777
8541	053554	177777				.WORD	177777
8542	053556	177777				.WORD	177777
8543	053560	177777				.WORD	177777
8544	053562	177777				.WORD	177777
8545	053564	177777				.WORD	177777
8546	053566	177777				.WORD	177777
8547	053570	177777				.WORD	177777
8548	053572	177777				.WORD	177777
8549	053574	001252			OPI1:	.WORD	001252
8550	053576	141123				.WORD	141123
8551	053600	140371				.WORD	140371
8552	053602	001251				.WORD	001251
8553	053604	141123				.WORD	141123
8554	053606	140372				.WORD	140372
8555	053610	001257				.WORD	001257
8556	053612	141123				.WORD	141123
8557	053614	140374				.WORD	140374
8558	053616	001243				.WORD	001243
8559	053620	141123				.WORD	141123
8560	053622	140360				.WORD	140360
8561	053624	001273				.WORD	001273
8562	053626	141123				.WORD	141123
8563	053630	140350				.WORD	140350
8564	053632	001213				.WORD	001213
8565	053634	141123				.WORD	141123
8566	053636	140330				.WORD	140330
8567	053640	001353				.WORD	001353
8568	053642	141123				.WORD	141123
8569	053644	140270				.WORD	140270
8570	053646	001053				.WORD	001053
8571	053650	141123				.WORD	141123
8572	053652	140170				.WORD	140170
8573	053654	001653				.WORD	001653
8574	053656	141123				.WORD	141123
8575	053660	140770				.WORD	140770
8576	053662	000253				.WORD	000253
8577	053664	141123				.WORD	141123
8578	053666	141370				.WORD	141370
8579	053670	003253				.WORD	003253
8580	053672	141123				.WORD	141123
8581	053674	142370				.WORD	142370

8582	053676	005253	.WORD	005253
8583	053700	141123	.WORD	141123
8584	053702	144370	.WORD	144370
8585	053704	011253	.WORD	011253
8586	053706	141123	.WORD	141123
8587	053710	150370	.WORD	150370
8588	053712	021253	.WORD	021253
8589	053714	141123	.WORD	141123
8590	053716	160370	.WORD	160370
8591	053720	041253	.WORD	041253
8592	053722	141123	.WORD	141123
8593	053724	100370	.WORD	100370
8594	053726	101253	.WORD	101253
8595	053730	141123	.WORD	141123
8596	053732	040370	.WORD	040370
8597	053734	001253	.WORD	001253
8598	053736	141122	.WORD	141122
8599	053740	140371	.WORD	140371
8600	053742	001253	.WORD	001253
8601	053744	141121	.WORD	141121
8602	053746	140372	.WORD	140372
8603	053750	001253	.WORD	001253
8604	053752	141127	.WORD	141127
8605	053754	140374	.WORD	140374
8606	053756	001253	.WORD	001253
8607	053760	141133	.WORD	141133
8608	053762	140360	.WORD	140360
8609	053764	001253	.WORD	001253
8610	053766	141103	.WORD	141103
8611	053770	140350	.WORD	140350
8612	053772	001253	.WORD	001253
8613	053774	141163	.WORD	141163
8614	053776	140330	.WORD	140330
8615	054000	001253	.WORD	001253
8616	054002	141023	.WORD	141023
8617	054004	140270	.WORD	140270
8618	054006	001253	.WORD	001253
8619	054010	141323	.WORD	141323
8620	054012	140170	.WORD	140170
8621	054014	001253	.WORD	001253
8622	054016	141523	.WORD	141523
8623	054020	140770	.WORD	140770
8624	054022	001253	.WORD	001253
8625	054024	140123	.WORD	140123
8626	054026	141370	.WORD	141370
8627	054030	001253	.WORD	001253
8628	054032	143123	.WORD	143123
8629	054034	142370	.WORD	142370
8630	054036	001253	.WORD	001253
8631	054040	145123	.WORD	145123
8632	054042	144370	.WORD	144370
8633	054044	001253	.WORD	001253
8634	054046	151123	.WORD	151123
8635	054050	150370	.WORD	150370
8636	054052	001253	.WORD	001253
8637	054054	161123	.WORD	161123

CZR6DBD RK611 DSKLS CTRL PRTH
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 161
DATA BUFFERS

SEQ 0161

8638	054056	160370	.WORD	160370
8639	054060	001250	.WORD	001250
8640	054062	141123	.WORD	141123
8641	054064	140373	.WORD	140373
8642	054066	141253	.WORD	141253
8643	054070	141123	.WORD	141123
8644	054072	000370	.WORD	000370
8645	054074	000240	OP12: .WORD	000240
8646	054076	140000	.WORD	140000
8647	054100	140240	.WORD	140240
8648	054102	000240	.WORD	000240
8649	054104	140000	.WORD	140000
8650	054106	140240	.WORD	140240
8651	054110	000240	.WORD	000240
8652	054112	140000	.WORD	140000
8653	054114	140240	.WORD	140240
8654	054116	000240	.WORD	000240
8655	054120	140000	.WORD	140000
8656	054122	140240	.WORD	140240
8657	054124	000240	.WORD	000240
8658	054126	140000	.WORD	140000
8659	054130	140240	.WORD	140240
8660	054132	000240	.WORD	000240
8661	054134	140000	.WORD	140000
8662	054136	140240	.WORD	140240
8663	054140	000240	.WORD	000240
8664	054142	140000	.WORD	140000
8665	054144	140240	.WORD	140240
8666	054146	000240	.WORD	000240
8667	054150	140000	.WORD	140000
8668	054152	140240	.WORD	140240
8669	054154	000240	.WORD	000240
8670	054156	140000	.WORD	140000
8671	054160	140240	.WORD	140240
8672	054162	000240	.WORD	000240
8673	054164	140000	.WORD	140000
8674	054166	140240	.WORD	140240
8675	054170	000240	.WORD	000240
8676	054172	140000	.WORD	140000
8677	054174	140240	.WORD	140240
8678	054176	000240	.WORD	000240
8679	054200	140000	.WORD	140000
8680	054202	140240	.WORD	140240
8681	054204	000240	.WORD	000240
8682	054206	140000	.WORD	140000
8683	054210	140240	.WORD	140240
8684	054212	000240	.WORD	000240
8685	054214	140000	.WORD	140000
8686	054216	140240	.WORD	140240
8687	054220	000240	.WORD	000240
8688	054222	140000	.WORD	140000
8689	054224	140240	.WORD	140240
8690	054226	000240	.WORD	000240
8691	054230	140000	.WORD	140000
8692	054232	140240	.WORD	140240
8693	054234	000240	.WORD	000240

8694	054236	140000	.WORD	140000
8695	054240	140240	.WORD	140240
8696	054242	000240	.WORD	000240
8697	054244	140000	.WORD	140000
8698	054246	140240	.WORD	140240
8699	054250	000240	.WORD	000240
8700	054252	140000	.WORD	140000
8701	054254	140240	.WORD	140240
8702	054256	000240	.WORD	000240
8703	054260	140000	.WORD	140000
8704	054262	140240	.WORD	140240
8705	054264	000240	.WORD	000240
8706	054266	140000	.WORD	140000
8707	054270	140240	.WORD	140240
8708	054272	000240	.WORD	000240
8709	054274	140000	.WORD	140000
8710	054276	140240	.WORD	140240
8711	054300	000240	.WORD	000240
8712	054302	140000	.WORD	140000
8713	054304	140240	.WORD	140240
8714	054306	000240	.WORD	000240
8715	054310	140000	.WORD	140000
8716	054312	140240	.WORD	140240
8717	054314	000240	.WORD	000240
8718	054316	140000	.WORD	140000
8719	054320	140240	.WORD	140240
8720	054322	000240	.WORD	000240
8721	054324	140000	.WORD	140000
8722	054326	140240	.WORD	140240
8723	054330	000240	.WORD	000240
8724	054332	140000	.WORD	140000
8725	054334	140240	.WORD	140240
8726	054336	000240	.WORD	000240
8727	054340	140000	.WORD	140000
8728	054342	140240	.WORD	140240
8729	054344	000240	.WORD	000240
8730	054346	140000	.WORD	140000
8731	054350	140240	.WORD	140240
8732	054352	000240	.WORD	000240
8733	054354	140000	.WORD	140000
8734	054356	140240	.WORD	140240
8735	054360	000240	.WORD	000240
8736	054362	140000	.WORD	140000
8737	054364	140040	.WORD	140040
8738	054366	000240	.WORD	000240
8739	054370	140000	.WORD	140000
8740	054372	140040	.WORD	140040
8741	054374	000300	.WORD	000300
8742	054376	140000	.WORD	140000
8743	054400	140300	.WORD	140300
8744	054402	000300	.WORD	000300
8745	054404	140000	.WORD	140000
8746	054406	140300	.WORD	140300
8747	054410	000300	.WORD	000300
8748	054412	140000	.WORD	140000
8749	054414	140300	.WORD	140300

OPI3:

8750	054416	000300	.WORD	000300
8751	054420	140000	.WORD	140000
8752	054422	140300	.WORD	140300
8753	054424	000300	.WORD	000300
8754	054426	140000	.WORD	140000
8755	054430	140300	.WORD	140300
8756	054432	000300	.WORD	000300
8757	054434	140000	.WORD	140000
8758	054436	140300	.WORD	140300
8759	054440	000300	.WORD	000300
8760	054442	140000	.WORD	140000
8761	054444	140300	.WORD	140300
8762	054446	000300	.WORD	000300
8763	054450	140000	.WORD	140000
8764	054452	140300	.WORD	140300
8765	054454	000300	.WORD	000300
8766	054456	140000	.WORD	140000
8767	054460	140300	.WORD	140300
8768	054462	000300	.WORD	000300
8769	054464	140000	.WORD	140000
8770	054466	140300	.WORD	140300
8771	054470	000300	.WORD	000300
8772	054472	140000	.WORD	140000
8773	054474	140300	.WORD	140300
8774	054476	000300	.WORD	000300
8775	054500	140000	.WORD	140000
8776	054502	140300	.WORD	140300
8777	054504	000300	.WORD	000300
8778	054506	140000	.WORD	140000
8779	054510	140300	.WORD	140300
8780	054512	000300	.WORD	000300
8781	054514	140000	.WORD	140000
8782	054516	140300	.WORD	140300
8783	054520	000300	.WORD	000300
8784	054522	140000	.WORD	140000
8785	054524	140300	.WORD	140300
8786	054526	000300	.WORD	000300
8787	054530	140000	.WORD	140000
8788	054532	140300	.WORD	140300
8789	054534	000300	.WORD	000300
8790	054536	140000	.WORD	140000
8791	054540	140300	.WORD	140300
8792	054542	000300	.WORD	000300
8793	054544	140000	.WORD	140000
8794	054546	140300	.WORD	140300
8795	054550	000300	.WORD	000300
8796	054552	140000	.WORD	140000
8797	054554	140300	.WORD	140300
8798	054556	000300	.WORD	000300
8799	054560	140000	.WORD	140000
8800	054562	140300	.WORD	140300
8801	054564	000300	.WORD	000300
8802	054566	140000	.WORD	140000
8803	054570	140300	.WORD	140300
8804	054572	000300	.WORD	000300
8805	054574	140000	.WORD	140000

CZR6080 RK611 DSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 164
DATA BUFFERS

SEQ 0164

8806	054576	140300	.WORD	140300
8807	054600	000300	.WORD	000300
8808	054602	140000	.WORD	140000
8809	054604	140300	.WORD	140300
8810	054606	000300	.WORD	000300
8811	054610	140000	.WORD	140000
8812	054612	140300	.WORD	140300
8813	054614	000300	.WORD	000300
8814	054616	140000	.WORD	140000
8815	054620	140300	.WORD	140300
8816	054622	000300	.WORD	000300
8817	054624	140000	.WORD	140000
8818	054626	140300	.WORD	140300
8819	054630	000300	.WORD	000300
8820	054632	140000	.WORD	140000
8821	054634	140300	.WORD	140300
8822	054636	000300	.WORD	000300
8823	054640	140000	.WORD	140000
8824	054642	140300	.WORD	140300
8825	054644	000300	.WORD	000300
8826	054646	140000	.WORD	140000
8827	054650	140300	.WORD	140300
8828	054652	000300	.WORD	000300
8829	054654	140000	.WORD	140000
8830	054656	140300	.WORD	140300
8831	054660	000300	.WORD	000300
8832	054662	140000	.WORD	140000
8833	054664	140200	.WORD	140200
8834	054666	000300	.WORD	000300
8835	054670	140000	.WORD	140000
8836	054672	140300	.WORD	140300
8837	054674	000040	.WORD	000040
8838	054676	140000	.WORD	140000
8839	054700	140000	.WORD	140000
8840	054702	000040	.WORD	000040
8841	054704	140000	.WORD	140000
8842	054706	140040	.WORD	140040
8843	054710	000040	.WORD	000040
8844	054712	140000	.WORD	140000
8845	054714	140040	.WORD	140040
8846	054716	000040	.WORD	000040
8847	054720	140000	.WORD	140000
8848	054722	140040	.WORD	140040
8849	054724	000040	.WORD	000040
8850	054726	140000	.WORD	140000
8851	054730	140040	.WORD	140040
8852	054732	000040	.WORD	000040
8853	054734	140000	.WORD	140000
8854	054736	140040	.WORD	140040
8855	054740	000040	.WORD	000040
8856	054742	140000	.WORD	140000
8857	054744	140040	.WORD	140040
8858	054746	000040	.WORD	000040
8859	054750	140000	.WORD	140000
8860	054752	140040	.WORD	140040
8861	054754	000040	.WORD	000040

OPI4:

8862	054756	140000	.WORD	140000
8863	054760	140040	.WORD	140040
8864	054762	000040	.WORD	000040
8865	054764	140000	.WORD	140000
8866	054766	140040	.WORD	140040
8867	054770	000040	.WORD	000040
8868	054772	140000	.WORD	140000
8869	054774	140040	.WORD	140040
8870	054776	000040	.WORD	000040
8871	055000	140000	.WORD	140000
8872	055002	140040	.WORD	140040
8873	055004	000040	.WORD	000040
8874	055006	140000	.WORD	140000
8875	055010	140040	.WORD	140040
8876	055012	000040	.WORD	000040
8877	055014	140000	.WORD	140000
8878	055016	140040	.WORD	140040
8879	055020	000040	.WORD	000040
8880	055022	140000	.WORD	140000
8881	055024	140040	.WORD	140040
8882	055026	000040	.WORD	000040
8883	055030	140000	.WORD	140000
8884	055032	140040	.WORD	140040
8885	055034	000040	.WORD	000040
8886	055036	140000	.WORD	140000
8887	055040	140040	.WORD	140040
8888	055042	000040	.WORD	000040
8889	055044	140000	.WORD	140000
8890	055046	140040	.WORD	140040
8891	055050	000040	.WORD	000040
8892	055052	140000	.WORD	140000
8893	055054	140040	.WORD	140040
8894	055056	000040	.WORD	000040
8895	055060	140000	.WORD	140000
8896	055062	140040	.WORD	140040
8897	055064	000040	.WORD	000040
8898	055066	140000	.WORD	140000
8899	055070	140040	.WORD	140040
8900	055072	000040	.WORD	000040
8901	055074	140000	.WORD	140000
8902	055076	140040	.WORD	140040
8903	055100	000040	.WORD	000040
8904	055102	140000	.WORD	140000
8905	055104	140040	.WORD	140040
8906	055106	000040	.WORD	000040
8907	055110	140000	.WORD	140000
8908	055112	140040	.WORD	140040
8909	055114	000040	.WORD	000040
8910	055116	140000	.WORD	140000
8911	055120	140040	.WORD	140040
8912	055122	000040	.WORD	000040
8913	055124	140000	.WORD	140000
8914	055126	140040	.WORD	140040
8915	055130	000040	.WORD	000040
8916	055132	140000	.WORD	140000
8917	055134	140040	.WORD	140040

8918	055136	000040				.WORD	000040
8919	055140	140000				.WORD	140000
8920	055142	140040				.WORD	140040
8921	055144	000040				.WORD	000040
8922	055146	140000				.WORD	140000
8923	055150	140040				.WORD	140040
8924	055152	000040				.WORD	000040
8925	055154	140000				.WORD	140000
8926	055156	140040				.WORD	140040
8927	055160	000040				.WORD	000040
8928	055162	140000				.WORD	140000
8929	055164	140040				.WORD	140040
8930	055166	000040				.WORD	000040
8931	055170	140000				.WORD	140000
8932	055172	140040				.WORD	140040
8933	055174	C.253			VRCHDR:	.WORD	001253
8934	055176	141123				.WORD	141123
8935	055200	140370				.WORD	140370
8936	055202	177777	177777	177777	MOD2BF:	.WORD	177777,177777,177777,177777,177777,177777
8937	055210	177777	177777	177777			
8938	055216	000400			BUFF:	.BLKW	400
8939	056216	125252	125252	125252	ECCBUF:	.WORD	125252,125252,125252,125252,125252,125252
8940	056224	125252	125252	125252			
8941	056232	125252	125252	177777		.WORD	125252,125252,177777,177777,177777,177777
8942	056240	177777	177777	177777			
8943	056246	177777	177777	177777		.WORD	177777,177777,177777,177777,000000,000000
8944	056254	177777	000000	000000			
8945	056262	000000	000000	000000		.WORD	000000,000000,000000,000000,000000,000000
8946	056270	000000	000000	000000			
8947	056276	052525	052525	052525		.WORD	052525,052525,052525,052525,052525,052525
8948	056304	025252	052525	052525			
8949	056312	052525	052525	121105		.WORD	052525,052525,121105,150442,064221,132110
8950	056320	150442	064221	132110			
8951	056326	055044	026422	013211		WORD	055044,026422,013211,105504,042642,021321
8952	056334	105504	042642	021321			
8953	056342	110550	044264	022132		.WORD	110550,044264,022132,011055,104426,042213
8954	056350	011055	104426	042213			
8955	056356	133333	066666	155555		.WORD	133333,066666,155555,155555,133333,066666
8956	056364	155555	133333	066666			
8957	056372	066666	155555	155555		.WORD	066666,155555,155555,133333,133333,133333
8958	056400	133333	133333	133333			
8959	056406	133333	133333	133333		.WORD	133333,133333,133333,133333,177777,177777
8960	056414	133333	177777	177777			
8961	056422	177777	052525	052525		.WORD	177777,052525,052525,052525,177777,177777
8962	056430	052525	177777	177777			
8963	056436	052525	052525	177777		.WORD	052525,052525,177777,052525,177252,177252
8964	056444	052525	177252	177252			
8965	056452	172765	172765	072307		.WORD	172765,172765,072307,135143,156461,167230
8966	056460	135143	156461	167230			
8967	056466	073514	035646	016723		.WORD	073514,035646,016723,107351,143564,061672
8968	056474	107351	143564	061672			
8969	056502	030735	114356	046167		.WORD	030735,114356,046167,123073,151453,164616
8970	056510	123073	151453	164616			
8971	056516	125252	125252	125252		.WORD	125252,125252,125252,125252,125252,125252
8972	056524	125252	125252	125252			
8973	056532	125252	125252	177777		.WORD	125252,125252,177777,177777,177777,177777

8974	056540	177777	177777	177777		
8975	056546	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8976	056554	177777	000000	000000		
8977	056562	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8978	056570	000000	000000	000000		
8979	056576	052525	052525	052525	.WORD	052525,052525,052525,052525,052525,052525
8980	056604	025252	052525	052525		
8981	056612	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8982	056620	150442	064221	132110		
8983	056626	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8984	056634	105504	042642	021321		
8985	056642	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8986	056650	011055	104426	042213		
8987	056656	133333	066666	155555	.WORD	133333,066666,155555,155555,133333,066666
8988	056664	155555	133333	066666		
8989	056672	066666	155555	155555	.WORD	066666,155555,155555,133333,133333,133333
8990	056700	133333	133333	133333		
8991	056706	133333	133333	133333	.WORD	133333,133333,133333,133333,177777,177777
8992	056714	133333	177777	177777		
8993	056722	177777	052525	052525	.WORD	177777,052525,052525,052525,177777,177777
8994	056730	052525	177777	177777		
8995	056736	052525	052525	177777	.WORD	052525,052525,177777,052525,177252,177252
8996	056744	052525	177252	177252		
8997	056752	172765	172765	072307	.WORD	172765,172765,072307,135143,156461,167230
8998	056760	135143	156461	167230		
8999	056766	073514	035646	016723	.WORD	073514,035646,016723,107351,143564,061672
9000	056774	107351	143564	061672		
9001	057002	030735	114356	046167	.WORD	030735,114356,046167,123073,151453,164616
9002	057010	123073	151453	164616		
9003	057016	125252	125252	125252	.WORD	125252,125252,125252,125252,125252,125252
9004	057024	125252	125252	125252		
9005	057032	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
9006	057040	177777	177777	177777		
9007	057046	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
9008	057054	177777	000000	000000		
9009	057062	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
9010	057070	000000	000000	000000		
9011	057076	052525	052525	052525	.WORD	052525,052525,052525,052525,052525,052525
9012	057104	025252	052525	052525		
9013	057112	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
9014	057120	150442	064221	132110		
9015	057126	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
9016	057134	105504	042642	021321		
9017	057142	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
9018	057150	011055	104426	042213		
9019	057156	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
9020	057164	177777	177777	177777		
9021	057172	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
9022	057200	177777	177777	177777		
9023	057206	177777	177777	177777	.WORD	177777,177777,177777,177777
9024	057214	177777				
9025	057216	177777	177777	177777	BADPAR: .WORD	177777,177777,177777,177777,177777,177777
9026	057224	177777	177777	177777		
9027		000001			.END	

CZP6DB0 RK611 DSKLS CTRL PRTH
CZP6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 169
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0168

ABASE = 177440	1024#	1333	1374	
ABORT = 043754	6515	7742#		
ACDW1 = 000000	1333	1376		
ACDW2 = 000000	1333	1377		
ACLO = 000010	1119#			
ACPUOP = 000000	1333	1348		
ADDW0 = 000000	1333			
ADDW1 = 000000	1333			
ADDW10 = 000000	1333			
ADDW11 = 000000	1333			
ADDW12 = 000000	1333			
ADDW13 = 000000	1333			
ADDW14 = 000000	1333			
ADDW15 = 000000	1333			
ADDW2 = 000000	1333			
ADDW3 = 000000	1333			
ADDW4 = 000000	1333			
ADDW5 = 000000	1333			
ADDW6 = 000000	1333			
ADDW7 = 000000	1333			
ADDW8 = 000000	1333			
ADDW9 = 000000	1333			
ADEVCT = 000000	1333	1339		
ADEVN = 000000	1333	1375		
AENV = 000000	1333	1344		
AENVN = 000000	1333	1345		
AFATAL = 000000	1333	1336		
AMADR1 = 000000	1333	1361		
AMADR2 = 000000	1333	1365		
AMADR3 = 000000	1333	1368		
AMADR4 = 000000	1333	1371		
AMAMS1 = 000000	1333	1355		
AMAMS2 = 000000	1333	1363		
AMAMS3 = 000000	1333	1366		
AMAMS4 = 000000	1333	1369		
AMSGAD = 000000	1333	1341		
AMSGLG = 000000	1333	1342		
AMSGTY = 000000	1333	1335		
AMTYP1 = 000000	1333	1356		
AMTYP2 = 000000	1333	1364		
AMTYP3 = 000000	1333	1367		
AMTYP4 = 000000	1333	1370		
APASS = 000000	1333	1338		
APRIOR = 000240	1023#	1333		
APTCU = 000040	6788#	6874		
APTENV = 000001	6744	6786#	6820	6867
APTSIZ = 000200	2226	6785#		
APTSPO = 000100	6746	6787#	6869	
ASWREG = 000000	1333	1346		
ATESTN = 000000	1333	1337		
AUNIT = 000000	1333	1340		
AUSWR = 000000	1333	1347		
AVECT1 = 000210	1022#	1333	1372	
AVECT2 = 000000	1333	1373		
BADPAR = 057216	5777	5791	5903	9025#
BAI = 000020	1082#			

CZR6DBO RK611 DSKLS CTRL PRT4
CZR6DB.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 170
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0169

BA16 = 000400	1067*													
BA17 = 001000	1068*													
BITCNT 003262	2157*	2941*	2947*	2955*	2971*	4965*	4969*	4975*	4997*	5077*	5081*	5087*	5111*	
	5191*	5195*	5201*	5225*	5297*	5302*	5311*	5332*	5343*	5359*	5370*	5378*	5490*	
	5495*	5504*	5525*	5540*	5556*	5567*	5575*	5694*	5698*	5706*	5729*	5843*	5847*	
	5855*	5878*	5983*	5987*	5995*	6018*	6052*	6067*	6088*	6094*	6102*	7503	7533	
	1003*	1064	1097	1116										
BIT0 = 000001	993*	1003												
BIT00 = 000001	992*	1002												
BIT01 = 000002	991*	1001												
BIT02 = 000004	990*	1000												
BIT03 = 000010	989*	999												
BIT04 = 000020	988*	998												
BIT05 = 000040	987*	997												
BIT06 = 000100	986*	996												
BIT07 = 000200	985*	995	6350											
BIT08 = 000400	984*	994	6368	6831										
BIT09 = 001000	1002*	1098												
BIT1 = 000002	983*	1069	1088	1107	1140	1155	1211	1226	6610	6612	6808			
BIT10 = 002000	982*	1070	1089	1108	1125	1141	1156	1212	1227	6375				
BIT11 = 004000	981*	1072	1090	1109	1142	1213								
BIT12 = 010000	980*	1073	1091	1110	1126	1143	1214	6606	6608	6815				
BIT13 = 020000	979*	1074	1092	1111	1127	1144	1215	6336						
BIT14 = 040000	978*	1075	1076	1093	1112	1128	1145	1216						
BIT15 = 100000	1001*	1099	1118											
BIT2 = 000004	1000*	1081	1100	1119										
BIT3 = 000010	999*	1082	1101	1120	1134	1149	1220	6606	6608					
BIT4 = 000020	998*	1083	1102	1121	1135	1150	1221							
BIT5 = 000040	997*	1065	1084	1103	1122	1136	1151	1207	1222					
BIT6 = 000100	996*	1066	1085	1104	1123	1137	1152	1208	1223					
BIT7 = 000200	995*	1067	1086	1105	1124	1138	1153	1209	1224	6610	6612			
BIT8 = 000400	994*	1068	1087	1106	1139	1154	1210	1225	6575					
BIT9 = 001000	1010*													
BPTVEC = 000014	1104*	3384	3466	4064	4608									
BSE = 000200	2329	2375	2422	2468	2514	2561	2619	2676	2730	2785	2928	3019	3098	
BUFF = 055216	3179	3260	3339	3421	3499	3620	3703	3790	3893	3997	4101	4206	4332	
	4451	4555	4645	4735	4825	8938*								
CALHDR = 036006	2615	2672	2726	2781	6555*									
CCLR = 100000	1076*	2326	2349	2372	2395	2419	2442	2465	2488	2511	2534	2558	2581	
	2616	2634	2673	2691	2727	2745	2782	2800	2832	2855	2885	2925	3013	
	3093	3174	3255	3334	3416	3494	3575	3608	3698	3788	3891	3995	4099	
	4201	4282	4327	4408	4446	4527	4553	4618	4643	4708	4733	4798	4823	
	4888	4916	5028	5142	5249	5442	5636	5785	5925	6307				
CDT = 002000	1069*	2333	2343	2379	2389	2426	2436	2472	2482	2518	2528	2565	2575	
	3600	3624												
CERR = 100000	1075*	3571	4278	4404	4523	4606	4696	4786	4876					
CFMT = 010000	1072*	2333	2343	2379	2389	2426	2436	2472	2482	2518	2528	2565	2575	
	2810	2812	3503	3512	3600	3624	5644	5793	5933	6573				
CKSWR = 104407	6335	6804	6830	7493*										
CLEAR = 000005	1049*													
COE = 001000	1106*													
CR = 000015	918*	6913	6923											
CRLF = 000200	919*	2251	6884	6923										
CTO = 004000	1070*													
CTRHLT = 034570	6306*	7109	7192											
CYLN = 003300	2165*	3086*	3095	3167*	3176	3248*	3257	3309*	3310	6527	6540			

7663

[illegible]

7671	7905#											
7673	7909#											
7681	7913#											
7683	7917#											
7695	7921#											
7697	7932#											
7707	7943#											
1074#												
1306#	2214*	2222*	6390*	6807*								
1164#	1235#	2222										
1093#												
1135#	2327	2336	2337	2373	2382	2383	2420	2429	2430	2466	2475	2476
2512	2521	2522	2559	2568	2569	2617	2624	2625	2674	2681	2682	2728
2735	2736	2783	2790	2791	2833	2838	2839	2926	2932	2933	2936	2937
2983	3017	3023	3024	3028	3029	3030	3033	3065	3094	3102	3103	3106
3107	3175	3183	3184	3187	3188	3256	3264	3265	3268	3269	3335	3343
3344	3347	3348	3417	3425	3426	3429	3430	3498	3506	3507	3515	3517
3518	3612	3626	3627	3630	3631	3699	3707	3708	3711	3712	3789	3797
3798	3806	3808	3809	3857	3892	3900	3901	3909	3911	3912	3960	3996
4004	4005	4013	4015	4016	4100	4108	4109	4117	4119	4120	4205	4213
4214	4222	4224	4225	4331	4339	4340	4348	4350	4351	4450	4458	4459
4467	4469	4470	4554	4562	4563	4566	4567	4599	4600	4644	4652	4653
4656	4657	4689	4690	4734	4742	4743	4746	4747	4779	4780	4824	4832
4833	4836	4837	4869	4870	4917	4923	4924	4927	4928	4966	5029	5035
5036	5039	5040	5078	5143	5149	5150	5153	5154	5192	5250	5256	5257
5260	5261	5299	5382	5383	5443	5449	5450	5453	5454	5492	5579	5580
5641	5647	5648	5652	5653	5695	5790	5796	5797	5801	5802	5844	5930
5936	5937	5941	5942	5984	6620	6637	6638	6651	6652	6678	6679	6691
6692	6718	6721	6722	6723	6724	6727	6728	6729	6730			
1116#												
1123#												
1121#												
1100#												
1080#												
1127#												
924#	1305	2213										
1109#												
1102#												
1398	7502#											
1403	7503#											
1409	1415	1421	1427	1433	1439	1445	1451	1457	1463	1469	1475	1481
1487	1493	1499	1505	1511	1518	1525	1532	1539	1546	1553	1560	1567
1574	1581	1588	1595	7506#								
1601	1607	1613	1619	1625	7509#							
1631	1637	1643	1710	1716	1722	1729	173					

[illegible]

E14

CZR6080 RK611 CSKLS CTRL PRT4
CZR608.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 174
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0173

EM334	051345	5369	5566	6093	8250*											
EM335	051403	2005	2011	2017	2023	2029	2035	2041	8256*							
EM336	051457	2047	8264*													
EM337	051534	2053	2059	2065	2071	2077	2083	2089	8272*							
EM338	051613	5710	5859	5999	6032	8280*										
EM339	051701	5736	5885	6025	6063	8290*										
EM340	052005	2095	2101	8302*												
EM4000	052052	1408	1426	1444	1462	1480	1498	1517	1538	1559	1580	1600	1654	1709		
		1728	1748	1766	1790	1814	1838	1862	1886	1910	1934	1970	1988	2006		
		2048	2054	8309*												
EM4001	052070	1414	1432	1450	1468	1486	1504	8312*								
EM4002	052111	1420	1438	1456	1474	1492	1510	8315*								
EM4003	052132	1524	1545	1566	1587	8318*										
EM4004	052162	1531	1552	1573	1594	8322*										
EM4005	052212	1606	1715	1735	1754	1772	1796	1820	1844	1868	1892	1916	1940	1976		
		1994	2012	2060	8326*											
EM4006	052230	1612	1721	1742	1760	1778	1802	1826	1850	1874	1898	1922	1946	1982		
		2000	2018	2066	8329*											
EM4007	052254	1618	2024	2072	8333*											
EM4008	052302	1624	2030	2078	8337*											
EM4009	052327	1630	8341*													
EM4010	052377	1636	8348*													
EM4011	052447	1642	8355*													
EM4012	052525	1648	8363*													
EM4013	052565	1660	1784	1808	1832	1856	1880	1904	1928	1952	8369*					
EM4014	052633	1666	1678	1690	2042	2090	8376*									
EM4015	052667	1672	1684	1696	2036	2084	8381*									
EM4016	052727	1733	8387*													
EM4017	052745	1958	1964	2096	2102	8390*										
ERRCNT	003250	2152*	2230*	6510*	6513											
ERRVEC	000004	1006*	2211	2212*	2223*	6251*	6252*	6279*	6280*	6341	6342*	6344*	6347*			
E.ASOF	003216	2134*														
E.BA	003204	2129*	2850*	2861	5398*	5409	5595*	5606	7509	7538						
E.CS1	003200	2127*	2343*	2346	2389*	2392	2436*	2439	2482*	2485	2528*	2531	2575*	2578		
		2610*	2622	2631	2667*	2679	2688	2721*	2733	2742	2776*	2788	2797	2810		
		2812*	2847*	2852	2882	2938*	2945	2953	2969	3382*	3385	3464*	3467	3512*		
		3553	3570*	3571*	3579*	3580	3600*	3665	3746*	3749	3803*	3843	3906*	3946		
		4010*	4050	4114*	4154	4219*	4260	4277*	4278*	4286*	4287	4345*	4386	4403*		
		4404*	4412*	4413	4464*	4505	4522*	4523*	4531*	4532	4606*	4609	4622*	4623		
		4696*	4699	4712*	4713	4786*	4789	4802*	4803	4876*	4879	4892*	4893	5389*		
		5400	5586*	5597	657?	7506	7509	7513	7518	7524	7528	7535				
E.CS2	003210	2131*	2848*	2858	2873*	2888	2900*	3383*	3388	3465*	3470	3513*	3556	3601*		
		3668	3747*	3752	3804*	3846	3907*	3949	4011*	4053	4115*	4157	4220*	4263		
		4346*	4389	4465*	4508	4607*	4612	4697*	4702	4787*	4792	4877*	4882	5390*		
		5403	5587*	5600	7509	7513	7524	7528	7535							
E.DA	003206	2130*	2609*	2621	2666*	2678	2701*	2702	2720*	2732	2755*	2756	2775*	2787		
		3140	3221	3302	5397*	5418	5594*	5615	6541*	6542*	6543	6545*	6546*	6547		
		6549*	6564	6571	7522	7538										
E.DB	003222	2136*	2874*	2894	7516											
E.DCYL	003220	2135*	2608*	2620	2644*	2645	2647*	2648	2665*	2677	2719*	2731	2774*	2786		
		3143	3224	3305	5396*	5415	5593*	5612	6540*	6550*	6556	7522	7538			
E.DS	003212	2132*														
E.ECPS	003232	2140*														
E.ECPT	003234	2141*	4977*	4994	5108	5222	5329	5522	5726	5875	6015	6049	6105*	6107		
		6614*	6615*	7533	7541											
E.EP	003214	2133*	2851*	2867	2891	3384*	3391	3466*	3473	3514*	3559	3569*	3602*	3671		

		3748*	3755	3805*	3849	3908*	3952	4012*	4056	4064*	4116*	4160	4168*	4221*
		4266	4276*	4347*	4392	4402*	4466*	4511	4521*	4608*	4615	4698*	4705	4788*
		4795	4878*	4885	5391*	5406	5588*	5603	7509	7513	7524	7528	7535	
E.MR1	003224	2137*	2983*	2984	3065*	3066	3515*	3563	3806*	3853	3857*	3909*	3956	3960*
		4013*	4060	4117*	4164	4222*	4270	4348*	4396	4467*	4515	4966*	5078*	5192*
		5299*	5337*	5364*	5492*	5530*	5561*	5695*	5844*	5984*	6055*	6087*	6620*	6621*
		6626*	6631*	6636*	6640	6650*	6654	6672*	6677*	6681	6690*	6694	7503	7520
		7531												
E.MR2	003226	2138*	2344*	2352	2390*	2398	2437*	2445	2483*	2491	2529*	2537	2576*	2584
		2637	2694	2748	2803	6563*	7506							
E.MR3	003230	2139*	2345*	2355	2391*	2401	2438*	2448	2484*	2494	2530*	2540	2577*	2587
		2640	2697	2751	2806	6584*	7506							
E.SPAR	003236	2142*												
E.WC	003202	2128*	2849*	2864	5399*	5412	5596*	5609	7509	7538				
FMTE	= 000020	1101*												
GNS	= ***** U	1163	1234	2250	6134	6141	7485	7486	7487	7488	7489	7491	7493	7494
		7495	7496	7497	7498	7499								
GO	= 000001	1064*	3570	4277	4403	4522	4606	4696	4786	4876	5389	5586		
GTSWR	= 104406	2245	7491*											
MDPCNT	003310	2167*	3516*	3519	3566*	3567	3807*	3856*	3910*	3959*	4014*	4063*	4118*	4167*
		4223*	4226	4273*	4274	4349*	4352	4399*	4400	4468*	4471	4518*	4519	7528
		7531												
HEADER	003302	2166*	3116	3197	3278	6527*	6528*	6529*	6530*	6531*	6532*	6533*	6534*	6535
		6536*	6537*	6538	6539*									
HEAD1	053364	2956	3042	4938	5050	5164	5270	5463	8486*					
HEAD10	053474	4666	4756	4846	8522*									
HEAD11	053502	5666	5815	5955	8525*									
HEAD2	053372	3357	4576	8489*										
HEAD3	053400	3439	8492*											
HEAD4	053406	3721	8495*											
HEAD5	053414	3802	8498*											
HEAD6	053430	3905	8504*											
HEAD7	053444	4009	8510*											
HEAD8	053460	4113	8516*											
HT	= 000011	916*	6882	6923										
HVRC	= 000400	1105*	3602	3748	4168	4276	4402	4521	4698	4788	4878			
IDAE	= 002000	1107*												
IE	= 000100	1065*												
ILF	= 000001	1097*												
INCHOR	035624	3092	3173	3254	6527*									
INTR	= 000300	1060*												
IOTVEC	= 000020	1011*	2195*	2196*										
IR	= 000100	1084*	2873	2900	3383	3465	3513	3601	3747	3804	3907	4011	4115	4220
		4346	4465	4607	4697	4787	4877	5390	5587					
		917*	6917	6923										
LF	= 000012	1138*	1209*	2336	2382	2429	2475	2521	2568	2624	2681	2735	2790	2838
MCLK	= 000400	2932	3023	3029	3102	3183	3264	3343	3425	3506	3626	3707	3797	3900
		4004	4108	4213	4339	4458	4562	4599	4652	4689	4742	4779	4832	4869
		4923	5035	5149	5256	5382	5449	5579	5647	5796	5936	6637	6651	6678
		6691	6718	6721	6723	6727	6729							
		1087*												
MDS	= 001000	1019*	6181	6193	6219	6254								
MEMBAS	= 172100	1182	1253	6229	6290*									
MEMERR	034526	2162*	5763	6182	6194	6220	6253*	6259*	6263*	6274*	6281			
MEMPAR	003274	1018*	1181	1252	6202*	6229*	6230*	6283*	6284*					
MEMVEC	= 000114	1139*	1210*	6721	6729									
MERC	= 001000													

CZP6080 RK611 DSKLS CTRL PRTH
CZP608.P11 02-DEC-77 10:00MPCY11 30(1046) 02-DEC-77 10:20 PAGE 176
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0175

MESMSK= 000017
MEWD = 002000
MIND = 000200
MOD2BF 055202
MSP = 000100

M1.BIT 003256

M2.BIT 003260

NED = 010000
NEM = 004000
NEWPAS 004316
NPRBUF 053154
NXF = 000004
OFFSET= 000015
OFST = 000004
OPI = 020000
OPI1 053574
OPI2 054074
OPI3 054374
OPI4 054674
OPR001 043522
OPR002 043551
OPR003 043557
OPR004 043607
OPR005 043631
OPR006 043671
OR = 000200PACK = 000003
PARCHK 034366
PARM 003314
PAR.EN= 000001
PAT = 000020
PCA = 004000
PCD = 010000
PGE = 002000
PIP = 020000
PIRQ = 177772
PIRQVE= 000240
PR.BIT 003254

1132*	1211*	2983	3065	3515	3806	3857	3909	3960	4013	4117	4222	4348
1140*	4966	5078	5192	5299	5492	5695	5844	5984	6620	6626	6650	6690
4467	1137*	3028	3029	3030								
5642	5707	5856	8936*									
1136*	1207*	2936	3106	3187	3268	3347	3429	3517	3630	3711	3808	3911
4015	4119	4224	4350	4469	4566	4656	4746	4836	4927	5039	5153	5260
5453	5652	5801	5941									
2155*	2940*	2960*	2977*	3035*	3046*	3059*	3109*	3120*	3133*	3190*	3201*	3214*
3271*	3282*	3295*	3350*	3361*	3374*	3432*	3443*	3456*	3522*	3532*	3545*	3633*
3644*	3657*	3714*	3725*	3738*	3811*	3821*	3835*	3914*	3924*	3938*	4018*	4028*
4042*	4122*	4132*	4146*	4229*	4239*	4252*	4355*	4365*	4378*	4474*	4484*	4497*
4569*	4580*	4593*	4659*	4670*	4683*	4749*	4760*	4773*	4839*	4850*	4863*	4930*
4941*	4954*	4963*	4982	4983*	5042*	5053*	5066*	5075*	5095	5096*	5156*	5167*
5180*	5189*	5209	5210*	5263*	5274*	5287*	5294*	5316	5317*	5348	5349*	5372
5373*	5456*	5467*	5480*	5487*	5509	5510*	5545	5546*	5569	5570*	5656*	5670*
5683*	5692*	5712	5713*	5805*	5819*	5832*	5841*	5861	5862*	5945*	5959*	5972*
5981*	6001	6002*	6035	6036*	6072	6073*	6096	6097*	6624	6670	6675	6716
7503												
2156*	4964*	4982*	5076*	5095*	5190*	5209*	5295*	5316*	5348*	5372*	5488*	5509*
5545*	5569*	5693*	5712*	5842*	5861*	5982*	6001*	6035*	6072*	6096*	6629	6634
7503												
1090*												
1089*												
2307*												
2835	2850	2870	8417*									
1099*												
1053*												
1118*												
1110*	3569	4276	4402	4521								
3511	8549*											
4218	8645*											
4344	8741*											
4463	8837*											
2257	7711*											
2260	2271	2287	7715*									
2267	7717*											
2278	7722*											
6308	7726*											
2254	7732*											
1085*	2848	2873	3383	3465	3513	3601	3747	3804	3907	4011	4115	4220
4346	4465	4607	4697									
1048*												
2307	6251*	7446										
1170	1241	2171*										
1020*	6198	6224	6257									
1134*												
1141*	1212*	6621	6636	6672								
1142*	1213*	6621	6631	6677								
1088*												
1126*												
923*												
1017*												
2154*	2939*	2950*	2960	2963*	2966*	2977	2978*	3034*	3040*	3046	3049*	3052*
3059	3060*	3108*	3114*	3120	3123*	3126*	3133	3134*	3189*	3195*	3201	3204*
3207*	3214	3215*	3270*	3276*	3282	3285*	3288*	3295	3296*	3349*	3355*	3361

CZRB080 RK611 DSKLS CTRL PRY4
CZRB08.P11 02-DEC-77 10:00

MACY11 30(1046) 02-DEC-77 10:20 PAGE 177
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0176

PRO = 000000
PR1 = 000040
PR2 = 000100
PR3 = 000140
PR4 = 000200
PR5 = 000240
PR6 = 000300
PR7 = 000340
PS = 177776
PSW = 177776
PWARVEC = 000024
P1.BIT = 003252

RDBIT 037026

RDCHR = 104410
RDATA = 000021
ROGATE = 100000
RDHEAD = 000025
RDLIN = 104411
RDOCT = 104412
RDY = 000200

RECAL = 000013
RESREG = 104414
RESTR = 003324
RESVEC = 000010
RKASOF = 000016
RKBA = 000004

3364*	3367*	3374	3375*	3431*	3437*	3443	3446*	3449*	3456	3457*	3521*	3527*
3532	3535*	3538*	3545	3546*	3632*	3638*	3644	3647*	3650*	3657	3658*	3713*
3719*	3725	3728*	3731*	3738	3739*	3810*	3816*	3821	3824*	3827*	3835	3836*
3913*	3919*	3924	3927*	3930*	3938	3939*	4017*	4023*	4028	4031*	4034*	4042
4043*	4121*	4127*	4132	4135*	4138*	4146	4147*	4228*	4234*	4239	4242*	4245*
4252	4253*	4354*	4360*	4365	4368*	4371*	4378	4379*	4473*	4479*	4484	4487*
4490*	4497	4498*	4568*	4574*	4580	4583*	4586*	4593	4594*	4658*	4664*	4670
4673*	4676*	4683	4684*	4748*	4754*	4760	4763*	4766*	4773	4774*	4838*	4844*
4850	4853*	4856*	4863	4864*	4929*	4935*	4941	4944*	4947*	4954	4955*	4962*
4983	4984*	5041*	5047*	5053	5056*	5059*	5066	5067*	5074*	5096	5097*	5155*
5161*	5167	5170*	5173*	5180	5181*	5188*	5210	5211*	5262*	5268*	5274	5277*
5280*	5287	5288*	5293*	5317	5318*	5349	5350*	5373	5374*	5455*	5461*	5467
5470*	5473*	5480	5481*	5486*	5510	5511*	5546	5547*	5570	5571*	5655*	5663*
5670	5673*	5676*	5683	5684*	5691*	5713	5714*	5804*	5812*	5819	5822*	5825*
5832	5833*	5840*	5862	5863*	5944*	5952*	5959	5962*	5965*	5972	5973*	5980*
6002	6003*	6036	6037*	6073	6074*	6097	6098*	6622	6714	7503		
940*	2308	6231										
941*												
942*												
943*												
944*												
945*	1023	2146										
946*												
947*	1183	1254	2182	6230	6252	7429	7443					
920*	921											
921*												
1012*	2201*	2202*	7428*	7429*	7442*	7443*						
2153*	4961*	4972*	4984	4987*	4990*	5073*	5084*	5097	5100*	5103*	5187*	5198*
5211	5214*	5217*	5292*	5305*	5318	5321*	5324*	5350	5353*	5356*	5374	5375*
5485*	5498*	5511	5514*	5517*	5547	5550*	5553*	5571	5572*	5690*	5701*	5714
5718*	5721*	5839*	5850*	5863	5867*	5870*	5979*	5990*	6003	6007*	6010*	6037
6041*	6044*	6074	6077*	6080*	6098	6099*	6590	6627	6668	7503		
2943	2951	2967	2979	3037	3041	3053	3061	3111	3115	3127	3135	3192
3196	3208	3216	3273	3277	3289	3297	3352	3356	3368	3376	3434	3438
3450	3458	3524	3528	3539	3547	3635	3639	3651	3659	3716	3720	3732
3740	3813	3817	3828	3837	3916	3920	3931	3940	4020	4024	4035	4044
4124	4128	4139	4148	4231	4235	4246	4254	4357	4361	4372	4380	4476
4480	4491	4499	4571	4575	4587	4595	4661	4665	4677	4685	4751	4755
4767	4775	4841	4845	4857	4865	4932	4936	4948	4956	5044	5048	5060
5068	5158	5162	5174	5182	5265	5269	5281	5289	5458	5462	5474	5482
5659	5664	5677	5685	5808	5813	5826	5834	5948	5953	5966	5974	6714*
7268	7494*											
1055*	2333	2343	2472	2482	2610	2667	2721	2776	4739	4786		
1145*	1216*	6621										
1057*												
7342	7495*											
2261	2272	2288	7496*									
1066*	3571	3579	4278	4286	4404	4412	4523	4531	4606	4696	4786	4876
5389	5586											
1052*												
6509	7498*											
1168	1239	2174*										
1007*												
1035*												
1030*	2329*	2375*	2422*	2468*	2514*	2561*	2619*	2676*	2730*	2785*	2835*	2845
2928*	3019*	3098*	3179*	3260*	3339*	3421*	3499*	3620*	3703*	3790*	3893*	3997*

RKWC = 000002

4101*	4206*	4332*	4451*	4555*	4645*	4735*	4825*	4918*	5030*	5144*	5251*	5394
5444*	5591	5642*	5791*	5931*								
1028*	2326*	2333*	2340	2349*	2372*	2379*	2386	2395*	2419*	2426*	2433	2442*
2465*	2472*	2479	2488*	2511*	2518*	2525	2534*	2558*	2565*	2572	2581*	2616*
2622*	2628	2634*	2673*	2679*	2685	2691*	2727*	2733*	2739	2745*	2782*	2788*
2794	2800*	2832*	2836*	2842	2855*	2879	2885*	2925*	2929*	2944	2952	2968
3013*	3020*	3093*	3099*	3174*	3180*	3255*	3261*	3334*	3340*	3379	3416*	3422*
3461	3494*	3503*	3550	3575*	3576	3608*	3624*	3662	3698*	3704*	3743	3788*
3794*	3840	3891*	3897*	3943	3995*	4001*	4047	4099*	4105*	4151	4201*	4210*
4257	4282*	4283	4327*	4336*	4383	4408*	4409	4446*	4455*	4502	4527*	4528
4553*	4559*	4603	4618*	4619	4643*	4649*	4693	4708*	4709	4733*	4739*	4783
4798*	4799	4823*	4829*	4873	4888*	4889	4916*	4920*	5028*	5032*	5142*	5146*
5249*	5253*	5386	5442*	5446*	5583	5636*	5644*	5785*	5793*	5925*	5933*	6307*
1032*	2332*	2378*	2425*	2471*	2517*	2564*	2843	2880	2989	3380	3462	3551
3577	3663	3744	3841	3944	4048	4152	4258	4284	4384	4410	4503	4529
4604	4620	4694	4710	4784	4800	4874	4890	5387	5584			
1031*	2331*	2377*	2424*	2470*	2516*	2563*	2621*	2678*	2732*			
3177*	3219	3258*	3300	3337*	3419*	3502*	3622*	3701*	3792*	2787*	3096*	3138
4209*	4335*	4454*	4558*	4648*	4738*	4828*	5393	5590		3895*	3999*	4103*
1037*	2875											
1036*	2330*	2376*	2423*	2469*	2515*	2562*	2620*	2677*	2731*	2786*	3095*	3139
3176*	3220	3257*	3301	3336*	3418*	3501*	3623*	3700*	3793*	3896*	4000*	4104*
4208*	4334*	4453*	4557*	4647*	4737*	4827*	5392	5589				
1033*												
1041*												
1042*	4993	5106	5220	5328	5521	5724	5873	6013	6047	6106		
1034*	2846	2881	2990	3381	3463	3552	3578	3664	3745	3842	3945	4049
4153	4259	4285	4385	4411	4504	4530	4605	4621	4695	4711	4785	4801
4875	4891	5388	5585									
1038*	2327*	2336*	2337*	2373*	2382*	2383*	2420*	2429*	2430*	2466*	2475*	2476*
2512*	2521*	2522*	2559*	2568*	2569*	2617*	2624*	2625*	2674*	2681*	2682*	2728*
2735*	2736*	2783*	2790*	2791*	2833*	2838*	2839*	2926*	2932*	2933*	2936*	2937*
2982	3017*	3023*	3024*	3028*	3029*	3030*	3033*	3064	3094*	3102*	3103*	3106*
3107*	3175*	3183*	3184*	3187*	3188*	3256*	3264*	3265*	3268*	3269*	3335*	3343*
3344*	3347*	3348*	3417*	3425*	3426*	3429*	3430*	3498*	3506*	3507*	3517*	3518*
3562	3612*	3626*	3627*	3630*	3631*	3699*	3707*	3708*	3711*	3712*	3789*	3797*
3798*	3808*	3809*	3852	3892*	3900*	3901*</						

[illegible]

SEJ 0179

[illegible]

CZR6DB0 RK611 DSKLS CTRL PRTH
CZR6DB.P11 02-DEC-77 10:00MACY11 30(1096) 02-DEC-77 10:20 PAGE 181
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0180

	7275	7281	7286	7290	7295	7296	7298	7301	7305	7371	7373	7445	7485*
TYPERR 035326	6456*	6817											
TYPOC = 104402	2259	2270	2286	5771	6489	7175	7486*						
TYPON = 104404	7488*												
TYPOS = 104403	7487*												
T.ASOF 003156	2115*												
T.BA 003144	2110*	2845*	2861	5394*	5409	5591*	5606	7509	7538				
T.CS1 003140	2108*	2340*	2346	2386*	2392	2433*	2439	2479*	2485	2525*	2531	2572*	2578
	2628*	2631	2685*	2688	2739*	2742	2794*	2797	2842*	2852	2879*	2882	2944*
	2945	2952*	2953	2968*	2969	3379*	3385	3461*	3467	3550*	3553	3576*	3580
	3662*	3665	3743*	3749	3840*	3843	3943*	3946	4047*	4050	4151*	4154	4257*
	4260	4283*	4287	4383*	4386	4409*	4413	4502*	4505	4528*	4532	4603*	4609
	4619*	4623	4693*	4699	4709*	4713	4783*	4789	4799*	4803	4873*	4879	4889*
	4893	5386*	5400	5583*	5597	7506	7509	7513	7518	7524	7528	7535	
T.CS2 003150	2112*	2843*	2858	2880*	2888	2989*	3380*	3388	3462*	3470	3551*	3556	3577*
	3663*	3668	3744*	3752	3841*	3846	3944*	3949	4048*	4053	4152*	4157	4258*
	4263	4284*	4384*	4389	4410*	4503*	4508	4529*	4604*	4612	4620*	4694*	4702
	4710*	4784*	4792	4800*	4874*	4882	4890*	5387*	5403	5584*	5600	7509	7513
	7518	7524	7528	7535									
T.DA 003146	2111*	3138*	3140	3219*	3221	3300*	3302	5393*	5418	5590*	5615	7522	7538
T.DB 003162	2117*	2875*	2894	7516									
T.DCYL 003160	2116*	3139*	3143	3220*	3224	3301*	3305	5392*	5415	5589*	5612	7522	7538
T.DS 003152	2113*												
T.ECPS 003172	2121*												
T.ECPT 003174	2122*	4993*	4994	5106*	5108	5220*	5222	5328*	5329	5521*	5522	5724*	5726
	5873*	5875	6013*	6015	6047*	6049	6106*	6107	7533	7541			
T.ER 003154	2114*	2846*	2867	2881*	2891	2990*	3381*	3391	3463*	3473	3552*	3559	3578*
	3664*	3671	3745*	3755	3842*	3849	3945*	3952	4049*	4056	4153*	4160	4259*
	4266	4285*	4385*	4392	4411*	4504*	4511	4530*	4605*	4615	4621*	4695*	4705
	4711*	4785*	4795	4801*	4875*	4885	4891*	5388*	5406	5585*	5603	7509	7513
	7518	7524	7528	7535									
T.MR1 003164	2118*	2982*	2984	3064*	3066	3562*	3563	3852*	3853	3955*	3956	4059*	4060
	4163*	4164	4269*	4270	4395*	4396	4514*	4515	6639*	6640	6653*	6654	6680*
	6681	6693*	6694	7503	7520	7531							
T.MR2 003166	2119*	2341*	2352	2387*	2398	2434*	2445	2480*	2491	2526*	2537	2573*	2584
	2629*	2637	2686*	2694	2740*	2748	2795*	2803	7506				
T.MR3 003170	2120*	2342*	2355	2388*	2401	2435*	2448	2481*	2494	2527*	2540	2574*	2587
	2630*	2640	2687*	2697	2741*	2751	2796*	2806	7506				
T.SPAR 003176	2123*												
T.WC 003142	2109*	2844*	2864	5395*	5412	5592*	5609	7509	7538				
UFE = 000400	1086*												
UNLOAD= 000007	1050*												
UNS = 040000	1111*												
UPE = 020000	1091*												
VRCHDR 055174	3613*	3614	3616*	3619*	3640	7527	8933*						
VV = 000100	1122*												
WCE = 040000	1092*												
WLE = 004000	1108*												
WRDATA= 000023	1056*	2379	2389	2518	2528	2836	2847	2929	2938	3020	3099	3180	3261
	3340	3382	3422	3464	3503	3512	3600	3624	3704	3746	3794	3803	3897
	3906	4001	4010	4105	4114	4210	4219	4336	4345	4455	4464	4559	4606
	4649	4696	4920	5032	5146	5253	5389	5446	5586	5644	5793	5933	
	2158*	2872*	2897*	7516									
WRDCNT 003264	1058*												
WRHEAD= 000027	1125*												
WRL = 004000	4967	4973	4991	5079	5085	5104	5193	5199	5219	5300	5306	5325	5357

530 0181

CROSS REFERENCE TABLE -- USER SYMBOLS

SETABL 001234

[illegible]

[illegible]

CZRBDBO RV611 DSCLS CTRL PRTH
CZFBDB.P11 02-DEC-77 10:00MACY11 30(1046) 02-DEC-77 10:20 PAGE 184
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0183

SPWRDN 042232	2201	7427*	7442														
SPWRUP 042260	7428	7436*															
SQJES 001210	1326*	6843	6923	7227	7298	7315	7373	7376									
SROCHK 041400	7240*	7494															
SRODEC= ***** U	7497																
SROLIN 041470	7263*	7495															
SROCT 041776	7337*	7496															
SRL - 000010	7256*																
SRL XE 042174	7409*	7498															
SRTNAD 034074	6157*																
SR2A = ***** U	7499																
SSAVRE 042136	7393*	7497															
SSCOPE 034646	2195	6334*															
SSETUP= 000137	2171*	2194	2195	2197	2199	2201	2203	2204	2205	2207	2235	2238	6122				
	6335	6804	6830	6838	7110	7115	7116	7146	7322								
SSTUP = 177777	2171*																
SSVLAD 035106	6345	6384*															
SSVPC = 000220	1175*	1180	1246*	1251													
SSWR = 167400	884*	896	900	901	902	903	904	905	906	907	1323	1324	1325				
	2204	2205	2207	2208	2324	2370	2417	2463	2509	2556	2606	2663	2717				
	2772	2830	2923	3011	3084	3165	3246	3332	3414	3492	3598	3696	3786				
	3889	3993	4097	4199	4325	4444	4551	4641	4731	4821	4914	5026	5140				
	5246	5439	5633	5761	5901	5921	6117	6123	6150	6156	6158	6326	6327				
	6328	6329	6330	6336	6348	6350	6351	6364	6365	6366	6373	6374	6375				
	6387	6390	6393	6795	6796	6797	6798	6799	6808	6815	6827	6831	6843				
SSWREG 001236	1346*	2228															
SSWRM= 000000	907	908	6330	6331	6354												
SSWOST 035160	6360	6394*															
STESTN 001220	1337*	5770	6385*	6457*	6458*	7502	7503	7506	7509	7513	7516	7518	7520				
	7522	7524	7528	7531	7533	7535	7541										
STIMES 001200	1323*	2204*	2324*	2370*	2417*	2463*	2509*	2556*	2606*	2663*	2717*	2772*	2830*				
	2923*	3011*	3084*	3165*	3246*	3332*	3414*	3492*	3598*	3696*	3786*	3889*	3993*				
	4097*	4199*	4325*	4444*	4551*	4641*	4731*	4821*	4914*	5026*	5140*	5246*	5439*				
	5633*	5761*	5766*	5901*	5921*	6123*	6373*	6380	6383*	6393							
STKB 001146	1308*	7070	7091	7102	7127	7155	7182										
STKCN 040540	7071*	7086*	7116	7133*	7247	7249*											
STKINT 040550	2181	7086*	7107	7168													
STKQEN= 040547	7075*	7141	7252														
STKQIN 040542	7072*	7087*	7088	7139*	7140*	7141	7143*										
STKQOU 040544	7073*	7088*	7250	7251*	7252	7254*											
STKQSP 040546	7074*	7087	7143	7254													
STKS 02. 4	1307*	7070	7092*	7123*	7125	7131*	7153	7169*	7179	7191*	7211*						
STKSRV 02. 3	7089	7102*															
STMPD 02. 3	1315*	2262*	2264	2266	2273*	2275	2277	2289*	2291	2293	2295*	2296*	2297*				
	2298*	2299*	2301														
STMP1 001162	1316*																
STMP2 001164	1317*																
STMP3 001166	1318*																
STMP4 001170	1319*	3603*	3614	3616	3619	3675*											
STMP5 001172	1320*																
STMP6 001174	1321*																
STMP7 001176	1322*																
STN = 000052	885*	896	2313	2324*	2350	2356	2359	2370*	2396	2402	2405	2417*	2443				
	2449	2452	2463*	2489	2495	2498	2509*	2535	2541	2544	2556*	2582	2588				
	2593	2606*	2651	2663*	2705	2717*	2759	2772*	2811	2817	2830*	2856	2886				
	2905	2923*	2985	2987	2993	3011*	3067	3070	3084*	3151	3165*	3232	3246*				

BSETST	1278#	3333	3415	3697											
BYPASS	1278#	5765													
CLRMSG	1278#	2464	2510	2557											
CLRPSW	1278#														
COMMEN	1018#														
DDOGEN	8417#	8418	8419	8420	8421	8422	8423	8424	8425	8426	8427	8428	8429	8430	8431
	8432	8433	8434	8435	8436	8437	8438	8439	8440	8441	8442	8443	8444	8445	8446
	8447	8448	8449	8450	8451	8452	8453	8454	8455	8456	8457	8458	8459	8460	8461
	8462	8463	8464	8465	8466	8467	8468	8469	8470	8471	8472	8473	8474	8475	8476
	8477	8478	8479	8480	8481	8482	8483	8484	8485						
ECCTST	1278#														
ENDCOM	1018#	5027	5141												
ERROR	912#	2348	2354	2357	2394	2400	2403	2441	2447	2450	2487	2493	2496	2533	2539
	2542	2580	2586	2589	2633	2639	2642	2690	2696	2699	2744	2750	2753	2799	2805
	2808	2854	2860	2863	2866	2869	2884	2890	2893	2896	2986	2991	3068	3142	3145
	3223	3226	3304	3307	3387	3390	3393	3469	3472	3475	3555	3558	3561	3565	3582
	3667	3670	3673	3751	3754	3757	3845	3848	3851	3855	3948	3951	3954	3958	4052
	4055	4058	4062	4156	4159	4162	4166	4262	4265	4268	4272	4289	4388	4391	4394
	4398	4415	4507	4510	4513	4517	4534	4611	4614	4617	4625	4701	4704	4707	4715
	4791	4794	4797	4805	4881	4884	4887	4895	4968	4974	4992	4996	5080	5086	5105
	5110	5194	5200	5219	5224	5301	5307	5326	5331	5358	5377	5402	5405	5408	5411
	5414	5417	5420	5494	5500	5519	5524	5555	5574	5599	5602	5605	5608	5611	5614
	5617	5697	5703	5723	5728	5846	5852	5872	5877	5986	5992	6012	6017	6046	6051
	6082	6101	6109	6292											
ESCAPC	1018#														
GETPRI	1018#														
GETSWR	1018#	2238#													
HORERR	1278#	4552	4642	4732	4822										
HORGEN	1278#	2607	2664	2718	2773										
INCHD	1278#	3085	3166	3247											
LDLPER	1278#	2611	2668	2722	2777	3088	3169	3250	3604						
MSG	2313#	2315	2359#	2361	2405#	2407	2452#	2454	2498#	2500	2544#	2546	2593#	2595	2651#
	2653	2705#	2707	2759#	2761	2817	2819	2905#	2907	2993#	2995	3070#	3072	3151#	3153
	3232#	3234	3313#	3315	3395#	3397	3477#	3479	3584#	3586	3679#	3681	3759#	3761	3862#
	3864	3965#	3967	4069#	4071	41									

[illegible]

CZR6DBD RK611 DSKLS CTRL PRT4 MACY11 30(1046) 02-DEC-77 10:20 PAGE 189
CZR6DB.P11 02-DEC-77 10:00 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0187

. ABS. 057232 000

ERRORS DETECTED: 0

RM03:CZR6DB, RM03:CZR6DB.SEG/SOL/CRF/NL:TOC/DOC=RM03:CZR6DB.P11
RUN-TIME: 31 26 2 SECONDS
RUN-TIME RATIO: 1188/60=19.6
CORE USED: 34K (67 PAGES)

DOCUMENT PAGES: 187